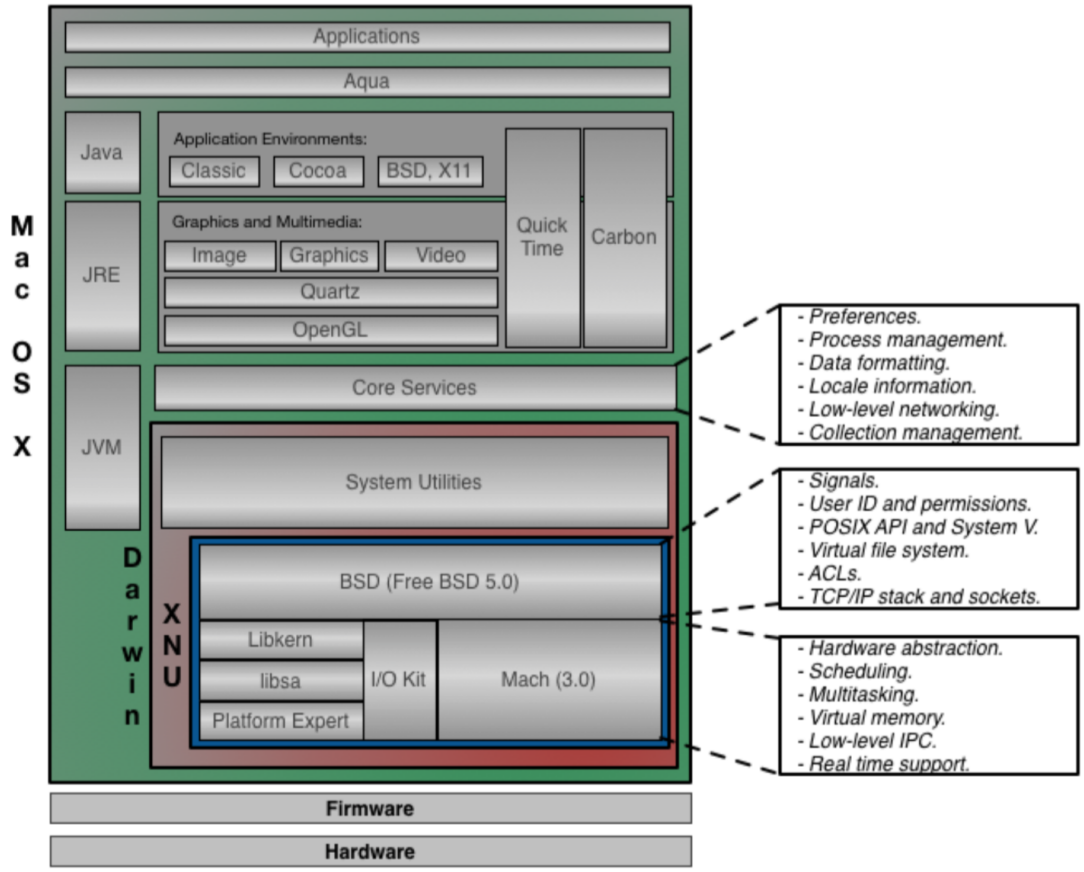
ДАЛЬНЕВОСТОЧНЫЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ

Операционные системы

Реферат
"Безопасность OS X"

Выполнил
Савинов П. А
Группа Б8303б

Содержание



1 Ядро

Ядро Mac OS X - основано на BSD и Mach. BSD предоставляет базовую файловую систему, сетевые сервисы и реализует пользовательское и групповое разделение. BSD устанавливает ограничения доступа на основе пользователя или группы.

Mach занимается управлением памятью, контролем потоков, аппаратной абстракцией и межпроцессовым взаимодействием. Mach контролирует доступ, определяя какие задачи могут отправлять сообщения на порт Mach. (Порт Mach представляет собой задачу или какой-либо иной ресурс). Политики безопасности BSD и контроль доступа Mach составляют важную часть в безопасности Mac OS X.

1.1 ASLR

Address Space Layout Randomization (ASLR) - большое количество эксплойтов в малвари опирается на фиксированные местоположения хорошо известных системных функций. Чтобы подавить эти риски OS X случайным образом перемещает ядро, кексты и системные фреймворки во время загрузки системы.

2 Права доступа

Права доступа являются важной составляющей любой операционной системы.

В OS X права даются на уровне: папок, подпапок, файлов или приложений.

Права доступа так-же могут выдаваться на специфичные данные в файле или функции приложения.

Управление над правами доступа осуществляется на большом количестве уровней, начиная с компонентов Mach/BSD ядра, заканчивая наивысшими уровнями операционной системы. Для сетевых приложений - через сетевые протоколы.

3 Security Framework

Security framework в Mac OS X - реализация архитектуры CDSA. Она содержит расширяемый набор криптографических алгоритмов, для подписания кода и операций шифрования. Она так-же содержит библиотеки, которые позволяют интерпретировать сертификаты типа X.509.

CDSA код используется внутри OS X в таких приложениях, как Keychain и URL Access для защиты данных используемых для входа.

4 Слои безопасности

Безопасность OS X построена на нескольких слоях обороны, для максимальной безопасности.

1. Безопасные подключения | Интернет – файрволл и фильтрация почты
2. Безопасные приложения | Приложения – шифрованные образы дисков и FileVault
3. Безопасные сетевые протоколы | Сеть – файрволл, SSL, Kerberos(Authentication)
4. Сервисы безопасности | Операционная система – keychain, POSIX/ACL(permissions)
5. Secure Boot/"Lock Down" | Аппаратная часть – firmware password utility

5 Keychain

Keychain используется для хранения ключей, сертификатов, паролей и других данных, помещенных в него пользователем. Внутренности шифруются.

Может быть разблокирован, после аутентификации пользователя(пароль, цифровой токен, смарт-карту, или биометрический сканнер). Приложения, так-же могут хранить свои данные в Keychain, соотвественно пользователь будет получать уведомление об этом(реализовано через системное API).

6 Приложения

В OS X существуют два типа приложений: исторические бинарники, как в *nix, и собственный формат .app.

В первом случае, дистрибуция весьма ограничена. Нет централизованного, официального репозитория(или даже пакетного менеджера), но есть подобное от сообщества, MacPorts или Brew. В этом случае применяются всё то-же, что можно сказать про любой *nix.

Второй тип, распространяется изначально, только через App Store, то есть приложение должно быть подписано ключом(private/public key) - code signing, который выдан разработчику Apple, эта функциональность именуется Gatekeeper. В настройках системы возможно отключить такое поведение, тогда

возможно устанавливать в том числе и распространяемые не через App Store приложения.

7 Sandbox

Sandbox включается по решению разработчика, чтобы ограничить доступ извне к его приложению. Например, mDNSResponder использует подобную стратегию, а так-же паттерн pub/sub (по факту он даже не знает о своих подписчиках)

Стандартные приложения вроде Safari, Mail и т.д. работают внутри Sandbox'a.

Sandbox на уровне ядра. Его стратегия следующая:

1. Позволяет описать, как приложение взаимодействует с системой, система даёт приложению права
2. Позволяет пользователю неявно давать приложению дополнительные права, например диалог Открыть/Сохранить, dragdrop и другие похожие пользовательские действия

7.1 Mandatory access controls

Sandbox'ы построены на низкоуровневном механизме контроя доступа в подсистеме ядра - kauth. kauth идентифицирует валидного actor'a(обычно процесс) с помощью его данных. Затем, он опрашивает одного или более слушателей, чтобы определить может ли данный actor выполнить данное действие в заданной области(авторизационном домене). Только первоначальный(дефолтный) слушатель может разрешить запрос; последующие могут только отклонить или отложить. Если все слушатели откладывают, kauth отклоняет запрос.

7.2 Entitlements

Sandbox'ы собирают эти низкоуровневые действия в отдельные entitlement'ы, которые приложение должно явно запросить добавляя соотвествующий ключ в список свойств(файл plist) в свой application bundle(.app).

Могут контролировать доступ к:

1. Вся файловая система
2. Отдельные папки
3. Сеть
4. iCloud
5. Hardware (например, камера или микрофон)
6. Персональная информация(например, контакты)

Дополнительно, они могут контролировать, наследует ли процесс родительские права, а так-же могут предоставлять временные исключения на отправку/получение событий и чтение/запись файлов.

7.3 Пользовательские намерения

Различные действия, вроде DragDrop файла, автоматически отслеживаются системой и система автоматически открывает брешь в Sandbox для этого конкретного файла, так что приложение сможет прочитать его, не запрашивая entitlement'ы.

8 Файловый Карантин

Карантин - мера безопасноти, которая помещает в карантин файлы загруженные из интернета.

Встроенные приложения, вроде Safari, iMessage, Mail будут генерировать предупреждение для пользователя, спрашивая уверен ли он, что хочет открыть файл. Так-же возможность известная, как XProtect проверяет на предмет известной малвари(насколько известно проверяет по хэшу ВСЕГО файла, база заполняется Apple), перед тем, как пользователь откроет файл.

9 System Integrity Protection

Одним из наибольших изменений в OS X 10.11 стало обрезание безграничных прав доступа ко всем частям системы у root пользователя, то есть наследия Unix-основанной системы. Эта возможность называется Rootless.

Защита целостности состоит из нескольких пунктов:

1. Файловая система – системные пути не могут быть перезаписаны, даже root аккаунтом. Системные файлы могут быть изменены только процессами подписанными ключами, принадлежащими Apple. Процессы принадлежащие приложениям, должны записывать данные в места предназначенные для сторонних разработчиков

2. Защита времени исполнения(Runtime) – сторонние приложения не могут прицепляться(attach) к системным процессам. Системные исполняемые файлы могут быть изменены только установщиками или системным обновлением от Apple. Внедрение кода или runtime attachment сторонними приложения более невозможно.

3. Расширения ядра – расширения ядра(kext) должны быть подписаны валидным сертификатом Apple Developer.

10 Summary

Из коробки система имеет весьма неоптимальные настройки безопасности, но здесь хорошо подходит фраза «Безопасность системы в руках её администратора».

В основном, все заявления о бесконечной безопасности это разумеется маркетинг.

Просто существуют объективные сложности при reverse engineering'е системы, а так-же информация об устройстве системы не особо распространена.

Список литературы

1 *Apple*. Mac OS X Security Configuration. — 2010. — URL: http://www.apple.com/support/security/guides/docs/SnowLeopard_Security_Config_v10.6.pdf.

2 *Garijo Joaquin Moreno*. Mac OS X Forensics. — 2015. — URL: https://reverse.put.as/wp-content/uploads/2015/11/RHUL-MA-2015-8.pdf.

3 *Symantec*. Apple Threat Land. — 2016. — URL: http://www.symantec.com/content/en/us/enterprise/media/security_response/whitepapers/apple-threat-landscape.pdf.

4 *Apple*. El Captain Technologies Overview. — 2015. — URL: https://www.apple.com/osx/all-features/pdf/osx_elcapitan_core_technologies_overview.pdf.

5 *Apple*. Security White Paper - отдельное спасибо корпорации Apple, за удаленный документ на английском. — 2010. — URL: http://www.apple.com/jp/training/pdf/wp_osx_security_108_jp.pdf.

6 *Esser Stefan*. OS X El Captain Sinking the ship. — 2016. — URL: https://reverse.put.as/wp-content/uploads/2016/05/syscan360stefanesserosxelcapitansinkingtheship.pdf.

7 *Wardle Patrick*. Methods of Malware Persistence OS X. — 2015. — URL: https://reverse.put.as/wp-content/uploads/2015/11/shakacon_methods_of_malware_persistence.pdf.

8 Code Signing. — 2015. — URL: https://reverse.put.as/wp-content/uploads/2015/12/CodeSigning-RSA.pdf.