Что такое SIEM система?

SIEM (Security information and event management) – это система управления событиями безопасности. Данная технология обеспечивает анализ в реальном времени событий безопасности, исходящих от сетевых устройств и приложений, и позволяет реагировать на них до наступления существенного ущерба. Основными задачами систем данного класса является сбор, обработка и анализ событий безопасности, поступающих в систему из множества источников. На основе получаемых событий ИБ осуществляется оперативная оценка защищенности системы. Во многих таких системах есть механизмы принятия решений и инструменты расследования инцидентов. Также неотъемлемой частью СИЕМ систем является наличие средств для формирования отчетных документов.

Информация в СИЕМ поступает с разных источников - таких как IDS, DLP – системы, маршрутизаторы, межсетевые экраны и так далее. Как упоминалось ранее, бывают внешне безобидные события, полученные с разных источников, в совокупности несут в себе угрозу. Предположим, когда происходит отправка письма с чувствительными данными для компании человеком, имеющим на это право, но на адрес, находящийся вне его обычного круга адресов. DLP может этого не отловить, но СИЕМ, используя накопленную статистику, сгенерирует инцидент.

В чем проблема журналирования?

Любое серьезное приложение или ОС ведет логи. То есть, в том или ином виде в системе ведется журнал, в который фиксируется наиболее важные события. Такими событиями может быть успешный вход в систему или не очень успешный, изменение прав и многое другое.

НО! Если все эти логи хранятся только на локальном узле, то их полезность немного снижается. Представим, что злоумышленник атакует сервер или сетевое устройство. Скорее всего его попытки не сразу увенчаются успехом и в логах все это будет видно. В Виндовс события хранятся в журнале **Event Log**, при этом каждое событие имеет свой номер и описание.

В случае с ОС Linux события хранятся в каталоге **/var/log/.** В нем имеется множество различных *log файлов содержащих события от различных источников в текстовом виде. В зависимости от установленных в системе приложений, в каталоге /var/log могут находиться различные файлы журналов.

Однако, когда хакеру удастся захватить административные права в данной системе, он сможет почистить логи, скрыв следу своего присутствия. В случае с Linux, он может просто удалить файлы журналов, или очистить их.

В случае с Windows при очистке логов генерируется событие 1102. Очистка журналов событий. Однако, злоумышленник может легко избавиться от этого события. Имея необходимые права (будем считать, что взломщик уже получил административные права в системе) можно в свойствах журнала событий легко настроить параметры ротации таким образом, чтобы событие очистки логов было быстро удалено.

В качестве альтернативы локальному хранению логов можно предложить централизованный сбор событий на одном узле. В Linux это делается с помощью службы Syslog, в Windows события могут передаваться с помощью механизма Windows Event Forwarding. В результате мы можем собрать события с нескольких узлов на одной машине. При этом, можно с помощью самописных скриптов анализировать приходящие события на

предмет подозрительных активностей и отправлять уведомления об инцидентах в случае их появления. Удобно, но только если у вас не более десятка таких узлов источников и среднее количество событий с каждого из них также немногим больше десяти. В случае если у вас узлов источников или событий больше, начнется потеря событий, так как сервер получатель начнет захлебываться и будет терять события.

И вот здесь у нас возникает необходимость в решения класса SIEM.

Как устроены SIEM

В классическое реализации сием состоит из сборщика, событий, ядра, в котором осуществляется корреляция и хранение событий и веб консоли управления.

За получение событий с источников отвечает **компонент сбора событий**. У разных вендоров этот элемент может называться по-разному: *агент, коннектор, коллектор*. Суть одна: он должен получить событие от источника. Сбор событий может осуществляться в активном режиме – сборщик событий сам подключается к источнику по различным протоколам (RPC, SMB и т.д.) и собирает у него события. Или же источник сам присылает события посредством протокола Syslog или SNMP.

Компонент сбора событий осуществляет агрегацию, нормализацию, фильтрацию сырых событий, полученных от источника, и затем пересылает эти события в нормализованном виде **ядру** системы. Агрегация представляет собой объединение нескольких одинаковых событий в одно, тем самым позволяя сэкономить пропускную способность канала. Нормализация событий обеспечивает приведение их к общему стандарту и подготовке к дальнейшей обработке.

Подозрительные корреляции

В ядре каждое событие проверяется правилами корреляции. В случае соответствия одному или нескольким правилам создается **инцидент**. Например, несколько попыток неудачного входа в систему это возможная попытка подбора пароля. Проверка события от системы СКУД о том, что пользователь входил в здание, при входе в домен AD это тоже пример правила. Также, отсутствие обновлений антивирусных баз - это тоже инцидент, требующий расследования.

Правила корреляции, как и в случае с нормализацией имеются в поставке, но при необходимости в состав SIEM также входят инструменты для создания собственных правил.

Некоторые популярные российские SIEM-системы:

- MaxPatrol SIEM (Positive Technologies). Поддерживает свыше 350 источников данных и поставляется с 1300 готовыми правилами корреляции. Система автоматически адаптируется к изменениям в инфраструктуре, поддерживает динамическую группировку активов на основе заданных критериев.
- KUMA (Kaspersky Unified Monitoring and Analysis) от «Лаборатории Касперского». Модульная SIEM-система, ориентированная на защиту от сложных атак. Интегрируется с другими продуктами Касперского, такими как Kaspersky EDR и Anti Targeted Attack Platform.
- RuSIEM - российская SIEM-система для управления логами, в платной версии - с расширенными функциями корреляции и управления инцидентами. Есть простой

графический конструктор правил корреляции без необходимости писать код, интеграция с внешними системами через API.