Справка

OpenBSD — **свободная** операционная система, основанная на **4.4BSD** — BSD-реализации UNIX-системы; самостоятельный проект, ответвление NetBSD, возникшее в конце 1995 года в результате раскола в команде разработчиков. **Тео де Раадт**, один из четырёх основателей NetBSD, был вынужден покинуть проект после конфронтации по поводу дальнейшего развития ОС.

Взяв за основу дерево исходных кодов NetBSD и переделав его в соответствии со своим видением, он создал свой собственный проект — OpenBSD, в который, вслед за ним, перешли и некоторые другие разработчики NetBSD.

Основным отличием OpenBSD от других свободных операционных систем, базирующихся на 4.4BSD (таких, как NetBSD, FreeBSD), является изначальная ориентированность проекта на создание наиболее безопасной, свободной и лицензионно чистой из существующих операционных систем.

Довольно сложно судить о том, насколько широко распространена OpenBSD, т. к. разработчики ни публикуют, ни собирают подобную статистику.

Цели проекта

- Переносимость (поддерживается 12 аппаратных платформ: Alpha, x86-64, ARMv7, ARMv8, LANDISK, Omron LUNA-88K, MIPS64, PowerPC, SPARC64)
- Стандартизация, корректная работа
- Активная безопасность
- Интегрированные криптографические средства

Сферы применения

- Маршрутизаторы и точки доступа

- Персональные компьютеры: более 8000 пакетов в репозитории:
    - Xenocara – релизация X Window System
    - Рабочие окружения: GNOME, KDE, XFCE
    - Прикладное ПО: веб-браузеры, офисные пакеты и т. п.
- Всевозможные серверы: почтовые, веб, FTP, DNS, NFS

Другие проекты OpenBSD

- LibreSSL - библиотека с реализацией протоколов SSL/TLS
- OpenSSH - набор программ, реализующих протокол SSH
- Пакетный фильтр PF (используется в macOS, NetBSD, FreeBSD)
- Демоны маршрутизации OpenBGPD и OpenOSPFD
- Утилита синхронизации файлов OpenRSYNC
- Демон синхронизации локального системного времени OpenNTPD

Управление памятью

Память процессу выделяется в виде регионов – непрерывных участков памяти. Регион определяется адресом начала и размером в байтах. Информация о них хранится в служебных структурах данных – каталогах регионов. Всего каталогов несколько – в зависимости от ID потока выбирается нужный. За счет этого обеспечивается неблокирующее управление памятью.

Каждый каталог хранит информацию о всех выделенных регионах памяти, о кэшированных регионах (те, которые были освобождены процессами, но не возвращены системе) и прочая статистика.

Кол-во кэшированных регионов ограничено, во-первых, кол-вом слотов для них и, во-вторых, общим объемом кэшированной памяти. В случае если освобождаемый блок памяти слишком большой для кэша, то он сразу же возвращается системе. Если свободных слотов недостаточно, то старые кэшированные регионы возвращаются системе.

При выделении памяти выполняется системный вызов map, который в случае успеха возвращает адрес начала региона. Затем информация о

новом регионе заносится в соответствующий каталог. В зависимости от настроек системы может выполняться зануление региона.

При удалении сначала выполняется по возможности кэширование, затем, то, что не поместилось в кэш каталога возвращается системе.

При перераспределении (изменении размера) возможны три случая:

1. Размер уменьшается. Размер региона уменьшается на соответствующую разницу – обрезается конец. Освобожденная память кэшируется по мере возможности.
2. Размер увеличивается. Сначала пытаемся расширить текущий регион, если область памяти не размечена, по пути удаляя кэшированные регионы в этом участке памяти. Если не удалось, то делаем по «простому»: выделяем новый регион, копируем данные, освобождаем старый.
3. Размер не изменился, либо изменился в пределах размера одной страницы. Ничего не делать.

Планировщик задач

В OpenBSD имеется 128 приоритетов потоков (меньше число – выше приоритет), планировщик располагает 32 очередями, которые представлены двусвязными списками. Переключение контекста выполняется каждые 100 мс.

Среди параметров потока, используемых в планировщике, можно выделить:

- priority - приоритет потока
- nice - параметр, определяющий, насколько данный поток лучше других с таким же приоритетом (0…20)
- estcpu - параметр, определяющий уровень использования потоком CPU
- slptime - счетчик, определяющий, сколько секунд поток провел в состоянии ожидания

Основная идея:

- Чем больше поток потребляет ресурсов CPU, тем ниже становится его приоритет
- Чем дольше поток пребывает в состоянии ожидания, тем меньше значение estcpu и тем выше становится его приоритет
- Через определенный промежуток времени система «забывает» 90% информации о потреблении ресурсов

В планировщике независимо друг от друга с разной периодичностью выполняются следующие процедуры:

- Для потоков в очереди и исполняющихся. Пересчет estcpu и priority каждую секунду по формулам (см. слайд)
- Для исполняющегося потока. Инкремент estcpu и пересчет priority с частотой ~12-16 Гц (на основании анализа исходного кода)
- Смена контекста каждые 100 мс. Берем поток из самой приоритетной очереди и запускаем его. Предыдущий поток кладем в соответствующую очередь

Средняя загрузка системы – среднее кол-во процессов в очереди и на процессорах.

При выходе потока из состояния ожидания выполняется пересчет estcpu и priority, а также добавление в очередь и смена контекста при необходимости.

Специальные средства защиты ОС

Write XOR Execute

Суть мехнизма в том, что страницы памяти процесса не могут быть одновременно доступны на запись и исполнение. Таким образом, код может быть исполнен только после запрещения записи, а запись в страницу памяти возможна только после запрета исполнения. Механизм W^X помогает защитить приложения в пространстве пользователя от типовых

атак, осуществляемых через переполнение буфера, в том числе от переполнений стека и активен в OpenBSD по умолчанию.

Для приложений, использующих «грязный» метод JIT-компиляции, доступна опция отключения метода.

Программное отключение SMT

После заявления Teo де Раадта (июнь 2018) о подозрении на наличие аппаратной уязвимости в реализации технологии SMt, в OpenBSD был разработан патч, позволяющий де-факто отключить SMT на уровне ОС

Через интерфейс «hw.smt» можно разрешать или запрещать запуск двух потоков на одном ядре одновременно

В скором времени было официально заявлено о наличии в процессорах Intel, AMD и ARM64 уязвимостей Spectre, Meltdown и прочие.

Системный вызов unveil()

Системный вызов unveil() предоставляет новый способ изоляции доступа к файловой системе, интегрируемый в код приложений. Суть защиты заключается в том, что первым вызовом unveil() для приложения полностью блокируется доступ ко всей ФС. После чего выборочно открывается возможность доступа для отдельных путей, с которыми может работать приложение.

Поддерживаются флаги ограничения доступа, т.е. можно отдельно открыть доступ на чтение, запись и исполнение, можно запретить создание или удаление файлов. Например, можно открыть доступ на запись к /tmp, на запуск /bin/sh и на чтение /var/spool.

Блокировка осуществляется через интеграцию дополнительных фильтров, работающих на уровне системных вызовов, связанных с файловыми операциями (open(), chmod(), rename() и т.п.).

Механизм защиты RETGUARD

В состав компилятора Clang, используемого для сборки базовой системы OpenBSD, интегрирован механизм защиты RETGUARD, нацеленный на

усложнение выполнения эксплоитов, построенных с использованием заимствования кусков кода и приёмов возвратно-ориентированного программирования. Механизм включен только при сборке для архитектуры AMD64.

Суть метода защиты RETGUARD заключается в искажении адреса возврата обработчиков функций. Перед началом обработчика функции добавляется вызов XOR, комбинирующий адрес возврата с генерируемым для каждой функции случайным значением Cookie, которое затем сохраняется в стек. Перед командой возврата управления из функции (ret) значение Cookie извлекается из стека и при помощи операции XOR повторно применяется к адресу возврата, что восстанавливает его исходное значение и позволяет убедиться в том, что адрес перехода не изменился. Для усложнения атак код проверки дополнительно снабжается добавочным заполнением в виде инструкций int 03 перед каждой инструкцией ret.

При штатном ходе выполнения изначальный адрес перехода остаётся неизменен, но при выполнении эксплоита осуществляется переход на составляющий эксплоит блок заимствованных машинных инструкций (гаджет), точка входа в который как правило не совпадает с началом функции. Так как управление передано не на начало, а в определённую часть тела функции, первый "xor" будет пропущен и "xor" перед выходом исказит переданный эксплоитом адрес возврата. При защите функций ядра RETGUARD оценивается как эффективный для блокирования 50% из всех ROP-гаджетов и 15% уникальных гаджетов, по сравнению с ядром OpenBSD 6.3.

Метод реализован в виде патча к компилятору clang, который на этапе компиляции производит автоматическую подстановку кода проверки адреса возврата во все функции. Для функций системной библиотеки и ядра, написанных на языке ассемблер, требуется применение отдельных патчей. Для активации нового метода защиты в приложениях не требуется отдельных действий, достаточно пересобрать их предлагаемым в базовой системе OpenBSD-Current компилятором Clang, в который уже включены

все необходимые патчи. Для отключения сборки с RETGUARD в Clang добавлена опция "-fno-ret-protector".

File Flags – дополнительный уровень защиты данных

- sappnd. Файл доступен для чтения, его нельзя ни изменять, ни удалять – только дописывать. Флаг может быть установлен только пользователем root.
- uappnd. Аналогичен sappnd, за исключением того, что этот флаг может установить также владелец файла.
- schg. Файл нельзя изменить никоим образов. Устанавливается пользователем root.
- uchg. Аналогичен schg, за исключением того, что этот флаг может установить также владелец файла.
- nodump. Файл с таким флагом должен игнорироваться при создании резервной копии.

Securelevels

Securelevels - механизм ограничения доступных для ОС действий.

- Уровень -1. Дополнительные защитные механизмы отключены
- Уровень 0. Используется только при первой загрузке системы, автоматически переходит на уровень 1
- Уровень 1. Уровень по умолчанию
    - Файлы /dev/mem и /dev/kmem доступны только для чтения
    - Символьные файлы смонтированных устройств доступны только для чтения
    - Флаги файлов sappnd и schg нельзя удалить
    - Нельзя загружать/выгружать модули ядра
- Уровень 2. Максимальный уровень
    - Символьные файлы всех устройств доступны только для чтения
    - Многие настройки сети нельзя изменить
    - Системные часы нельзя откатить назад