DOI: 10.58168/MoInSyTe2024_226-232
УДК 004.9

СОЗДАНИЕ СИСТЕМЫ МОНИТОРИНГА БЕЗОПАСНОСТИ В ОПЕРАЦИОННОЙ СИСТЕМЕ LINUX

М.А. Шацких[1], В.И. Анциферова[1], Н.В. Данилов [1]

[1]ФГБОУ ВО «Воронежский государственный лесотехнический университет имени Г.Ф. Морозова»

Аннотация. В работе рассматриваются способы мониторинга в операционной системе Linux с точки зрения безопасности. Разобраны основные пути воздействия на систему и предложены способы защиты и отслеживания состояния системы при помощи утилит с открытым исходным кодом.

Ключевые слова: кибербезопасность, Linux, защита данных, вирусы, мониторинг безопасности.

CREATING SECURITY MONITORING SYSTEM IN LINUX OPERATING SYSTEM

M.A. Shatskikh[1], V.I. Antsiferova[1], N.V. Danilov [1]

[1]Voronezh State University of Forestry and Technologies named after G.F. Morozov

Abstract. This paper are discussed about ways for monitoring Linux operating system from a cybersecurity point of view. Researched commonly ways impact on system and suggested methods by protect and monitoring system state with open source utils.

Keywords: cybersecurity, Linux, data protecting, malware, security monitoring.

С каждым годом защита информации и тема кибербезопасности становиться все актуальнее. Своевременное выявление угроз и реагирование на них помогает сохранить данные и работоспособность системы. Важной частью этих мероприятий является мониторинг.

Мониторинг системы безопасности может проводиться в вручную, но сбор, анализ и настройку системы безопасности, достаточно трудоемкая задача.

При этом могут возникать недосмотры и ошибки, особенно если приходится работать не с одной системой, а обслуживать группу устройств. На помощь приходят утилиты, проводящие анализ системы в автоматическом режиме с использованием скриптов.

Философия Unix заключается в том, что одна программа делает только одну задачу, но делает ее хорошо. Для Linux существует достаточно много программ предназначенных для сбора информации о безопасности системы. Но каждая из них по отдельности не собирает абсолютно всю необходимую информацию, поэтому их приходится комбинировать, что несет дополнительные сложности. Во первых, в полученных отчетах некоторые данные могут пересекаться, во вторых, собирать и анализировать большой отчет трудоемко.

Большая часть программ безопасности анализирует угрозы уже после инцидента, после того как система была взломана. Это кажется нелогично, но такие программы проще реализовать, т. к. найти следы взлома сильно проще, чем пытаться каждую секунду угадать, какие действия являются вредоносными, они требуют меньше вычислительных ресурсов и дают меньше ложноположительных результатов.

Небольшая часть программ проводит анализ в реальном времени. Например, анализаторы сетевого трафика, типа Suricata, Snort и другие, или антивирусы, как Kaspersky.

Далее речь пойдет о том, как можно выявить попытку воздействия на систему, какие утилиты помогают защитить систему и проводить автоматизированный мониторинг.

Появление новой учетной записи или группы, изменение прав и групп доступа для пользователя, может свидетельствовать о воздействии на систему. Отслеживание кем и когда был совершен вход, что владелец учетной записи делал в системе, какие запускал программы и приложения, а так же отслеживание фактов удаления и редактирования журналов.

В Linux аутентификация пользователей реализована через встроенный модуль аутентификации (Pluggable Authentication Modules). Для взаимодействия с ним используется PAM API, поэтому для взаимодействия достаточно изменить конфигурационный файл не пересобирая модуль.

Kerberos усложненная и обширная система аутентификации, разработанная в Массачусетском университете в 1980 году. Kerberos — сетевой протокол аутентификации, позволяющий передавать данные через незащищённые сети для безопасной идентификации. Он содержит централизованную базу данных об

одном или более хостов и выступает как центр распределенных ключей (Key Distribution Center). Участники, действующие в системе Kerberos (пользователи, хост или программа, работающая от имени пользователя) отправляют запрос на аутентификацию и получают «билет» (Ticket Granting Ticket — билет для получения билета) от KDC для отдельной службы, такой как удаленный вход в систему, печать и т. д.

Для защиты сети, анализа сетевого трафика используются межсетевые карты, системы обнаружения проникновения (Intrusion Detection System) и система предотвращения вторжения (Intrusion Prevention System), работающая в реальном времени. Постоянная отправка запросов на определенный порт с него, может говорить о вирусной активности, как и наличие сетевых интерфейсов в неразборчивом режиме (Promiscuous mode), что позволяет прослушивать все проходящие пакеты независимо от адресата.

Анализ запускаемых программ, время запуска и использования, выявление скрытых процессов. Rootkit это программа или скрипт, помогающая расширить привилегии в системе до root, что дает возможность делать в системе, что угодно, в т.ч. настроить удаленный доступ, который будет работать, даже если уязвимость, через которую был произведен первоначальный вход, устранили.

Наличие подобной программы в системе тщательно скрывается. Более того, даже если узнать о его наличии и удалив, нельзя быть полностью уверенным, что угрозы больше нет.

Например, Rootkit Diamorphine, один из представителей Linux Kernel

Module руткитов, он может скрывать себя и другие процессы от утилит типа lsmod, ps или top. Но при этом их можно обнаружить в директории /proc. Выявление таких расхождений и поиском скрытых модулей и процессов

может заниматься программа chkrootkit.

В случае нахождения руткита, что бы быть уверенным, что система безопасна, можно сделать чистую установку с удалением всех предыдущих данных на диске, либо вернуться к точке восстановления, где этой проблемы не было и обновить систему, чтобы не заразиться вновь.

Сигнатурный поиск - это когда в файлах ищется определенная последовательность байт схожая на такую, во вредоносном ПО. Такой тип поиска могут проводить, например, утилиты: chkrootkit, rkhunter, lynis, антивирусы Linux Malware Detect и другие.

Любая программа для работы обращается к ядру системы — это называется системный вызов. Если поведение программы, т. е. последовательность таких системных вызовов, совпадает с поведение вредоносной программы, то такой процесс убивается, а программа его создавшая, помещается в карантин. Так работает анализ вирусов на основе моделей поведения. Антивирусы запускают файл сначала в контейнере, и только если никакой подозрительной активности обнаружено не было, разрешает открыть этот файл основной ОС.

Запуск программ и скриптов из директории пользователя. Злоумышленнику, попавшему в систему как обычный пользователь необходимо как-то ее исследовать, повысить права, украсть данные для входа или сделать, что-то еще. Для выполнения этих действий используют эксплойты т. е. программу, скрипт, их набор или все сразу, что бы воздействовать на уязвимость и получить результат.

По возможности нужно запретить запуск любых приложений и скриптов из домашней директории пользователя. Таким образом, можно защититься от эксплойтов, запускаемых от имени пользователя.

Большой объем разделяемой памяти для процесса может выдавать вредоносное ПО. IPC (Inter Process Communication — Межпроцессное взаимодействие) используется, когда одной программе необходимо, в процессе работы, сообщить какую-то информацию другой программе или получить, для этого в Linux системах используется система DBUS (Desktop Bus).

Например, некий вирус или руткит, скрыто работает в системе, собирает информацию о ней информацию, о пользователях, приложениях и прочем. Такую программу может выдать большой объем разделяемой памяти. Однако многие графические приложения, медиа плееры, веб-сервера и др. являясь

вполне легитимными тоже прожорливы и могут давать ложноположительный результат.

В Linux для запуска стандартных приложений существует спецификация XDG. Например, введя в терминале "ls" запускается программа по пути /usr/bin/ls. Узнать эти пути можно используя команду "whereis". В случае подмены этих путей модификации или изменения программы лежащей по этим путям пользователь может запустить вредоносное ПО или экплойт.

Поэтому важно, что бы пути и приложения не были изменены. Отследить это можно с помощью утилит Tripwire и rkhunter. Они реализуют такую технологию как (file integrity checker). Создается, что-то вроде точки восстановления

для системных файлов и директорий и периодически сравниваются файлы из этой базы данных с теми, что используются в данный момент в системе.

Каждый день в ПО находят уязвимости, публикуются отчеты и выходят, заплатки безопасности. Своевременная установка обновлений и патчей, помогает защититься от большинства угроз. Предупреждение о наличии устаревшего или уязвимого ПО и обновлений к нему, важная часть системы мониторинга.

Частое создание резервных копий поможет защититься от вирусов шифровальщиков и вымогателей. В случае выявления заражения системы, можно вернуться к одной из предыдущих версий системы, где все было в порядке.

Программные и общесистемные ошибки и предупреждения, иногда могут говорить о подозрительной активности. Но в любом случае, по возможности лучше их не игнорировать.

Сбор информации о подключении и отключении съемных носителей информации, т. к. это нередко является источником заражения. Если политика безопасности компании позволяет лучше запретить использование личных и сторонних USB устройств на рабочем месте.

Некоторые вирусы любят добавлять себя в автозагрузку. Cron планировщик задач в Linux, тоже лучше проверять, на наличие подозрительных скриптов.

Подмена ядра системы сложно реализуемый, но возможный сценарий. Для защиты достаточно самостоятельно подписать ядро своим ключом и настроит UEFI secure boot, где будет находиться этот ключ. Таким образом, это не позволит запускать модифицированные или сторонние ядра.

Существует множество способов воздействия на систему и для самых основных разработаны методы и утилиты для защиты от них. В последнюю пару лет набирает популярность использования искусственного интеллекта и нейросетевых моделей во всех сферах и для многих целей.

Популярный проект GPT4ALL, позволяет запускать и использовать ИИ модели локально, без интернет подключения, в т.ч. и свои собственные модели. В дальнейшем можно исследовать возможности этой программы для использования в мониторинге системы безопасности. Например, у программы есть возможность загрузить текст, который обработает модель и далее по этому тексту можно задавать вопросы. Так же для общения с моделью, возможно, использовать терминал и получать вывод в него же. Что позволяет писать сложные сценарии с использованием технологии GPT.

Список литературы

1. Защита микропроцессоров от одиночных сбоев / В.А. Смерек, В.М. Антимиров, А.Ю. Кулай, А.Л. Савченко // Моделирование систем и процессов. – 2018 – Т. 11, № 2 – С.71-77.

2. Модель индивидуально группового назначения доступа к иерархически организованным объектам критических информационных систем с использованием мобильных технологий / Е.А. Рогозин, В.А. Хвостов, В.В. Суханов [и др.]// Моделирование систем и процессов. – 2021. – Т. 14, № 1. – С. 73-79. – DOI: 10.12737/2219-0767-2021-14-1-73-79.

3. Проектирование интерфейсов сбоеустойчивых микросхем / В.К. Зольников, Н.В. Мозговой, С.В. Гречаный [и др.] // Моделирование систем и процессов. – 2020. – Т. 13, № 1. – С. 17-24.

4. Суханов, В.В. Методика аналитического мониторинга аномального поведения пользователей в распределенной информационной системе критического применения / В.В. Суханов, О.В. Ланкин // Моделирование систем и процессов. – 2021 – Т. 14, № 1 – С. 79-85. – DOI: 10.12737/2219- 767-2021-14-1-79-85

5. Daniel J. Barrett. Linux Security Cookbook / Daniel J. Barrett, Robert G. Byrnes, Richard Silverman // O'Reilly. - 2003. - 499.

6. Maurice J. Bach. The design of the unix operating system // Prentice-Hall, Inc. - 1986. - 486. - 370-405.

7. Полуэктов А.В., Макаренко Ф.В., Ягодкин А.С. Использование сторонних библиотек при написании программ для обработки статистических данных // Моделирование систем и процессов. – 2022. – Т. 15, № 2. – С. 33-41.

References

1. Protection of microprocessors from single failures / V.A. Smerek, V.M. Antimirov, A.Y. Kulai, A.L. Savchenko // Modeling of systems and processes. - 2018 – Vol. 11, No. 2 – pp.71-77.

2. The model of individually group assignment of access to hierarchically organized objects of critical information systems using mobile technologies / E.A. Rogozin, V.A. Khvostov, V.V. Sukhanov [et al.]// Modeling of systems and processes. - 2021. – Vol. 14, No. 1. – pp. 73-79. – DOI: 10.12737/2219-0767-2021-14-1-73-79.

3. Designing interfaces of fault-tolerant microcircuits / V.K. Zolnikov, N.V. Mozgovoy, S.V. Grechany [et al.] // Modeling of systems and processes. – 2020. – Vol. 13, No. 1. – pp. 17-24.

4. Sukhanov, V.V. Methodology of analytical monitoring of abnormal user behavior in a distributed information system of critical application / V.V. Sukhanov, O.V. Lankin // Modeling of systems and processes. – 2021 – Vol. 14, No. 1 – pp. 79-85. – DOI: 10.12737/2219- 767-2021-14-1-79- 85

5. Daniel J. Barrett. Linux Security Cookbook / Daniel J. Barrett, Robert G. Byrnes, Richard Silverman // O'Reilly. - 2003. - 499.

6. Maurice J. Bach. The design of the Unix operating system // Prentice-Hall, Inc. - 1986. - 486. - 370-405.

7. Poluektov A.V., Makarenko F.V., Yagodkin A.S. The use of third-party libraries when writing programs for processing statistical data // Modeling of systems and processes. - 2022. – Vol. 15, No. 2. – pp. 33-41.