OWASP
Open Web Application Security Project

Standard

Mobile AppSec Verification

Version 1.1

(Russian Translation)

Project Leaders: Bernhard Mueller and Sven Schleier

This is document is currently under development. We welcome contributions and industry feedback. Contact us on the OWASP Mobile Testing Guide Slack channel:

https://owasp.slack.com/messages/project-mobile_omtg/

You can sign up here:

http://owasp.herokuapp.com/

The Open Web Application Security Project (OWASP) is a worldwide free and open community focused on improving the security of application software. Our mission is to make application security "visible", so that people and organizations can make informed decisions about application security risks. Every one is free to participate in OWASP and all of our materials are available under a free and open software license. The OWASP Foundation is a 501c3 not-for-profit charitable organization that ensures the ongoing availability and support for our work.

Вступительное слово от Bernhard Mueller, OWASP Mobile Project

Технологические революции могут происходить молниеносно: менее десятилетия назад смартфоны были громоздкими устройствами с клавиатурой, и являлись дорогими игрушками для некоторых компаний. Сегодня же смартфоны прочно вошли в нашу жизнь. Мы доверяем им информацию, навигацию, коммуникацию, а также они являются неотъемлемой составляющей как бизнеса, так и жизни в целом.

Каждая новая технология приносит новые риски безопасности, и попытка поспеть за этими изменениями и есть один из основных вызовов, которые стоят перед индустрией безопасности. Сторона защиты всегда на несколько шагов позади. Например, обычной реакцией многих была бы попытка применить старый подход: смартфоны- это маленькие компьютеры и мобильные приложения- обычное ПО, значит требования безопасности абсолютно такие же? Но так это не работает. Операционные системы смартфонов отличаются от ОС компьютеров и мобильные приложения отличаются от веб приложений. Например, классический метод поиска вирусов, основанный на подписях, не представляется возможным в современных мобильных средах: не только потому что это не соотносится с моделью распространения мобильного ПО, но еще и по тому, что это технически невозможно из- за ограничений среды выполнения приложений(sandbox). Также, некоторые классы уязвимостей, такие как: переполнение буфера и XSS менее релевантны к мобильным приложения, нежели к компьютерным программам и веб приложения(есть исключения).

С течением времени, наша индустрия получила больше опыта в борьбе с мобильными угрозами. Как оказалось, мобильная безопасность- это про сохранение данных: приложения хранят нашу личную информацию, картинки, записи, заметки, учетные данные, информацию о бизнесе, местоположения и многое другое. Приложения играют роль клиентов, подключающих нас к сервисам, которыми мы используем ежедневно, а также они играют роль коммуникационных хабов, обрабатывающих каждое сообщение, которым мы обмениваемся. Скомпрометировав телефон человека, вы получите неограниченный доступ к его личной жизни. Когда мы берем во внимание тот факт, что мобильные устройства легко теряются или похищаются и мобильные вирусы сейчас активно развиваются, необходимость защиты данных становиться еще более очевидной.

Следовательно, стандарты безопасности мобильных приложений должны сконцентрироваться на том, как они обрабатывают, хранят и защищают чувствительную информацию. Даже не смотря на тот факт что современные мобильные операционные системы такие как iOS и Android предоставляют хорошие API для защищенного хранения данных и их передачи, они должны быть реализованы и использованы правильно для того, чтобы быть эффективными. Хранение данных, коммуникация внутри приложения, правильное использование криптографических API и защищенный сетевой обмен - это только некоторые из аспектов, которые требуют внимательного рассмотрения.

Важный вопрос,в котором необходимо достич консенсуса в нашей индустрии- это как далеко должны заходить специалисты в вопросах защиты конфиденциальности и целостности данных. Например, многие из нас согласятся, что мобильное приложение должно проверять сертификат сервера во время соединения TLS. Но что насчет SSL pinning? Невыполнение этого требования ведет к уязвимости? Должно ли это быть требованием, если приложение обрабатывает чувствительную информацию или же, может быть, это контрпродуктивно? Нужно ли нам зашифровывать данные, хранящиеся в SQlite, несмотря на то, что ОС выполняет приложение в песочнице(sandboxing)? То что

подходит одному приложению может быть совсем нереалистично для другого. MASVS - это попытка стандартизации требований, используя уровни проверок, которые подходят для разных моделей угроз.

Появление вредоносных программ, работающих как root, а также инструменты удаленного управления заставляют осознать тот факт, что мобильные ОС, как таковые, имеют недостатки, которыми могут воспользоваться злоумышленники. Стратегии контейнеризации используются все повсеместнее, чтобы дополнительная защита чувствительных данных и предотвращение tampering(подделывание/искажение/вмешательство) на стороне клиента стало возможным. Это место, где все усложняются. Меры безопасности, обеспеченные на уровне железа, и решения контейнеризации на уровне ОС, такие как: Android for Work и Samsung Knox, существуют, но они бывают недоступны на некоторых устройствах. В качестве костыля, возможно реализовать программные меры защиты, но, к сожалению, не существует стандартов или процессов тестирования для подтверждения такого способа защиты.

В результате, отчеты по тестированию информационной безопасности мобильных приложений повсюду. Например, некоторые тестировщики приложений Android сообщают о недостаточной обфускации или о нахождении root процесса как о "недостатке безопасности". С другой стороны, такие метрики как шифрование строк, обнаружение отладчика или обфускации потока управления не считаются необходимыми. Однако, бинарный взгляд на вещи не имеет смысла, потому что отказоустойчивость не является бинарным предположением: это зависит от конкретной модели угроз клиентского приложения, от которой пытаются защититься. Защита программного уровня не является бесполезной, но в конечном итоге ее можно обойти, так что она не должна быть использована как замена контрмер безопасности.

Общая цель MASVS - предложить фундамент для безопасности мобильных приложений(MASVS- L1), вместе с тем, позволить включить усиленные меры защиты (MASVS-L2) и защиту против угроз на стороне клиента (MASVS-R). MASVS предназначен для достижения следующих целей:

- Предоставить требования для системных архитекторов и разработчиков, желающих создать безопасные мобильные приложения.
- Предложить промышленный стандарт, с которым можно сверять аудит безопасности мобильных приложений.
- Прояснить роль механизмов защиты ПО в мобильной безопасности и предоставить требования для проверки их эффективности.
- Предоставить конкретные рекомендации, для разных уровней безопасности, которые зависят от конкретного варианта использования.

Мы пониманием что 100% консенсуса в отрасли невозможно достичь. Однако, мы надеемся что MASVS будет полезен в предоставлении руководства на всех этапах разработки и тестировании мобильных приложений. Как открытый стандарт, MASVS, будет развиваться с течением времени и мы рады любым предложениям или вкладам в развитием проекта.

Вступительное слово

О стандарте

Добро пожаловать в стандарт проверки безопасности мобильных приложений (MASVS) 1.1. MASVS- это усилие сообщества в создании библиотеки с требованиями безопасности, необходимыми для проектирования, разработки и тестирования безопасных мобильных приложений на iOS и Android.

MASVS - кульминация усилий сообщества и обратной связи от индустрии. Мы надеемся на развитие стандарта со временем и мы приветствуем обратную связь от сообщества. Лучший вариант связаться с нами- это через OWASP Mobile Project Slack channel.

Аккаунты можно создать по этому адресу.

Авторское право и лицензия

Руководитель проекта	Главные авторы	Авторы и рецензенты
Bernhard Mueller, Sven Schleier	Bernhard Mueller	Abdessamad Temmar, Abhinav Sejpal, Alexander Antukh, Anant Shrivastava, Ben Gardiner, Francesco Stillavato, Jeroen Willemsen, Manuel Delgado, Nikhil Soni, Prabhant Singh, Roberto Martelloni, Stephen Corbiaux, Stephen Reda, Sjoerd Langkemper, Stefaan Seys, Sven Schleier and Yogesh Sharma
Перевод на русский язык	Gall Maxim	Oprya Egor, Chelnokov Vladislav, Tereshin Dmitrii, Bachevsky Artem, Mesheryakov Aleksey, Ratchenko Denis

Работа над документом была начата как ответвление OWASP Application Security Verification Standard, написанного Jim Manico.

The Mobile Application Security Verification Standard

MASVS может быть использован для подтверждения определенного уровня уверенности в безопасности мобильных приложений. Требования были сформированы для достижения следующих целей:

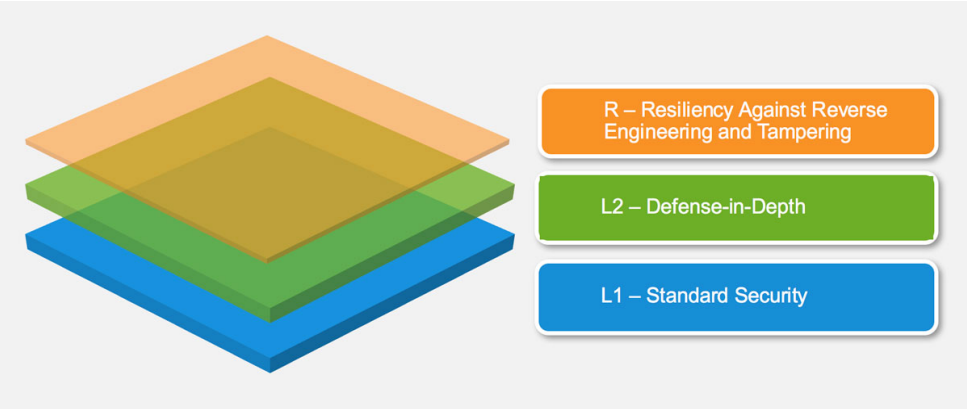
- Использование в качестве показателя: для предоставления стандарта безопасности, с которым разработчики и владельцы уже существующих мобильных приложений могут сравнивать свои продукты.
- Использование в качестве руководства: для предоставления рекомендаций во время всех этапов разработки и тестирования.
- Использование во время закупок: как основание для проверки безопасности мобильного приложения.

Модель безопасности мобильных приложений

MASVS определяет 2 строгих уровня проверки безопасности (L1 и L2), а также набор гибких требований для обеспечения защиты, затрудняющую обратную разработку/реверс инжиниринг(reverse engineering) (MASVS-R), то есть адаптируются к конкретной модели угроз приложения. MASVS-L1 и MASVS-L2 содержат общие требования к безопасности и рекомендуются для всех мобильных приложений (L1) и для приложений, которые обрабатывают очень чувствительные данные(L2). MASVS-R охватывает дополнительные меры защиты, которые могут применяться в случае, если предотвращение атак на стороне клиента является заложенным свойством архитектуры.

Выполнение требований MASVS-L1 приводит к созданию безопасного приложения, которое следует всем лучшим практикам безопасности и не подверженно часто встречающимся уязвимостям. MASVS-L2 добавляет эшелонированность защиты, например SSL pinning, приводящую к отказоустойчивости приложения при более ухущренных атаках, при условии, что безопасность мобильной операционной системы не скомпрометирована, а конечный пользователь не рассматривается как потенциальный злоумышленник. Выполнение всех или подмножества требований защиты программного обеспечения в MASVS-R помогает препятствовать конкретным угрозам на стороне клиента, когда конечный пользователь является злоумышленником и/или мобильная ОС скомпрометирована.

Обратите внимание, что меры защиты программного обеспечения, перечисленные в MASVS-R и OWASP Mobile Testing Guide, можно обойти и они никогда не должны использоваться в качестве замены управления безопасности мобильного приложения. Вместо этого они должны использоваться как специализированное дополнение, направленное на локализацию конкретных угроз, в мобильном приложение, которое удовлетворяет требования MASVS L1 или L2.



Структура документа

Первая часть MASVS содержит описание модели безопасности и доступных уровней проверок, а затем рекомендации о том, как использовать стандарт на практике. Подробные требования безопасности, а также сопоставление их с уровнем верификации, перечислены во второй части. Требования были сгруппированы в восемь категорий (от V1 по V8) по принадлежности к технической цели/области. Следующая номенклатура используется на протяжении всего MASVS и MSTG:

- *Категория требований:* MASVS-Vx, например, MASVS-V2: хранение и конфиденциальность данных
- *Требование:* MASVS-Vx.y, например, MASVS-V2.2: «В журналы приложений не записываются конфиденциальные данные».

MASVS-L1: Стандартная безопасность

Мобильное приложение, которое удовлетворяет MASVS-L1, придерживается лучших практик обеспечения безопасности мобильных приложений. Данный уровень верификации предоставляет основные требования с точки зрения качества кода, обработки конфиденциальных данных и взаимодействия с мобильной средой. Для подтверждения соответствия приложения данному уровню безопасности должен быть развернут процесс тестирования. Этот уровень подходит для всех мобильных приложений.

MASVS-L2: Глубокая эшелонированность защиты

MASVS-L2 предлагает расширенные средства верификации безопасности, выходящие за рамки стандартных. Чтобы соответствовать L2, должна существовать модель угроз, и безопасность должна быть неотъемлемой частью архитектуры и дизайна приложения. Этот уровень подходит для приложений, которые обрабатывают чувствительные данные, такие как мобильный банкинг.

MASVS-R: Устойчивость к обратному проектированию и фальсификации(tampering)

Приложение имеет самую современную безопасность и также устойчиво к конкретным, четко определенным атакам на стороне клиента, таким как подмена/фальсификация(tampering), модификация или обратное проектирование для извлечения чувствительного кода или данных. Такое приложение либо использует

аппаратные средства безопасности, либо достаточно надёжные и проверяемые методы программной защиты. MASVS-R применим к приложениям, которые обрабатывают высокочувствительные данные и могут служить средством защиты интеллектуальной собственности или защиты от несанкционированного доступа.

Рекомендованное использование

Приложения могут быть проверены на соответствие MASVS L1 или L2, основываясь на предварительной оценке рисков и общего понимания требуемого уровня безопасности. L1 применим ко всем мобильным приложениям, в то время как L2 обычно рекомендуется для приложений, которые обрабатывают более чувствительные данные и/или реализуют чувствительную функциональность. MASVS-R (или его части) может быть применен для верификации отказоустойчивости от конкретных угроз, таких как переупаковка или извлечение конфиденциальных данных, *или же*, для осуществления надежной проверки безопасности.

Таким образом, доступны следующие типы верификации:

- MASVS-L1
- MASVS-L1 + R
- MASVS-L2
- MASVS-L2 + R

Различные комбинации отражают различные уровни безопасности и отказоустойчивости. Цель состоит в том, чтобы обеспечить гибкость: например, мобильная игра может не гарантировать добавление средств безопасности MASVS-L2, таких как двухфакторная аутентификация по причинам удобства использования, но обладает потребностью бизнеса в предотвращении фальсификации.

Какой тип верификации выбрать

Соответсвие требованиям MASVS L2 повышает безопасность и в то же время увеличивает стоимость разработки и потенциально делает хуже опыт конечного пользователя (классический компромисс). В общем случае, L2 следует использовать для приложений, когда это имеет смысл с точки зрения риска и стоимости (т. е. когда потенциальная потеря, вызванная компромиссной реализацией конфиденциальности или целостности, выше, чем затраты, связанные с дополнительными проверками безопасности). Оценка риска должна быть первым шагом перед применением MASVS.

Примеры

MASVS-L1

- Все мобильные приложения. В MASVS-L1 перечислены рекомендации по безопасности, которые могут быть выполнены с разумным воздействием на стоимость разработки и пользовательский опыт. Применяйте требования MASVS-L1 для любого приложения, которое не подходит ни под один из более высоких уровней.

MASVS-L2

- Индустрия здравоохранения: мобильные приложения, которые хранят личную информацию, которая может использоваться для кражи личных данных, мошеннических платежей или различных схем мошенничества. Для сектора

здравоохранения США пункты проверок включают в себя закон о переносимости и подотчетности медицинского страхования (HIPAA), привила безопасности, правила уведомления о нарушениях и правило безопасности пациентов.

- Финансовая индустрия: приложения, которые обеспечивают доступ к высокочувствительной информации, такой как номера кредитных карт, личной информации или позволяют переводить средства. Эти приложения требуют дополнительных средств контроля безопасности для предотвращения мошенничества. Финансовым приложениям необходимо обеспечить соблюдение стандартов безопасности данных, изложенных в Payment Card Industry Data Security Standard (PCI DSS), акт Gramm Leech Bliley и акт Sarbanes-Oxley (SOX).

MASVS L1+R

- Мобильные приложения, где защита IP - это основа бизнеса. Проверки отказоустойчивости, перечисленные в MASVS-R, могут использоваться для увеличения усилий, необходимых для получения исходного кода, и для предотвращения фальсификации или взлома.

- Игровая индустрия: игры с существенной необходимостью предотвращения возможности изменять игру и использовать читы, например, конкурирующие онлайн игры. Читинг является важной проблемой в онлайн-играх, так как большое количество мошенников делают игроков недовольными и в конечном итоге может привести к краху конкретного игрового продукты. MASVS-R предоставляет основные средства проверки защиты, направленные на предотвращение несанкционированного доступа, которые увеличивают усилия, которые необходимо приложить читерам.

MASVS L2+R

- Финансовая индустрия: приложения для онлайн банкинга, которые позволяют пользователю переводить средства, где инъекция кода и инструментирование на взломанных(jailbreak/root) устройствах представляют собой риск. В этом случае, верификация MASVS-R может быть использована, чтобы препятствовать фальсификации, повышая сложность для авторов вредоносных программ.

- Все мобильные приложения, в архитектуре которых заложено хранение конфиденциальных данных на мобильном устройстве и в то же время заложена поддержка широкого спектра устройств и версий операционной системы. В этом случае проверка отказоустойчивости может использоваться в качестве меры эшелонированной защиты, чтобы извлечение конфиденциальных данных злоумышленником было затруднено максимально.

Оценка и сертификация

Позиция OWASP в отношении сертификатов MASVS и знаков доверия

OWASP, как некоммерческая организация, не сертифицирует поставщиков, аудиторов или программное обеспечение.

Все пройденные проверки, знаки доверия или сертификаты не были выданы официально, не зарегистрированы и не сертифицированы OWASP, поэтому организация, куда предоставили такую отметку, должна быть осторожна с доверием к любой третьей стороне или удостоверяющему центру, утверждающему пройденную сертификацию ASVS.

Это не должно препятствовать организациям предлагать услуги аудита, если они не претендуют на официальную сертификацию OWASP.

Руководство по сертификации мобильных приложений

Рекомендуемый способ проверки соответствия мобильного приложения MASVS заключается в whitebox тестировании, что означает, что аудиторы получают доступ к ключевым ресурсам, таким как: архитекторы и разработчики приложения, проектная документация, исходный код и доступ к конечным эндпоинтам(web endpoints), включая доступ к, по меньшей мере, одной учетной записи для каждой роли.

Важно отметить, что MASVS охватывает только безопасность мобильных приложений (на стороне клиента) и сетевую связь между приложением и его эндпоинтами, а также несколько базовых и общих требований, связанных с аутентификацией пользователя и управлением сессиями. Данный документ не содержит конкретных требований к удаленным службам (например, веб-службам), связанным с приложением, которые бы удовлетворяли набору базовых требований, обеспечивающих безопасность аутентификации и управления сессиями. Однако MASVS V1 указывает, что веб-сервисы должны быть охвачены общей моделью угроз и проверяться на соответствие определенным стандартам, таким как ASVS OWASP. Проверяющая организация должна включать в любой отчет сферу проверки (особенно, если ключевой компонент выходит за рамки), сводку результатов проверки, включая пройденные и неудачные тесты, с четкими указаниями о том, как разрешить проваленные тесты. Сохранение подробных рабочих документов, скриншотов или видео, сценариев для надежного воспроизведения проблемы, электронных записией тестирования, таких как логов прокси и связанных с ним заметок, таких как списки очистки- считаются стандартной отраслевой практикой. Недостаточно просто запустить инструмент и сообщить о сбоях, это не дает достаточных доказательств того, что всевозможные проблемы были протестированы должным образом проверяющим. В случае возникновения спора должны быть достаточные, подтверждающие доказательства, чтобы продемонстрировать, что каждое верифицируемое требование действительно было проверено.

Использование OWASP Mobile Security Testing Guide (MSTG)

The OWASP MSTG is a manual for testing the security of mobile apps. It describes the technical processes for verifying the requirements listed in the MASVS. The MSTG includes a

list of test cases, each of which map to a requirement in the MASVS. While the MASVS requirements are high-level and generic, the MSTG provides in-depth recommendations and testing procedures on a per-mobile-OS basis.

OWASP MSTG - это руководство по проверке безопасности мобильных приложений. В нем описываются технические процессы для проверки требований, перечисленных в MASVS. MSTG включает список тестовых примеров, каждый из которых соответствует требованию в MASVS. Хотя требования MASVS являются высокоуровневыми и универсальными, MSTG предоставляет подробные рекомендации и процедуры тестирования, для каждой целевой мобильной ОС.

Роль автоматических инструментов тестирования безопасности

Рекомендуется использовать сканеры исходного кода и инструменты blackbox(тестирования без исходного кода) тестирования, чтобы повысить эффективность, когда это возможно. Однако невозможно выполнить проверку MASVS, используя только автоматизированные инструменты: каждое мобильное приложение отличается, и понимание общей архитектуры, бизнес-логики и технических проблем конкретных технологий и фреймворков является обязательным требованием для верификации безопасности приложения.

Другие применения

Как подробное руководство по архитектуре безопасности

Одно из наиболее распространенных применений для MASVS - это ресурс для архитекторов безопасности. В двух основных инфраструктурах архитектуры безопасности, SABSA или TOGAF отсутствует большая информация, необходимая для завершения обзоров архитектуры безопасности мобильных приложений. MASVS можно использовать для заполнения этих пробелов, позволяя архитекторам безопасности выбирать лучшие требования безопасности, чаще встречающиеся в мобильных приложениях.

В качестве замены готовых чеклистов написания безопасного кода

Многие организации могут извлечь выгоду из соответсвия MASVS, выбрав один из двух уровней или путем форка MASVS и изменения того, что требуется, для уровня риска каждого приложения, беря во внимание его область применения. Мы поощряем этот тип модификации до тех пор, пока сохраняется приемственность, таким образом что, по мере развития стандарта, если приложение прошло требование 4.1, это означает то же самое и для измененнной копии требований.

В качестве основы для методологий тестирования безопасности

Хорошая методология тестирования безопасности мобильных приложений должна охватывать все требования, перечисленные в MASVS. В OWASP MSTG описываются примеры whitebox тестирования и blackbox для каждого требования.

Как руководство для автоматического юнит и интеграционного тестирования

MASVS был разработан очень тестируемым, за исключением архитектурных требований. Автоматическое юнит, интеграционное и приемочное тестирование на основе требований MASVS могут быть интегрированы в непрерывный жизненный цикл разработки. Это не только повышает уровень осведомленности о написании безопасного кода разработчиками, но также улучшает общее качество получаемых приложений и уменьшает количество неожиданных "находок" во время релизного тестирования безопасности.

Для курсов по обучению безопасной разработке

MASVS также может использоваться для определения характеристик безопасных мобильных приложений. Многие курсы «безопасного программирования» - это просто этические курсы взлома с лёгким намеком на подсказки по написанию кода, что, безусловно, не помогает разработчикам. При этом курсы безопасной разработки могут использовать MASVS, уделяя особое внимание проактивным средствам контроля, задокументированным в MASVS, а не, например, топ-десяти проблемам небезопасного кода.

V1: Требования к архитектуре, дизайну и моделе угроз

Цель верификации

В идеальном мире безопасность должна приниматься во внимание на всех этапах разработки. Однако на самом деле безопасность часто рассматривается только на поздней стадии SDLC. Помимо технических средств управления, MASVS требует наличия процессов, которые гарантируют, что безопасность была явно учтена при разработке архитектуры мобильного приложения и что функциональные и защитные роли всех компонентов известны. Поскольку большинство мобильных приложений выступают в качестве клиентов у веб-сервисов, необходимо обеспечить применение соответствующих стандартов безопасности: только тестирования изолированного мобильного приложения недостаточно.

В категории «V1» перечислены требования, касающиеся архитектуры и дизайна приложения. Таким образом, это единственная категория, которая не соответствует техническим тестам в OWASP MSTG. Чтобы охватить такие темы, как моделирование угроз, безопасный SDLC, управление ключами, пользователи MASVS должны проконсультироваться с соответствующими проектами OWASP и/или другими стандартами, такими как те, которые приведены ниже.

Требования к верификации безопасности

Ниже приведены требования к MASVS-L1 и MASVS-L2.

#	Description	L1	L2
1.1	Все компоненты приложения известны и необходимы.	✓	✓
1.2	Верификация безопасности никогда не применяется только на стороне клиента, но также и на соответствующих эндпоинтах.	✓	✓
1.3	Была определена высокоуровневая архитектура мобильного приложения и всех связанных эндпоинтов, и в этой архитектуре была рассмотрена безопасность.	✓	✓
1.4	Четко определены данные, считающиеся чувствительными в контексте мобильного приложения.	✓	✓
1.5	Все компоненты приложения определяются с точки зрения бизнес функций и/или функций безопасности, которые они предоставляют.		✓
1.6	Была создана модель угроз для мобильного приложения и связанных с ним удаленных эндпоинтов, которая идентифицирует потенциальные угрозы и контрмеры, применимые к ним.		✓
1.7	Все средства верификации безопасности имеют централизованную реализацию.		✓
1.8	Существует явная политика в отношении того, как происходит управление криптографическими ключами (если они есть), и обеспечивается жизненный цикл криптографических ключей. В идеале следуйте стандарту управления ключами, например, NIST SP 800-57.		✓
1.9	Существует механизм принудительных обновлений мобильного приложения.		✓

1.10	Безопасность рассматривается на всех этапах жизненного цикла разработки программного обеспечения.	✓

Ссылки

Для получения дополнительной информации смотрите также:

- OWASP Mobile Top 10: M10 - Extraneous Functionality: https://www.owasp.org/index.php/Mobile_Top_10_2016-M10-Extraneous_Functionality
- OWASP Security Architecture cheat sheet: https://www.owasp.org/index.php/Application_Security_Architecture_Cheat_Sheet
- OWASP Thread modelling: https://www.owasp.org/index.php/Application_Threat_Modeling
- OWASP Secure SDLC Cheat Sheet: https://www.owasp.org/index.php/Secure_SDLC_Cheat_Sheet
- Microsoft SDL: https://www.microsoft.com/en-us/sdl/
- NIST SP 800-57: http://csrc.nist.gov/publications/nistpubs/800-57/sp800-57-Part1-revised2_Mar08-2007.pdf

V2: Требования к конфиденциальности и хранению данных

Цель проверки

Защита конфиденциальных данных, таких как учетные данные пользователя и конфиденциальная информация, является ключевым аспектом безопасности мобильных устройств. Во-первых, конфиденциальные данные могут непреднамеренно быть раскрыты другим приложениям, работающим на том же устройстве, если механизмы операционной системы, такие как механизм межпроцессного взаимодействия(IPC), используются ненадлежащим образом. Данные также могут непреднамеренно попасть в облачное хранилище, резервную копию или кеш клавиатуры. Кроме того, мобильные устройства могут быть потеряны или украдены легче чем другие типы устройств, поэтому более вероятным сценарием является злоумышленник, получающий физический доступ. В этом случае могут быть реализованы дополнительные меры защиты, чтобы затруднить получение конфиденциальных данных.

Обратите внимание, что, поскольку MASVS ориентирован на приложения, он не охватывает политики безопасности на уровне устройств, такие как те, которые применяются решениями MDM(Mobile Device Managment). Мы поощряем использование таких политик в контексте предприятия для дальнейшего повышения безопасности данных.

Определение чувствительных данных

К чувствительным данным, в контексте MASVS, относятся как учетные записи пользователя, так и любые другие данные, которые считаются чувствительными в конкретном контексте, например:

- Личная информация (PII), которая может быть использована для кражи личных данных: номера социального страхования, номера кредитных карт, номера банковских счетов, информация о здоровье.
- Высокочувствительные данные, которые могут привести к репутационному ущербу и/или финансовым потерям, если они скомпрометированы: информация о договорах, информация, охватываемая соглашениями о неразглашении данных, управленческая информация;
- Любые данные, которые должны быть защищены законом или по причине предъявленных внутренних/внешних требований.

Требования к верификации безопасности

Подавляющее большинство проблем с раскрытием информации можно предотвратить, следуя простым правилам. Большинство требований, перечисленных в этой главе, являются обязательными для всех уровней проверки.

#	Description	L1	L2
2.1	Системные хранилища данных используются надлежащим образом для хранения конфиденциальных данных, таких как учетные данные пользователя или криптографические ключи.	✓	✓

2.2	Чувствительная информация должна храниться либо внутри контейнера приложения либо же в системном хранилище.	✓	✓
2.3	Чувствительная информация не записывается в лог приложения.	✓	✓
2.4	Никакие конфиденциальные данные не передаются третьей стороне, если это не является необходимой частью архитектуры.	✓	✓
2.5	Кэш клавиатуры выключен в полях ввода чувствительной информации.	✓	✓
2.6	Чувствительные данных недоступны для механизмов межпроцессного взаимодействия(IPC).	✓	✓
2.7	Никакие конфиденциальные данные, такие как пароли или контакты не видны через пользовательский интерфейс.	✓	✓
2.8	Никакие конфиденциальные данные не включены в резервные копии, созданные мобильной операционной системой.		✓
2.9	Приложение скрывает конфиденциальные данные с экрана(views), когда находится в фоновом режиме.		✓
2.10	Приложение не хранит конфиденциальные данные в памяти дольше, чем необходимо, и память очищается явно после использования.		✓
2.11	Приложение использует минимальную политику безопасности доступа к устройству, например, требуя от пользователя установить пароль для устройства.		✓
2.12	Приложение информирует пользователя о типах обрабатываемой персональной информации, а также о лучших приктиках безопасности, которым должен следовать пользователь при использовании приложения.		✓

Ссылки

OWASP MSTG содержит подробные инструкции по верификации требований, перечисленных в этом разделе.

- Для Android - https://github.com/OWASP/owasp-mstg/blob/master/Document/0x05d-Testing-Data-Storage.md
- Для iOS - https://github.com/OWASP/owasp-mstg/blob/master/Document/0x06d-Testing-Data-Storage.md

Для получения дополнительной информации смотрите также:

- OWASP Mobile Top 10: M2 - Insecure Data Storage: https://www.owasp.org/index.php/Mobile_Top_10_2016-M2-Insecure_Data_Storage
- CWE: https://cwe.mitre.org/data/definitions/922.html

V3: Требования к шифрованию

Цель верификации

Криптография является важным компонентом защиты данных, хранящихся на мобильном устройстве. Но также это и область, в которой все может пойти не так, особенно когда стандартные соглашения не соблюдаются. Назначение верификационных требований в этой главе состоит в том, чтобы убедиться – проверенное приложение использует криптографию в соответствии с отраслевыми передовыми методами, в том числе такими, как:

- Использование проверенных криптографических библиотек;
- Правильный выбор и настройка криптографических примитивов;
- Подходящий генератор случайных чисел везде, где требуется случайность.

Требования к верификации безопасности

#	Description	L1	L2
3.1	Приложение не полагается на симметричную криптографию с жестко закодированными ключами в качестве единственного метода шифрования.	✓	✓
3.2	Приложение использует проверенные реализации криптографических примитивов.	✓	✓
3.3	Приложение использует криптографические примитивы, которые подходят для конкретного прецедента, с параметрами, которые соответствуют лучшим практикам отрасли.	✓	✓
3.4	Приложение не использует криптографические протоколы или алгоритмы, которые считаются в сообществе устаревшими для целей безопасности.	✓	✓
3.5	Приложение не использует один и тот же криптографический ключ для нескольких целей.	✓	✓
3.6	Все случайные значения генерируются с использованием достаточно безопасного генератора случайных чисел.	✓	✓

References

OWASP MSTG содержит подробные инструкции по верификации требований, перечисленных в этом разделе.

- Android - https://github.com/OWASP/owasp-mstg/blob/master/Document/0x05e-Testing-Cryptography.md
- iOS - https://github.com/OWASP/owasp-mstg/blob/master/Document/0x06e-Testing-Cryptography.md

Для получения дополнительной информации смотрите также:

- OWASP Mobile Top 10: M5 - Недостаточное шифрование
- CWE: https://cwe.mitre.org/data/definitions/310.html

V4: Требования к аутентификации и управлению сессиями

Цель верификации

В большинстве случаев, процесс входа в удаленную службу, являются неотъемлемой частью общей архитектуры мобильного приложения. Несмотря на то, что большая часть логики происходит на эндпоинте, MASVS определяет некоторые основные требования, касающиеся управления учетными записями пользователей и сессиями.

Требования к верификации безопасности

#	Description	L1	L2
4.1	Если приложение предоставляет пользователям доступ к удаленной службе, на удаленном эндпоинте выполняется некоторая форма аутентификации, например аутентификация имени пользователя и пароля.	✓	✓
4.2	Если используется управление сессиям с использованием состояния(Statefull), эндпоинт использует случайно генерируемые идентификаторы сессии для аутентификации клиентских запросов, чтобы не отправлять учетные данные пользователя.	✓	✓
4.3	Если используется аутентификация без состояний (stateless), на основе токена, сервер предоставляет токен, который был подписан с использованием защищенного алгоритма.	✓	✓
4.4	Эндпоинт завершает существующую сессию, когда пользователь выходит из системы.	✓	✓
4.5	Политика паролей существует и применяется на удаленном веб-сервисе.	✓	✓
4.6	Эндпоинт реализует механизм для защиты от предоставления учетных данных чрезмерное количество раз.	✓	✓
4.7	Сессии становятся недействительны на эндпоинте после предопределенного периода бездействия, а также токен стоновится невалиден.		✓
4.8	Биометрическая аутентификация, если она есть, не связана с событиями (т.е. с использованием API, который просто возвращает «истина» или «ложь»). Вместо этого она основана на разблокировке keychain/keystore.		✓
4.9	Второй фактор аутентификации существует на эндпоинте, и требование 2FA последовательно используется.		✓
4.10	Для чувствительных транзакций требуется более высокий уровень аутентификации.		✓
4.11	Приложение информирует пользователя о всех действиях входа в систему, с использованием своей учетной записи. Пользователи могут просматривать список устройств, используемых для доступа к учетной записи, и блокировать определенные устройства.		✓

Ссылки

OWASP MSTG содержит подробные инструкции по верификации требований, перечисленных в этом разделе.

- Для Android - https://github.com/OWASP/owasp-mstg/blob/master/Document/0x05f-Testing-Authentication.md
- Для iOS - https://github.com/OWASP/owasp-mstg/blob/master/Document/0x06f-Testing-Authentication-and-Session-Management.md

Для получения дополнительной информации смотрите также:

- OWASP Mobile Top 10: M4 - Insecure Authentication, M6 - Insecure Authorization
- CWE: https://cwe.mitre.org/data/definitions/287.html

V5: Требования к сетевой передаче данных

Цель верификации

Целью требований, перечисленных в этом разделе, является обеспечение конфиденциальности и целостности информации, передаваемой между мобильным приложением и конечными эндпоинтами удаленной службы. По крайней мере, мобильное приложение должно настроить безопасный, зашифрованный канал для сетевой связи с использованием протокола TLS с соответствующими настройками. L2 содержит дополнительную меру защиты, такую как SSL pinning.

Требования к верификации безопасности

#	Description	L1	L2
5.1	Данные, передаваемыепо сети, шифруются с использованием TLS. Безопасный канал используется последовательно во всем приложении.	✓	✓
5.2	Настройки TLS соответствуют современным рекомендациям или максимально приближены к ним, если мобильная операционная система не поддерживает рекомендуемые стандарты.	✓	✓
5.3	Приложение проверяет сертификат X.509 удаленного эндпоинта, когда установлен защищенный канал. Принимаются только сертификаты, подписанные доверенным центром сертификации(CA).	✓	✓
5.4	Приложение использует свое собственное хранилище сертификатов или "прицепляется" к сертификату эндпоинта или использует открытый ключ и впоследствии не устанавливает соединения с эндпоинтами, которые предлагают другой сертификат или ключ, даже если они подписаны доверенным центром сертификации(CA).		✓
5.5	Приложение не полагается на один небезопасный канал связи (электронная почта или SMS) для критических операций: регистрация и восстановление аккаунта.		✓
5.6	Приложение полагается только на актуальные библиотеки проверки подключения к сети и безопасности.		✓

Ссылки

OWASP MSTG содержит подробные инструкции по верификации требований, перечисленных в этом разделе.

- Android - https://github.com/OWASP/owasp-mstg/blob/master/Document/0x05g-Testing-Network-Communication.md
- iOS - https://github.com/OWASP/owasp-mstg/blob/master/Document/0x06g-Testing-Network-Communication.md

Для получения дополнительной информации смотрите также:

- OWASP Mobile Top 10: M3 - Insecure Communication: https://www.owasp.org/index.php/Mobile_Top_10_2016-M3-Insecure_Communication
- CWE: https://cwe.mitre.org/data/definitions/319.html
- CWE: https://cwe.mitre.org/data/definitions/295.html

V6: Требования к взаимодействию с платформой

Цель верификации

Требования в этой группе гарантируют, что приложение использует API платформы и стандартные компоненты безопасно. Кроме того, содержатся требования относительно коммуникации между приложениями (IPC).

Требования к верификации безопасности

#	Description	L1	L2
6.1	Приложение запрашивает только минимально-необходимый набор разрешений.	✓	✓
6.2	Все вводы от внешних источников и пользователя проверяются и, при необходимости, дезинфицируются. Сюда входят данные, полученные через пользовательский интерфейс, механизмы IPC, такие как намерения(intents), кастомные URL схемы и сетевые источники.	✓	✓
6.3	Приложение не экспортирует чувствительные функции через настраиваемые схемы URL, если эти механизмы не защищены должным образом.	✓	✓
6.4	Приложение не экспортирует чувствительные функции через возможности IPC, если эти механизмы не защищены должным образом.	✓	✓
6.5	JavaScript отключен в WebViews, если явно не требуется.	✓	✓
6.6	WebViews настроены так, чтобы пропускать только минимальный набор обработчиков протоколов (в идеале поддерживается только https). Потенциально опасные обработчики (такие как: file, tel и app-id) отключены.	✓	✓
6.7	Если нативные методы приложения доступны в WebView, необходимо убедиться в том, что WebView выполняет тот JavaScript который содержиться в пакете приложения.	✓	✓
6.8	Десериализация объектов, если таковая имеется, реализована с использованием безопасных API сериализации.	✓	✓

Ссылки

OWASP MSTG содержит подробные инструкции по верификации требований, перечисленных в этом разделе.

- Android - https://github.com/OWASP/owasp-mstg/blob/master/Document/0x05h-Testing-Platform-Interaction.md
- iOS - https://github.com/OWASP/owasp-mstg/blob/master/Document/0x06h-Testing-Platform-Interaction.md

Для получения дополнительной информации смотрите также:

- OWASP Mobile Top 10: M1 - Improper Platform Usage
- CWE: https://cwe.mitre.org/data/definitions/20.html
- CWE: https://cwe.mitre.org/data/definitions/749.html

V7: Требования к качеству кода и настройкам сборки

Цель контроля

Целью этих проверок является обеспечение того, чтобы при разработке приложения соблюдались базовые практики безопасного написания кода, а также были активированы «бесплатные» функции безопасности, предлагаемые компилятором.

Требования к верификации безопасности

#	Description	L1	L2
7.1	Приложение подписано и обеспечено действительным сертификатом.	✓	✓
7.2	Приложение было скомпилировано в режиме release с настройками, подходящими для сборки релиза (например, без отладки).	✓	✓
7.3	Отладочные символы удалены из нативных бинарных файлов.	✓	✓
7.4	Код отладки был удален, и приложение не регистрирует подробные ошибки или отладочные сообщения.	✓	✓
7.5	Все сторонние компоненты, используемые мобильным приложением (библиотеки и фреймворки) известны и проверены на наличие известных уязвимостей.	✓	✓
7.6	Приложение отлавливает и обрабатывает возможные исключения.	✓	✓
7.7	Логика обработки ошибок(в элементах безопасности) запрещает доступ по умолчанию.	✓	✓
7.8	В неуправляемом коде память выделяется, освобождается и используется безопасно.	✓	✓
7.9	Активированы "бесплатные" функции безопасности, предлагаемые инструментальным набором(toolchain), такие как минификация байтового кода, защита стека, поддержка PIE и ARC(автоматический подсчет ссылок).	✓	✓

Ссылки

OWASP MSTG содержит подробные инструкции по проверке требований, перечисленных в этом разделе.

- Android - https://github.com/OWASP/owasp-mstg/blob/master/Document/0x05i-Testing-Code-Quality-and-Build-Settings.md
- iOS - https://github.com/OWASP/owasp-mstg/blob/master/Document/0x06i-Testing-Code-Quality-and-Build-Settings.md

Для получения дополнительной информации смотрите также:

- OWASP Mobile Top 10: M7 - Client Code Quality
- CWE: https://cwe.mitre.org/data/definitions/119.html
- CWE: https://cwe.mitre.org/data/definitions/89.html
- CWE: https://cwe.mitre.org/data/definitions/388.html
- CWE: https://cwe.mitre.org/data/definitions/489.html

V8: Требования устойчивости к внешним воздействиям

Цель проверки

В этом разделе рассматриваются меры по глубокой эшелонированности защиты, рекомендуемые для приложений, которые обрабатывают или предоставляют доступ к конфиденциальным данным или функциям. Отсутствие какого-либо из этих элементов защиты не приводит к образованию уязвимости - напротив, они призваны повысить устойчивость приложения к обратному проектированию и конкретным атакам на стороне клиента.

Требования в этом разделе должны применяться по мере необходимости, которая основывается на оценке рисков, вызванных несанкционированным вмешательством в приложение и/или обратным проектирование кода. Мы предлагаем обратиться к документу OWASP «Technical Risks of Reverse Engineering and Unauthorized Code Modification Reverse Engineering and Code Modification Prevention» (см. ссылки ниже) для списка бизнес рисков, а также связанных с ними технических угроз.

Для того чтобы любое из требований в приведенном ниже списке было эффективным, приложение должно выполнить, по меньшей мере, все MASVS-L1 (т. е. надежные требования по обеспечению безопасности должны быть соблюдены), а также все требования с более низким номером в V8. Например, требования обфускации, перечисленные в разделе «создание препятсвий для анализа», должны сочетаться с «изоляцией приложения», «препятствование динамическому анализу и изменению» и «привязке устройства».

Обратите внимание, что защита программного обеспечения никогда не должна использоваться в качестве замены средств контроля безопасности. Требования, перечисленные в MASVR-R, предназначены для добавления специальных средств защиты, завясящих от угрозы, приложения, которое должно соответствовать требованиям безопасности MASVS.

Следующие предположения необходимо иметь в виду:

i. Должна быть определена модель угроз, в которой они четко прописаны, и от которых происходит защита. Кроме того, должна быть указана степень защиты, которую должна обеспечить схема. Например, заявленная цель может заключаться в том, чтобы заставить авторов вредоносных программ, которые хотят изучить приложение, приложить значительные усилия для осуществления ручного обратного проектирования.

ii. Модель угрозы должна быть предельно ясной. Например, скрытие криптографического ключа в реализации whitebox не имеет смысла, если злоумышленник может просто посмотреть весь код.

iii. Эффективность защиты всегда должна проверяться экспертом, имеющим опыт тестирования конкретных типов защиты от фальсификации и обфускации (см. также главы «обратное проектирование» и «оценка защиты программного обеспечения» в OWASP MSTG.

Создание препятствий для обратного проектирования и фальсификации

#	Description	R
8.1	Приложение обнаруживает и реагирует на наличие root или jailbreak либо путем уведомления пользователя, либо прекращением работы приложения.	✓
8.2	Приложение не позволяет использовать отладчики и/или обнаруживает и реагирует на использование отладчика. Все доступные отладочные протоколы должны быть учтены.	✓
8.3	Приложение обнаруживает и реагирует на подмену исполняемых файлов и критических данных в своей песочнице.	✓
8.4	Приложение обнаруживает и реагирует на наличие широко используемых инструментов и фреймворков обратного проектирования на устройстве.	✓
8.5	Приложение обнаруживает и реагирует на запуск в эмуляторе.	✓
8.6	Приложение обнаруживает и реагирует на фальсификацию кода и данных в своем собственном пространстве памяти.	✓
8.7	Приложение реализует несколько механизмов в каждой категории защиты (с 8.1 по 8.6). Обратите внимание, что на отказоустойчивость влияет количество и разнообразие оригинальности используемых механизмов.	✓
8.8	Механизмы обнаружения инициируют ответы разных типов, включая отложенные и скрытые ответы.	✓
8.9	Обфускация применяется к программной защите, которая, в свою очередь, препятствует деобфускации посредством динамического анализа.	✓

Привязка устройства

#	Description	R
8.10	Приложение реализует функциональность привязки устройства, используя отпечаток устройства, полученный из нескольких свойств, уникальных для устройства.	✓

Создание препятсвий для анализа

#	Description	R
8.11	Все исполняемые файлы и библиотеки, принадлежащие приложению, зашифрованы на уровне файла, и/или важные сегменты кода и данных внутри исполняемых файлов зашифрованы или упакованы. Тривиальный статический анализ не показывает важный код или данные.	✓
8.12	Если целью обфускации является защита конфиденциальных вычислений, используется схема обфускации, которая подходит как для конкретной задачи, так и надежна против ручных и автоматизированных методов деобфускации, учитывая опубликованные в настоящее время исследования. Эффективность схемы обфускации должна быть проверена с помощью ручного тестирования. Обратите внимание, что изоляция аппаратных функций предпочтительнее по сравнению с обфускацией, если это возможно.	✓

Ссылки

OWASP MSTG содержит подробные инструкции по проверке требований, перечисленных в этом разделе.

- Android - https://github.com/OWASP/owasp-mstg/blob/master/Document/0x05j-Testing-Resiliency-Against-Reverse-Engineering.md
- iOS - https://github.com/OWASP/owasp-mstg/blob/master/Document/0x06j-Testing-Resiliency-Against-Reverse-Engineering.md

Для получения дополнительной информации смотрите также:

- OWASP Mobile Top 10: M8 - Code Tampering, M9 - Reverse Engineering
- WASP Reverse Engineering Threats - https://www.owasp.org/index.php/Technical_Risks_of_Reverse_Engineering_and_Unauthorized_Code_Modification
- OWASP Reverse Engineering and Code Modification Prevention - https://www.owasp.org/index.php/OWASP_Reverse_Engineering_and_Code_Modification_Prevention_Project

Приложение A: Список терминов

- **2FA** – Двухфакторная аутентификация (2FA) добавляет второй уровень аутентификации для входа в учетную запись.
- **Address Space Layout Randomization (ASLR)** – Метод, затрудняющий использование ошибок повреждения памяти.
- **Application Security** – Безопасность на уровне приложений сосредоточена на анализе компонентов, которые составляют прикладной уровень Open Systems Interconnection Reference Model (OSI Model), вместо того, чтобы сосредоточиться, например, на операционной системе или подключенных сетях.
- **Application Security Verification** – Техническая проверка приложения на предмет соблюдения требований MASVS OWASP.
- **Application Security Verification Report** – Отчет, который документирует общие результаты и побочный анализ, созданный верификатором для конкретного приложения.
- **Authentication** – Проверка заявленной пользователем приложения личности.
- **Automated Verification** – Использование автоматических инструментов (инструментов динамического анализа, инструментов статического анализа или обоих), использующих сигнатуры уязвимостей для поиска проблем.
- **Black box testing** – Это метод тестирования программного обеспечения, который анализирует функциональность приложения, не заглядывая в его внутренние структуры или принципы работы.
- **Component** – автономный блок кода, с соответствующими дисковыми и сетевыми интерфейсами, которые взаимодействуют с другими компонентами.
- **Cross-Site Scripting** (XSS) – Уязвимость, обычно обнаруживаемая в веб-приложениях, позволяющая вставлять клиентские скрипты в контент, обычно веб-страницы.
- **Cryptographic module** – Аппаратное обеспечение, программное обеспечение и/или прошивка, которая реализует криптографические алгоритмы и/или генерирует криптографические ключи.
- **CWE** - CWE - это общедоступный список часто-встречающихся недостатков безопасности программного обеспечения. Он служит в качестве общего языка, измерительной палочки для инструментов обеспечения безопасности ПО и в качестве основы для выявления слабостей, смягчения последствий и усилий по профилактике.
- **DAST** –Технологии динамического тестирования безопасности приложений (DAST) предназначены для обнаружения условий, указывающих на уязвимость безопасности в приложении, во время его работы.
- **Design Verification** – Техническая оценка архитектуры безопасности приложения.
- **Dynamic Verification** – Использование автоматических инструментов, использующих сигнатуры уязвимостей для поиска проблем при выполнении приложения.
- **Globally Unique Identifier** (GUID) – уникальный ссылочный номер, используемый в качестве идентификатора в программном обеспечении.
- **Hyper Text Transfer Protocol** (HTTP) – Протокол прикладного уровня для распределенных, совместных, гипермедийных информационных систем. Это основа передачи данных для Всемирной паутины.

- **Hardcoded keys** – Криптографические ключи, которые хранятся в самом устройстве.
- **IPC** – Inter Process Communications, в IPC процессы взаимодействуют друг с другом и с ядром, чтобы координировать свою деятельность.
- **Input Validation** – Канонизация и проверка ненадежного ввода пользователя.
- **JAVA Bytecode** - Java-байт-код - это набор команд виртуальной машины Java (JVM). Каждый байт-код состоит из одного или, в некоторых случаях, двух байтов, которые представляют команду (код операции), а также ноль или более байтов для передачи параметров.
- **Malicious Code** – Код, введенный в приложение во время его разработки без ведома владельца приложения, который обходит предполагаемую политику безопасности приложения. Не то же самое, что вредоносное ПО, такое как вирус или червь!
- **Malware** – Исполняемый код, который вводится в приложение во время выполнения без ведома пользователя или администратора приложения.
- **Open Web Application Security Project** (OWASP) – Проект Open Web Application Security (OWASP) является всемирным свободным и открытым сообществом, направленным на повышение безопасности прикладного программного обеспечения. Наша миссия заключается в том, чтобы сделать безопасность приложений «видимой», чтобы люди и организации могли принимать обоснованные решения о рисках безопасности приложений. http://www.owasp.org/
- **Personally Identifiable Information** (PII) - это информация, которая может использоваться сама по себе или с другой информацией для идентификации, контактирования или нахождения одного человека или для идентификации человека в контексте.
- **PIE** – Независимый от положения исполняемый файл (PIE) представляет собой тело машинного кода, которое, будучи помещенным где-то в первичной памяти, выполняется должным образом независимо от его абсолютного адреса.
- **PKI** – PKI - это соглашение, которое связывает открытые ключи с соответствующими идентификаторами объектов. Связывание устанавливается посредством процесса регистрации и выдачи сертификатов в Центре сертификации (CA).
- **SAST** – Статическое тестирование безопасности приложений (SAST) представляет собой набор технологий, предназначенных для анализа исходного кода приложения, байтового кода и двоичных файлов для обнаружения ошибок кодирования и проектирования, которые указывают на уязвимости безопасности. Решения SAST анализируют приложение «наизнанку» в неработающем состоянии.
- **SDLC** – Жизненный цикл разработки программного обеспечения.
- **Security Architecture** – Абстракция конструкции приложения, которая идентифицирует и описывает, где и как используются требования безопасности, а также идентифицирует и описывает местоположение и чувствительность данных пользователя и приложения.
- **Security Configuration** – Конфигурация времени выполнения приложения, влияющая на использование требований безопасности.
- **Security Control** – Функция или компонент, который выполняет проверку безопасности (например, проверку контроля доступа) или при вызове, приводит к эффекту безопасности (например, генерирует запись аудита).
- **SQL Injection (SQLi)** – Метод инъекции кода, используемый для атаки приложений, управляющих данными, в которые подаются вредоносные операторы SQL в точку входа.

- **SSO Authentication** – Single Sign On(SSO) возникает, когда пользователь входит в один клиент и затем автоматически входит и в другие клиенты, независимо от платформы, технологии или домена, который использует пользователь. Например, когда вы входите в Google, вы автоматически входите в систему на YouTube, документы и почтовую службу.
- **Threat Modeling** - Метод, состоящий в разработке все более совершенных архитектур безопасности для идентификации агентов угроз, зон безопасности, средств контроля безопасности и важных технических и бизнес-активов.
- **Transport Layer Security** – Криптографические протоколы, обеспечивающие безопасность связи через Интернет.
- **URI/URL/URL fragments** – Унифицированный идентификатор ресурса - это строка символов, используемых для идентификации имени или веб-ресурса. Единый указатель ресурса часто используется в качестве ссылки на ресурс.
- **User acceptance testing (UAT)**–Традиционно тестовая среда, которая ведет себя как производственная среда, где все тестирование программного обеспечения выполняется до перехода в лайв.
- **Verifier** – Лицо или команда, которая проверяет приложение на соответствие требованиям ASVS OWASP.
- **Whitelist** –Список разрешенных данных или операций, например список символов, которыми разрешено выполнять проверку ввода.
- **X.509 Certificate** – Сертификат X.509 является цифровым сертификатом, который использует широко распространенный международный стандарт инфраструктуры открытого ключа X.509 (PKI) для проверки того, что открытый ключ принадлежит идентификатору пользователя, компьютера или службы, содержащемуся в сертификате.

Приложение Б: Ссылки

Следующие проекты OWASP, скорее всего, будут полезны для пользователей/приемников этого стандарта:

- OWASP Mobile Security Project - https://www.owasp.org/index.php/OWASP_Mobile_Security_Project
- OWASP Mobile Security Testing Guide - https://www.owasp.org/index.php/OWASP_Mobile_Security_Testing_Guide
- OWASP Mobile Top 10 Risks - https://www.owasp.org/index.php/Projects/OWASP_Mobile_Security_Project_-_Top_Ten_Mobile_Risks
- OWASP Reverse Engineering and Code Modification Prevention - https://www.owasp.org/index.php/OWASP_Reverse_Engineering_and_Code_Modification_Prevention_Project

Аналогичным образом, следующие веб-сайты, скорее всего, будут полезны для пользователей/приемников этого стандарта:

- MITRE Common Weakness Enumeration - http://cwe.mitre.org/
- PCI Security Standards Council - https://www.pcisecuritystandards.org

- PCI Data Security Standard (DSS) v3.0 Requirements and Security Assessment Procedures https://www.pcisecuritystandards.org/documents/PCI_DSS_v3.pdf