A. M. Kanner

zLinux: Access Control inside Mainframes

Keywords: zLinux, IBM System z, z/VM, access control

The author describes singularities arising during access control in IBM System z and zLinux environment. The article provides appropriate requirements to protect zLinux and options that meet such requirements and are based on existing experience.

А. М. Каннер

zLINUX: РАЗГРАНИЧЕНИЕ ДОСТУПА В МЕЙНФРЕЙМАХ

Термин *zLinux* (*z/Linux*) или *Linux on System z* — это обобщенное название операционных систем GNU/Linux, функционирующих на современной линейке мейнфреймов IBM System z. Платформа System z предполагает использование bare-metal виртуализации на базе специального гипервизора (Processor Resource and System Manager — PR/SM), разделяя при этом «физические» ресурсы мейнфрейма между логическими разделами (Logical Partitions — LPARs) и организуя таким образом первый слой виртуализации.

В рамках LPAR могут быть загружены экземпляры гостевых ОС (виртуальных машин, ВМ), в качестве которых могут выступать z/OS и zLinux. Кроме того, на отдельном LPAR можно организовать второй (а затем третий и т. д.), вложенный слой виртуализации с помощью такого средства, как z/VM. Во вложенных слоях можно загружать столько ВМ, сколько доступно ресурсов на данном слое виртуализации (или сколько доступно ресурсов, выделенных LPAR, если говорить о втором слое виртуализации). С помощью такой «многослойной» виртуализации System z позволяет на одном физическом мейнфрейме запускать одновременно сотни и даже тысячи полностью изолированных друг от друга гостевых ОС, в зависимости от объема доступных ресурсов.

Итак, в качестве гостевых ОС на System z могут функционировать:

1. z/OS — 64-битная серверная ОС производства IBM (дальнейшее развитие OS/390);

2. z/VM — 64-битная ОС-гипервизор как N + 1 уровень вложенности (подходит для тестовых систем и разработки);

3. zLinux — 31-битная (архитектура s390, на данный момент не используется) или 64-битная (архитектура s390x, применяется на данный момент) ОС Linux, в качестве поддерживаемых дистрибутивов выступают:

- RedHat Enterprise Linux Server, SUSE Linux Enterprise Server (официально заявленная поддержка);
- Debian, Fedora, Slackware, Gentoo (поддерживаются неофициально).

Все гостевые ОС могут оперировать как с реальными, так и с мнимыми (виртуальными) ресурсами, причем суммарно количество виртуальных ресурсов может превосходить тот объем реальных объектов, на которые эти мнимые ресурсы отображаются. Если с оперативной памятью более-менее понятно: любая задача в современной ОС с виртуализацией памяти может занимать больший объем, чем реальная память, то в отношении, например, процессоров все несколько сложнее: в LPAR с z/VM, для которого, к примеру, выделено 32 процессора (это число обеспечивает максимальную производительность самого гипервизора z/VM), могут быть сотни машин (виртуальных, со своими гостевыми ОС) с одним, двумя и более виртуальными процессорами (теоретическое ограничение установлено в 64, но практика показывает, что

оптимальным является количество, равное числу логических процессоров в самой z/VM минус один, или меньше.

Сам гипервизор z/VM обладает мощными инструментами сбора и наблюдения за нагрузкой системы своего собственного LPAR, а также за работоспособностью всего аппаратного комплекса (всех соседних LPAR) в целом. Это позволяет собирать данные по эффективности использования ресурсов своими гостевыми ОС, не вмешиваясь в их работу, как бы со стороны.

Весь архитектурно-зависимый код для поддержки архитектуры s390x внесен в основную ветку ядра Linux, в связи с чем на System z можно с той или иной степенью сложности портировать любой современный дистрибутив Linux.

В данной статье речь пойдет о разграничении доступа в zLinux с точки зрения применения уже существующих идей по разграничению доступа в Linux [1–6].

Необходимые механизмы защиты zLinux

Основное предназначение мейнфрейма — обработка большого количества разнородных операций ввода-вывода. Мейнфрейм может выступать и как средство для проведения сложных вычислений, но, вообще говоря, это не его профиль. Типовым применением для данной платформы могут быть, например, работа с большими базами данных, CRM, SAP и какие-то другие серверные решения, требующие в своей работе выполнения большого количества мелких операций (к примеру, это могут быть банковские или другие транзакции).

В связи с этим разграничение доступа между пользователями одной отдельно взятой ВМ не является основной требуемой функцией для средств защиты информации от несанкционированного доступа (СЗИ НСД). В zLinux, скорее всего, либо вообще не существует реальных пользователей (с аппаратными идентификаторами или другими идентификационными признаками), либо их будет очень маленькое количество (2–3, все пользователи — администраторы с различными ролями), так как основная работа ВМ мейнфрейма в общем случае заключается в работе каких-либо процессов (web-сервер, СУБД и т. п.). Однако, с другой стороны, разграничение доступа сторонними средствами может потребоваться, к примеру, для недопущения повышения привилегий в системе из-за какой-либо уязвимости ОС (ядра ОС, прикладного ПО и т. п.) — в этом случае повышения привилегий в стороннем (внешнем по отношению к ОС) СЗИ НСД происходить не будет.

Вернемся к российским реалиям. Зачастую при обработке какой-либо конфиденциальной информации или персональных данных (а, учитывая профиль использования System z, такие данные с большой вероятностью могут обрабатываться на мейнфрейме) необходимо выполнять ряд нормативных требований, предъявляемых к:

– подсистеме идентификации и аутентификации и контроля доступа;

– подсистеме регистрации и учета (учет носителей информации, входа-выхода субъектов доступа, запуска-завершения программ и процессов, попыток НСД и изменения полномочий субъектов доступа и т. д.);

– подсистеме очистки оперативной памяти и памяти на внешних носителях;

– подсистеме обеспечения целостности;

– криптографической подсистеме (в данной статье эти требования рассматриваться не будут);

– подсистеме антивирусной защиты (в данной статье эти требования рассматриваться также не будут).

Чтобы корректно сформулировать основные требования по защите информации, которые целесообразно применять по отношению к System z, необходимо учитывать следующие особенности:

– процедуру идентификации и аутентификации проходить будут (в основном) только ответственные администраторы с целью первичной настройки, ввода каких-либо сервисов в эксплуатацию либо для проведения технического обслуживания уже функционирующих сервисов;

– обычных пользователей в ВМ System z фактически нет (при профильном использовании), в связи с чем процедура идентификации и аутентификации фактически не будет использоваться при эксплуатации;

– разграничение доступа будет применяться для серверных процессов, запуск которых не инициируется пользователями (администраторами) напрямую;

– System z не предполагает использования каких-либо внешних носителей информации (нет возможности физически подключить какие-либо стандартные носители информации из-за отсутствия соответствующих интерфейсов, в ядре ОС Linux для архитектуры s390x нет USB-стека). В связи с этим требования по очистке (обнулению, обезличиванию) освобождаемых областей памяти на внешних накопителях и по учету внешних носителей являются избыточными;

– System z, вообще говоря, не предполагает использования печати из ВМ (в настоящее время реальные устройства печати для IBM System z уже просто не выпускаются), однако печать на сетевых принтерах в целом не запрещена;

– на System z на данный момент нет возможности использования интерфейсов PCI/PCI-Express/mini-PCI, кроме как для специальных криптографических, сетевых карт и карт ускорения сжатия данных (актуально, так как большинство аппаратных модулей доверенной загрузки (АМДЗ) используют именно такие интерфейсы).

Таким образом, для разграничения доступа (то есть для перечисленных выше подсистем за исключением криптографической подсистемы и подсистемы антивирусной защиты) на zLinux наиболее актуальными требованиями можно считать:

1. идентификацию и аутентификацию субъектов доступа и «отслеживание» создания процессов ОС для однозначного определения субъектов доступа, которые не совершают действий login/logout (демоны, сервисы и другие процессы) и для определения субъектов доступа в лице реальных пользователей (как правило, администраторов);

2. разграничение доступа (в основном для «затруднения» повышения привилегий и для разграничения доступа процессов к объектам доступа), регистрация и учет запуска программ/процессов и попыток НСД;

3. динамический и статический контроль целостности данных;

4. очистка оперативной памяти (для высоких классов защищенности);

5. контроль целостности ОС непосредственно до старта ВМ (то есть обеспечение «доверенной» загрузки ВМ, актуально для более высоких классов защищенности).

Особенности реализации требуемых механизмов защиты для zLinux

Для реализации пунктов 1–4 из предыдущего раздела можно использовать все идеи и наработки, которые уже описаны ранее [1–6].

Причем важно отметить, что в данном случае нет необходимости каким-либо образом изменять или внедрять сторонний код в архитектуру System z – все требования реализуются на уровне ОС Linux и не затрагивают работу каких-либо средств System z / z/VM. Более того, сам гипервизор z/VM позволяет реализовать контроль и управление по многим из перечисленных пунктов на первичном уровне – контроль доступа всей гостевой ОС. Для этого необходимо использовать либо специальную подсистему z/VM RACF (разработка самой компании IBM), либо иной аналогичный программный продукт стороннего производителя для гипервизора z/VM.

Однако, может быть, в целях безопасности придется отказаться от использования некоторых возможностей System z, таких, например, как:

1. common shared memory (разделяемая память) между ВМ;

2. shared kernel in memory («разделяемое» ядро ОС в памяти) на уровне гипервизора, прозрачно для ВМ;

3. возможность подключения дисков к другим ВМ внутри z/VM / LPAR.

Из особенностей, с которыми придется столкнуться при «портировании» решений по защите Linux с других архитектур (x86, x86_64 или др.), можно отметить:

1. невозможность использования архитектурно-зависимого кода (например, код по изменению памяти ядра ОС Linux имеет архитектурную зависимость и должен быть реализован на каждой архитектуре по-своему);

2. другой формат бинарных данных для всех компонентов ОС (если где-либо используются особенности именно бинарного формата, например при поиске в памяти, могут возникнуть проблемы при портировании такого решения);

3. невозможность стандартного использования аппаратных идентификаторов (особенно по интерфейсу USB). Для решения данной проблемы проще всего работать с такими идентификаторами локально на клиенте, инкапсулируя все необходимые данные для идентификации в протоколы удаленного управления (ssh, telnet и т. д.).

Также в zLinux используется свой модифицированный загрузчик zipl (вместо привычных grub/lilo), который позволяет загружать ядро ОС Linux и образ initrd. К счастью, его настройка несильно отличается от настройки grub/lilo, конфигурационный файл /etc/zipl.conf предоставляет возможность задать опции ядра, изменить initrd/ядро ОС или сценарий загрузки по умолчанию (этих операций достаточно для использования существующих решений по защите Linux):

```
defaultboot!
default=linux+accordx+selinux=0
target=/boot/
linux!
        image=/boot/vmlinuz-2.6.18-348.el5
        ramdisk=/boot/initrd-2.6.18-348.el5.img,0x1800000
        parameters=»root=/dev/dasda1»
linux+accordx+selinux=0!
        image=/boot/vmlinuz-2.6.18-348.el5
        ramdisk=/boot/initrd,0x1800000
        parameters="root=/dev/dasda1 selinux=0»
```

Однако если ограничиться требованиями 1–4, то решение может соответствовать только низким классам защищенности — оно по своей сути не будет ничем отличаться от штатных средств защиты ОС Linux, так как будет существовать возможность изменить/отключить защитные механизмы, например, на раннем этапе загрузки ВМ.

Соответственно, для полноценной защиты необходимо дополнительно выполнять контроль целостности компонентов ОС до ее старта на ВМ (требование 5). Таким образом, при одновременном выполнении требований 1–5 может получиться решение, классом защищенности заметно выше. Однако реализовать такое требование непросто, и для этого есть причины:

– в System z нельзя использовать стандартные (аппаратные) АМДЗ, так как этого не позволяет архитектура, однако код гипервизора загружается из энергонезависимой памяти (доступной только на чтение). То есть загрузку гипервизора при загрузке мейнфрейма в целом можно считать «доверенной», так как на сегодняшний день нет возможности вместо гипервизора загружать на мейнфрейме что-либо еще для обеспечения работы множества изолированных ОС типа zLinux;

– в System z какой-либо дополнительный код на уровне гипервизора можно использовать только с ограничениями (код, который позволил бы контролировать целостность ВМ в LPAR, запускаемых на гипервизоре);

Поэтому для контроля целостности ВМ до своего старта можно контролировать целостность компонентов ОС, запущенной на базе z/VM, из другой ВМ (она будет играть роль «программного» АМДЗ и будет контролировать целостность всех ВМ). В процессе запуска

защищаемой ВМ в z/VM можно, используя дополнительный инструмент z/VM RACF, перехватить управление и запустить проверку целостности данных на другой ВМ. Если проверка завершится успешно, на z/VM можно продолжить загрузку защищаемой ВМ с дальнейшим запуском ОС и СПО разграничения доступа (выполняющего, как минимум, требования 1–4), в случае возникновения ошибки ВМ должна быть остановлена с созданием соответствующего события в журнале. При этом расположить «проверочную» ВМ можно и вне z/VM, она может быть доступна для взаимодействия из z/VM, но недоступна для обычных ВМ (сетевое взаимодействие, которое настраивается как угодно с помощью виртуальных сетевых устройств типа vSwitch, можно изолировать).

Краткая иллюстрация предлагаемого решения по защите zLinux приведена на следующей схеме (рис. 1).

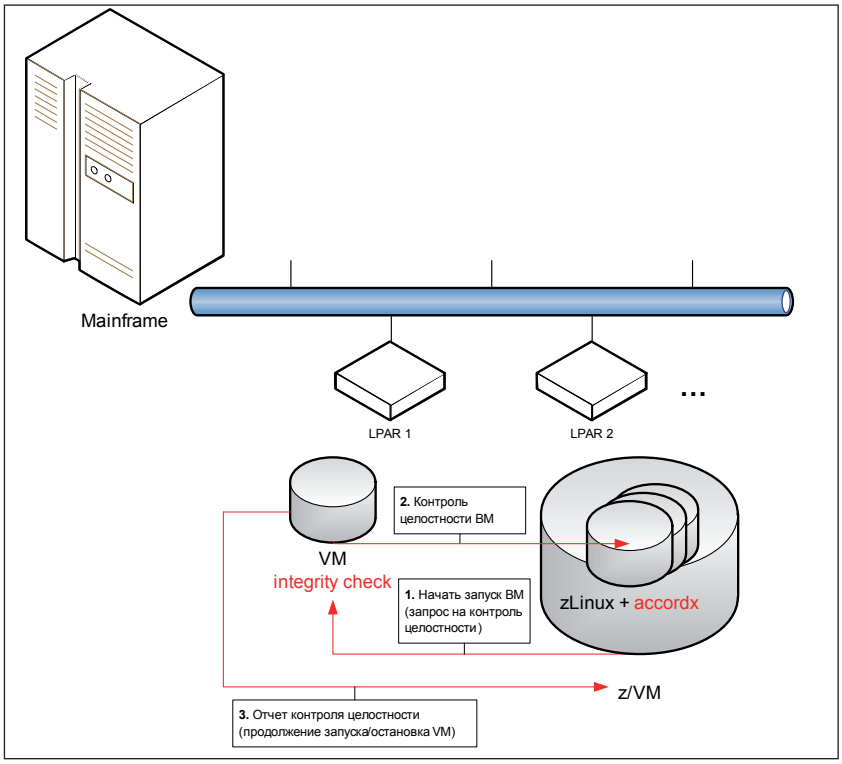

Рис. 1. Решение по защите zLinux

Однако такое решение имеет следующие ограничения:

1. Защищаемые ВМ с zLinux необходимо разворачивать в z/VM (z/VM является отдельным продуктом, и его нужно приобрести на условии OTC/One Time Charge). Справедливости ради надо заметить, что цена z/VM мала по сравнению с другими затратами на развертывание и сопровождение всего аппаратно-программного комплекса в целом, более того, использование z/VM крайне полезно при работе гостевых ОС типа zLinux, в которых проходят разработка, отладка и тестирование прикладного ПО;

2. Для организации перехвата процедуры начальной загрузки (IPL) гостевой ОС zLinux необходимо приобретение продукта z/VM RACF;

3. ВМ, которая будет играть роль «программного» АМДЗ, должна иметь доступ (на чтение) к дискам тех ВМ, которые требуется защищать, а также быть изолированной от этих ВМ;

4. В данном решении мы внедряемся в работу z/VM и, возможно, других компонентов System z;

5. Сложность формализации разграничения доступа администраторов z/VM (необходимо ограничивать полномочия администраторов, в том числе с использованием организационных мер).

Существует также «облегченный» вариант, при котором нет необходимости внедряться в архитектуру z/VM или гипервизор. Всем ВМ можно выделять некоторые, доступные в режиме RO, мини-диски для записи на них СПО разграничения доступа, ядра ОС и всех остальных объектов контроля целостности (целостность которых в первом предложенном варианте защиты необходимо контролировать с помощью «программного» АМДЗ). Такой вариант не устраняет проблему сложности разграничения доступа администраторов z/VM, так как режим RO может быть изменен на RW (доступ самого гипервизора z/VM, вообще говоря, RW).

Заключение

В заключение хотелось бы отметить, что в соответствии с идеями, изложенными выше, для защиты решений на базе zLinux согласно требованиям регуляторов по защите информации в РФ целесообразным видится следующее:

1. произвести портирование имеющейся подсистемы разграничения доступа на архитектуру s390x (zLinux) с учетом особенностей System z;

2. для усиления класса защищенности такого решения сформировать СПО, выполняющее роль «программного» АМДЗ при запуске защищаемых ВМ с zLinux.

Подсистема разграничения доступа при этом может применяться и как отдельное решение на более низкие классы защищенности. На данный момент в ОКБ САПР существует рабочий вариант такого СПО разграничения доступа в виде ПАК СЗИ НСД «Аккорд-Х К», полноценно функционирующего на zLinux.

При этом указанное СЗИ НСД отдельно и совместно с СПО для контроля целостности запускаемых ВМ можно сертифицировать на классы защищенности не ниже классов защищенности СЗИ НСД для защиты решений на базе обычных дистрибутивов ОС Linux.

Также необходимо отметить, что ВМ для контроля целостности в перспективе можно вынести на отдельный носитель информации (поддерживаемый System z), доступ к памяти которого можно на физическом уровне ограничить (например, только на чтение — RO).

СПИСОК ЛИТЕРАТУРЫ:

1. *Каннер А. М.* Linux: о доверенной загрузке загрузчика ОС // Безопасность информационных технологий. 2013. № 2. С. 41–46.
2. *Каннер А. М.* Linux: объекты контроля целостности // Информационная безопасность. Материалы XIII Международной конференции. Таганрог, 2013. Часть 1. С. 112–118.
3. *Каннер А. М.* Linux: к вопросу о построении системы защиты на основе абсолютных путей к объектам доступа // Информационная безопасность. Материалы XIII Международной конференции. Таганрог, 2013. Часть 1. С. 118–121.
4. *Каннер А. М., Ухлинов Л. М.* Управление доступом в ОС GNU/Linux // Вопросы защиты информации. 2012. № 3. С. 35–38.
5. *Parziale L. et al.* The Virtualization Cookbook for IBM z/VM 6.3, RHEL 6.4, and SLES 11 SP3. – SG24-8147-00 IBM Redbooks. 2014.
6. *Parziale L. et al.* Security for Linux on System z – SG24-7728-01 IBM Redbooks. 2013.

REFERENCES:

1. *Kanner A. M.* Linux: About Trusted Boot Loader's startup // Bezopasnost informatsionnykh tekhnologiy. 2013. № 2. P. 41–46.
2. *Kanner A. M.* Linux: integrity control objects // Informatsionnaya bezopasnost. Materiali XIII Mejdunarodnoi konferentsii. Taganrog, 2013. Part 1. P. 112–118.
3. *Kanner A. M.* Linux: developing data security tool based on absolute object paths // Informatsionnaya bezopasnost. Materiali XIII Mejdunarodnoi konferentsii. Taganrog, 2013. Part 1. P. 118–121.
4. *Kanner A. M., Uhlinov L. M.* Access control in GNU/Linux // Voprosi zashiti informatsii. 2012. № 3. P. 35–38.
5. *Parziale L. et al.* The Virtualization Cookbook for IBM z/VM 6.3, RHEL 6.4, and SLES 11 SP3 – SG24-8147-00 IBM Redbooks, 2014.
6. *Parziale L. et al.* Security for Linux on System z – SG24-7728-01 IBM Redbooks, 2013.