ORES
Научно-практический электронный журнал
Оригинальные исследования (ОРИС), *ISSN 2222-0402*
www.ores.su
original-research.ru

УДК 004.45

SYSTEMD В ДИСТРИБУТИВАХ LINUX

Гарманов Сергей Семенович
Кандидат военных наук, полковник запаса, доцент кафедры Воздушно-космических сил,
Военный учебный центр при Российском технологическом университете МИРЭА
119454, г. Москва, Проспект Вернадского, д. 78
Email: garmanov@mirea.ru

Решетников Даниил Дмитриевич
Студент 5 курса
Институт кибербезопасности и цифровых технологий
Российский технологический университет МИРЭА
119454, г. Москва, Проспект Вернадского, д. 78
Email: r.daniil1@outlook.com

Аннотация

В данной статье рассматривается система инициализации systemd, ее применение и сравнение с другими системами инициализации. Проведен анализ systemd с точки зрения безопасности.

Ключевые слова: информационная безопасность, операционные системы, система инициализации, программное обеспечение.

SYSTEMD IN LINUX DISTRIBUTIONS

Sergey S. Garmanov
Candidate of Military Sciences, Reserve Colonel, Associate Professor of the Department of Aerospace Forces
Military Training Center at the Russian Technological University MIREA
78 Prospekt Vernadskogo, Moscow, 119454
Email: garmanov@mirea.ru

Daniil D. Reshetnikov
Student 5 term
Institute of Cybersecurity and Digital Technologies
Russian Technological University MIREA
78 Prospekt Vernadskogo, Moscow, 119454
Email: r.daniil1@outlook.com

ABSTRACT

This article discusses the systemd init system, its application and comparison with other init systems. Systemd has been analyzed from a security point of view.

Keywords: information security, operating systems, initialization system, software.

При запуске операционной системы (далее - ОС) Linux, начинают работать программы, необходимые для ее использования. Такие программы, работающие в фоне, называют демонами. Для того, чтобы ОС корректно функционировала эти программы должны быть запущены. С этой целью используется система инициализации. В настоящее время их существует большое количество, но самой первой была SysVinit, после нее в Ubuntu использовали Upstart, а затем systemd [1]. Сейчас, systemd используется во многих дистрибутивах, таких как Debian, Ubuntu, Arch Linux.

Основная цель systemd заключается в ускорении загрузки ОС при использовании распараллеливания и отложенного запуска. Распараллеливание позволяет запустить одновременно не связанные службы, если у обоих есть достаточно ресурсов. Отложенный запуск достигается при подготовке всего необходимого (например, создание всех необходимых сокетов) к запуску службы, которая запускается позднее, по запросу [2].

Задачи, выполняемые systemd, описываются в файлах под названием «юниты» (unit). Существуют различные типы юнитов:

- service – описывает то, что можно запустить (служба, скрипт);
- target – осуществляет группировку юнитов (объединение для одновременного запуска);
- timer – определяет таймер, по которому службы могут запускаться по заданному расписанию или с определенной задержкой.

Юнит-файлы состоят из разделов, обозначаемых парой квадратных скобок «[» и «]» с их названиями, заключенными внутри. Каждый такой раздел продолжается до конца файла или до начала другой [3] Поведение юнитов в каждом разделе описывается при помощи формата ключ-значение, разделёнными знаком равно.

Имена разделов имеют четкое определение, а также зависят от регистра, поэтому, имя раздела [Unit] не будет правильным, если оно написано как [UNIT]. При необходимости можно добавить префикс «X-» к имени раздела, если такие имена не являются стандартными.

Первый раздел в большинстве юнит-файлов — это раздел [Unit]. Как правило, он используется для определения метаданных юнитов и настройки взаимосвязей между ними. Его часто ставят первым, поскольку он предоставляет обзор юнита, несмотря на то что при анализе файла их порядок не имеет значения.

В разделе [Service] описывается конфигурация, применимая только к службам. Одна из основных вещей, которая должна быть указана в [Service], является «Type=» сервиса. Эта запись классифицирует службы по их процессу и демонизирующему (запуску в фоновом режиме) поведению. Благодаря этому такая запись сообщает системе инициализации, как управлять сервисом и обнаруживать его состояние.

В конце файла расположен раздел [Install]. Этот раздел не является обязательным и используется для определения поведения юнита, когда он включен или выключен. При его включении он будет автоматически запускаться при загрузке.

Разделение функций на отдельные блоки позволяет внутренним процессам systemd не только оптимизировать параллельную инициализацию, но и сделать конфигурацию

простой, а также позволить изменять и перезапускать некоторые блоки без разрыва и перестройки связанных с ними соединений. Использование таких возможностей позволяет достигать больше гибкости при администрировании.

По сравнению с другими системами инициализации systemd использует простой декларативный синтаксис, который показывает назначение и эффекты различных юнитов при активации. Другие системы инициализации используют язык сценариев для интерпретации файлов инициализации, необходимых для загрузки служб.

Логические значения в конфигурационных файлах могут быть представлены в нескольких вариантах [4]. Например, для положительного допустимо 1, «yes», «on», «true», а отрицательного 0, «no», «off», «false». При использовании интервалов времени можно добавить единицу измерения «s», «min», «h», «d», «ms» (секунды, минуты, часы, дни, миллисекунды) и т. д. По умолчанию число без единиц измерения представлено в секундах, если же их несколько, то они суммируются.

Также можно применять юнит-файлы шаблонов. Их основная задача - создание нескольких экземпляров одного и того же юнита. Такие файлы шаблонов ничем не отличаются от обычных юнит-файлов, однако они обеспечивают гибкость в настройке, позволяя определенным частям файла использовать динамическую информацию, которая будет доступна во время выполнения.

Система инициализации systemd также включает в себя функции для поддержки работы служб. Например, при появлении ошибки система инициализации может перезапустить сервис или назначить юнит для выполнения.

Однако systemd с момента его появления имел большое количество уязвимостей [5]. Например, для получения привилегий root необходимо создать имя пользователя с ошибкой в файле systemd.unit [6]. Несмотря на то, что утилиты systemd не позволяют создать юнит-файлы с таким именем пользователя, другие инструменты могут это сделать.

Ошибку, описанную ранее, система инициализации проигнорирует и создаст запрошенную службу. При обнаружении неизвестной опции в файле, system оставит предупреждение в журнале, после чего продолжит загрузку устройства с правами суперпользователя, вместо того чтобы запретить доступ к ОС.

Злоумышленники также использовали функциональность systemd для установления постоянного доступа к системе жертвы путем создания и изменения служебных юнит-файлов, которые заставляют систему инициализации выполнять вредоносные команды при загрузке ОС [7]. Создание или изменение service юнит-файлов в каталогах /etc/systemd/system и /usr/lib/systemd/system позволяет запускать вредоносный код на уровне системы. В свою очередь аналогичные действия ~/.config/systemd/user/ проводят с целью закрепления на уровне пользователя.

В настоящее время, systemd используется по умолчанию во многих дистрибутивах Linux. Она может использовать сценарии инициализации SysV, а также является продолжением SysVinit, которая использовалась ранее во многих популярных дистрибутивах Linux.

При использовании SysVinit, ОС находится в одном из заранее определённых состояний, называемых уровнями выполнения [8]. Каждый из них организует определенный набор сценариев инициализации, которые выполняются при запуске или завершении работы ОС. Уровни выполнения соответствуют каталогу /etc/, который, в свою очередь, имеет символические ссылки на скрипты в /etc/init.d/. Система обычно загружается в многопользовательском режиме (уровень запуска 2). Другие уровни выполнения представляют однопользовательский режим (используется для восстановления системы), выключение системы и различные другие состояния. Переключение с одного уровня выполнения на другой вызывает запуск набора сценариев для каждого уровня

запуска, которые обычно монтируют файловые системы, запускают или останавливают служебные программы, запускают или останавливают оконную систему, выключают машину и т. д.

SysVinit запускает все службы в заранее определенной последовательности. Сценарий будет выполнен в нужном порядке только в том случае, если текущий скрипт выполняется или истекло время ожидания. При зависании во время выполнения, сценарию приходится ждать, пока не истечет его время ожидания. Такое ожидание повлияло на эффективность процесса инициализации.

В качестве замены SysVinit был создан Upstart. В отличие от SysVinit, созданной для работы в статической среде, Upstart применялся в гибкой среде [9].

Upstart обеспечивал три основных преимущества по сравнению с SysVinit:

- управление службами на основе событий;
- асинхронный запуск служб;
- автоматический перезапуск поврежденных служб.

Upstart вместо использования уровней выполнения использовал системные события для запуска и остановки служб. Событие — это изменение состояния системы. Когда происходило событие, Upstart обнаруживал его и вносил необходимые изменения.

Upstart был создан для устранения недочетов SysVinit, а не в качестве современной системы инициализации с новыми возможностями. Пока не была разработана другая система инициализации, Upstart использовался многими разработчиками Linux в качестве временного решения.

Таким образом, можно сделать вывод, что несмотря на удобства, предоставляемые systemd, (гибкая настройка, использование простого декларативного синтаксиса, запуск нескольких служб) система инициализации имеет и существенные недостатки, связанные с безопасностью. За время существования systemd в ней было обнаружено большое количество уязвимостей, позволяющих повысить привилегии до суперпользователя или управлять системой жертвы при помощи вредоносного кода. По этой причине некоторые пользователи отдают предпочтение дистрибутивам, не содержащим данную систему инициализации. Примером может служить ОС Devuan, которая не содержит systemd и базируется на Debian [10].

Список литературы:

1. System and Service Manager – URL: https://systemd.io/ (дата обращения 8.10.2023)
2. Система инициализации systemd – URL: https://sysadminium.ru/adm-serv-linux-systemd-init-system/ (дата обращения 9.10.2023)
3. Understanding Systemd Units and Unit Files – URL: https://www.digitalocean.com/community/tutorials/understanding-systemd-units-and-unit-files (дата обращения 11.10.2023)
4. Systemd.syntax – URL: https://man.archlinux.org/man/systemd.syntax.7 (дата обращения 13.10.2023)
5. Systemd Project. Systemd. Security Vulnerabilities – URL: https://www.cvedetails.com/vulnerability-list/vendor_id-15978/product_id-34874/Systemd-Project-Systemd.html (дата обращения 10.10.2023)
6. CVE-2017-1000082 Detail – URL: https://nvd.nist.gov/vuln/detail/CVE-2017-1000082 (дата обращения 13.10.2023)

7. Create or Modify System Process: Systemd Service – URL: https://attack.mitre.org/technique s/T1543/002/ (дата обращения 13.10.2023)
8. SysV init: Runlevels – URL: https://learn.adafruit.com/running-programs-automatically-on-your-tiny-computer/sysv-init-runlevels (дата обращения 17.10.2023)
9. Differences between SysVinit, Upstart and Systemd – URL: https://www.computernetworkingnotes.com/linux-tutorials/differences-between-sysvinit-upstart-and-systemd.html (дата обращения 20.10.2023)
10. Welcome to Devuan – URL: https://www.devuan.org/ (дата обращения 20.10.2023)