УЯЗВИМОСТИ VIEWSTATE

ТИМУР ЮНУСОВ, POSITIVE TECHNOLOGIES

1. ОБЩИЕ СВЕДЕНИЯ О VIEWSTATE

«Состояние представления — это метод, который платформа веб-страниц ASP.NET использует для сохранения значений страницы и элемента управления между циклами обработки. При отображении разметки HTML страницы, текущее состояние страницы и значения должны сохраняться во время обратного запроса, сериализуясь в строки в кодировке base64. Затем данные сведения помещаются в скрытое поле или поля состояния представления».

MSDN

«Что делает ViewState?

- Сохраняет данные элементов управления по ключу, как хэш-таблица.

- Отслеживает изменения состояния ViewState'а.

- Сериализирует и десериализирует сохраненные данные в скрытое поле на клиенте.

- Автоматически восстанавливает данные на postback'ах.»

Из статьи разработчика ASP.NET о механизмах ViewState

Если еще проще, то ViewState – это скрытый параметр в HTML-форме, который передает серверу текущую структуру содержимого страницы. Пример использования: сохранение значений полей формы на странице во время постраничного пролистывания списков.

Несмотря на то, что существуют и активно используются методы избавления от ViewState и работы без него (в основном, с помощью СУБД), эта технология по умолчанию включена в ASP.NET и часто ее используют бездумно.

«Еще более важно понимать, чего ViewState НЕ делает.

Что не делает ViewState?

- ViewState не сохраняет автоматически состояние полей класса (скрытых, защищенных или открытых).

- ViewState не запоминает какую-либо информацию при загрузке страницы (только postback'и).

- ViewState не снимает необходимость загружать данные при каждом запросе.

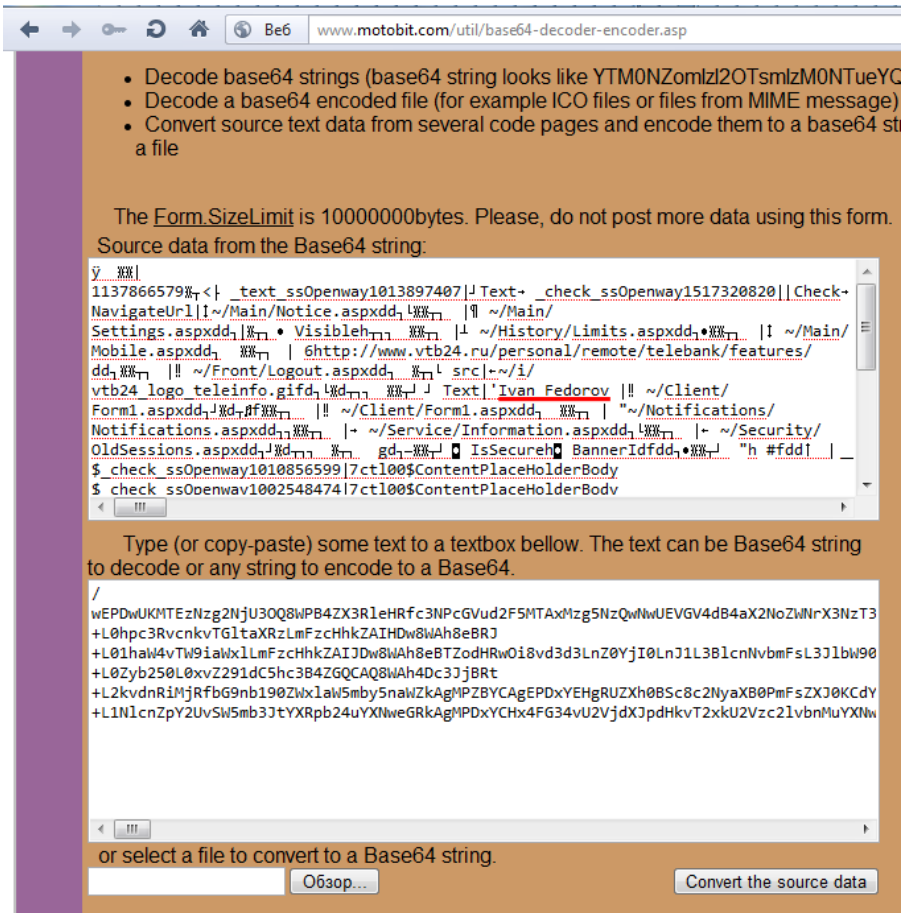
- ViewState не отвечает за загрузку данных, которые были отправлены на сервер, например, введенных в текстовое поле (хотя ViewState и играет здесь важную роль).»

Из статьи разработчика ASP.NET о механизмах ViewState

Это, естественно, приводит к более глубоким проблемам – отсутствию фильтрации и непониманию того, как правильно должно работать веб-приложение.

Разработчики подчас думают, что раз ViewState – это сериализованная структура, да еще и «зашифрованная» base64, то никакой злоумышленник не сможет добраться до ее содержимого...

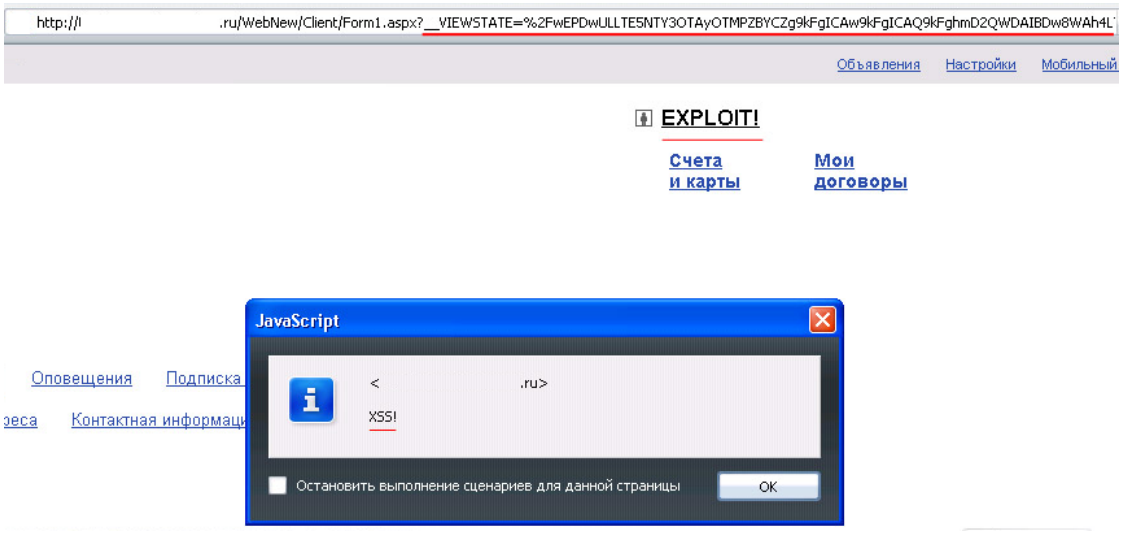
На самом же деле, если шифрование и проверка целостности (MAC) отключены, все гораздо проще. Декодируем base64:



Открываем декодированный файл в hex-редакторе и видим, что перед любой строковой переменной идут байты, указывающие на длину этой строки (количество байт зависит от длины строки: для строки <128 байт под длину переменной будет выделен один байт).

Offset	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	
00001280	64	64	02	0B	0F	0F	16	02	1F	1E	05	13	7E	2F	46	72	dd.....~/Fr
00001296	6F	6E	74	2F	4C	6F	67	6F	75	74	2E	61	73	70	78	64	ont/Logout.aspxd
00001312	64	02	01	0F	16	02	1E	03	73	72	63	05	1B	7E	2F	69	d.....src..~/i
00001328	2F	76	74	62	32	34	5F	6C	6F	67	Байт	размера				65	/vtb24_logo_tele
00001344	69	6E	66	6F	2E	67	69	66	64	02	переменной.					02	info.gifd...d...
00001360	01	0F	0F	16	04	1E	04	54	65	78	74	05	27	3C	73	63	.....Text.'<sc
00001376	72	69	70	74	3E	61	6C	65	72	74	28	27	58	53	53	21	ript>alert('XSS!
00001392	27	29	3C	2F	73	63	72	69	70	74	3E	54	45	53	54	20	')</script>TEST
00001408	54	45	53	54	1F	1E	05	13	7E	2F	43	6C	69	65	6E	74	TEST....~/Client
00001424	2F	46	6F	72	6D	31	2E	61	73	70	78	64	64	02	04	0F	/Form1.aspxdd...
00001440	64	16	0E	66	0F	0F	16	02	1F	1E	05	13	7E	2F	43	6C	d..f.....~/Cl
00001456	69	65	6E	74	2F	46	6F	72	6D	31	2E	61	73	70	78	64	ient/Form1.aspxd
00001472	64	02	01	0F	0F	16	02	1F	1E	05	22	7E	2F	4E	6F	74	d.....~/Not
00001488	69	66	69	63	61	74	69	6F	6E	73	2F	4E	6F	74	69	66	ifications/Notif
00001504	69	63	61	74	69	6F	6E	73	2E	61	73	70	78	64	64	02	ications.aspxdd.
00001520	02	0F	0F	16	02	1F	1E	05	1A	7E	2F	53	65	72	76	69	.....~/Servi
00001536	63	65	2F	49	6E	66	6F	72	6D	61	74	69	6F	6E	2E	61	ce/Information.a
00001552	73	70	78	64	64	02	03	0F	0F	16	02	1F	1E	05	1B	7E	spxdd.....~
00001568	2F	53	65	63	75	72	69	74	79	2F	4F	6C	64	53	65	73	/Security/OldSes
00001584	73	69	6F	6E	73	2E	61	73	70	78	64	64	02	04	0F	64	sions.aspxdd...d
00001600	16	02	02	01	0F	16	02	1F	1F	67	64	02	06	0F	0F	16	.....gd.....
00001616	04	1E	08	49	73	53	65	63	75	72	65	68	1E	08	42	61	...IsSecureh..Ba
00001632	6E	6E	65	72	49	64	66	64	64	02	07	0F	0F	16	04	1F	nnerIdfdd.....
00001648	22	68	1F	23	66	64	64	18	01	05	1E	5F	5F	43	6F	6E	"h.#fdd...__Con
00001664	74	72	6F	6C	73	52	65	71	75	69	72	65	50	6F	73	74	trolsRequirePost
00001680	42	61	63	6B	4B	65	79	5F	5F	16	0F	05	37	63	74	6C	BackKey__...7ctl
00001696	30	30	24	43	6F	6E	74	65	6E	74	50	6C	61	63	65	48	00$ContentPlaceH
00001712	6F	6C	64	65	72	42	6F	64	79	24	5F	63	68	65	63	6B	olderBody$_check

В авторитетных источниках указано, что алгоритм работы сериализации/десериализации в ASP.NET<2.0 – LosFormatter, а в >=2.0 – ObjectStateFormatter. Поэтому, для того чтобы изменить эту переменную, требуется вычислить длину новой строки, перезаписать строку, перезаписать байт (байты) с длиной строки, провести обратное кодирование base64 и подставить во __VIEWSTATE.



2. УЯЗВИМОСТИ И АТАКИ

Если помножить все вышеизложенное на низкий уровень знаний среднестатистического специалиста о правильной организации безопасности веб-приложения, становится понятно, что возможно проведение атак, использующих следующие уязвимости:

- Межсайтовое выполнение сценариев (Cross-Site Scripting, XSS)
- Подмена содержимого (Content Spoofing)
- Внедрение операторов SQL (SQL Injection)
- Утечка информации (Information Leakage)
- Логические атаки (Logical Attacks)
- Уязвимости самого ViewState
- Прочие уязвимости

2.1. Межсайтовое выполнение сценариев, Подмена содержимого

Возможность изменения содержимого HTML-страницы следует из главного предназначения ViewState – «Сохранение значений страницы и элементов управления». Если данные из ViewState, помещаемые в тело HTTP-ответа, не проходят необходимую фильтрацию, то мы получаем «Подмену содержимого» и/или «Межсайтовое выполнение сценариев».

Уязвимая конфигурация:

EnableViewStateMac=false

ViewStateEncryptionMode=never|auto

(зависит от RegisterRequiresViewStateEncryption)

ViewStateUserKey=EMPTY

2.2. Утечка информации, Логические атаки

Если разработчик не использует шифрование параметра VIEWSTATE (Securing View State), злоумышленник может декодировать структуру параметра VIEWSTATE и извлечь оттуда конфиденциальные данные. Если же не используется и проверка целостности (MAC), то злоумышленник получает возможность изменить параметры, которые способны повлиять на логику работы веб-приложения, что может повлечь за собой «Обход аутентификации» (Authentication Bypass), «Обход авторизации» (Authorization Bypass), «Злоупотребление функциональными возможностями» (Abuse of Functionality) и т.д.

Уязвимая конфигурация:

ViewStateEncryptionMode=never|auto

EnableViewStateMac=false|true

2.3. Атаки на ViewState

Атакам подвержен и сам механизм ViewState. Например, в сентябре 2010 г. была опубликована уязвимость, позволяющая расшифровать ViewState, зашифрованный алгоритмом AES, с помощью отправки множества запросов серверу и отслеживания разных кодов ошибок [1].

Также в старых версиях (1.0, 1.1) возможны атаки типа «Отказ в Обслуживании» (Denial Of Service, DoS) (в случае, если VIEWSTATE не шифруется) и Replay-атаки (на зашифрованный VIEWSTATE). Последние представляют собой атаки на криптографический протокол, когда перехватываемый пакет можно отправить еще раз и он будет корректно воспринят, что вызовет нарушения в работе алгоритма. Об этих атаках писал Michal Zalewski еще в 2005 году [2].

2.4. Прочие уязвимости

Переменные, записанные в структуру ViewState, так же как и обычные переменные, передаваемые методами GET/POST/COOKIES, могут и должны проверяться на все остальные уязвимости, присущие веб-приложениям, такие как «Внедрение SQL-кода» (SQL Injection), «Выполнение команд ОС» (OS Commanding), а также на другие уязвимости классов «Выполнение кода», «Разглашение информации» и проч.

Уязвимая конфигурация:

EnableViewStateMac=false

ViewStateEncryptionMode=never|auto

(зависит от RegisterRequiresViewStateEncryption)

3. ЗАЩИТА

3.1. ENABLEVIEWSTATEMAC

Значение по умолчанию: TRUE

Начиная с версии: 1.0

Включает MAC (Machine Authentication Check) – проверку значения параметра VIEWSTATE с помощью контрольной суммы.

Необходимо задать свойство EnableViewStateMac="True" в элементе Page.

Кроме того, необходимо для активации настроить свойства validationKey и validation элемента machineKey.

Поддерживаются встроенные алгоритмы шифрования: SHA1, MD5, 3DES, AES, HMACSHA256, HMACSHA384, HMACSHA512.

3.2. ViewStateEncryptionMode

Значение по умолчанию: Auto

Начиная с версии: 2.0

Позволяет шифровать параметр VIEWSTATE одним из следующих алгоритмов: DES, 3DES или AES.

Для активации необходимо настроить свойства decryptionKey и decryption элемента machineKey.

3.3. ViewStateUserKey

Значение по умолчанию: EMPTY

Начиная с версии: 1.1

Не все знают, что ViewState позволяет защитить не только себя от подмены, но и все приложение от CSRF с помощью параметра ViewStateUserKey.

ViewStateUserKey – это лишь механизм защиты. Обеспечивать случайность и непредсказуемость этого параметра – задача разработчика.

Необходимо задать свойство ViewStateUserKey="String" в элементе Page.

4. ЗАКЛЮЧЕНИЕ

Из разделов 2 и 3 видно, что в конфигурациях по умолчанию ViewState надежно защищен от уязвимостей ненулевого дня, однако очень часто, «намучившись» с постоянно выпадающими ошибками о нарушении целостности, неправильных аргументах и т.д., разработчики просто отключают «ключи, вызывающие ошибки», вместе с тем оставляя приложение незащищенным от различных атак.

Однако если веб-приложение настроено верно, вероятность возникновения ошибок и тем более уязвимостей можно свести к 0.

5. ССЫЛКИ

[1] http://weblogs.asp.net/scottgu/archive/2010/09/18/important-asp-net-security-vulnerability.aspx

[2] http://seclists.org/bugtraq/2005/May/27

[6] О КОМПАНИИ

«Позитив Текнолоджиз» (Positive Technologies) - лидирующая компания на рынке информационной безопасности.

Основные направления деятельности компании:

- разработка систем комплексного мониторинга информационной безопасности (XSpider, MaxPatrol);
- оказание консалтинговых услуг в области ИБ;
- предоставление сервисных услуг в области ИБ;
- развитие ведущего российского портала по ИБ Securitylab.ru.

Компания «Позитив Текнолоджиз» (Positive Technologies) – это команда квалифицированных разработчиков и консультантов. Эксперты компании имеют большой практический опыт, являются членами международных организаций, активно участвуют в развитии отрасли.