# Использование мандатного контроля целостности для изолированного запуска средств контейнеризации в ОС Astra Linux

*Старостин Алексей Александрович,*
*ООО «РусБИТех-Астра», 117105, г. Москва, Варшавское шоссе, д.26,*
*astarostin@astralinux.ru*

## Введение

Механизм мандатного контроля целостности (МКЦ) — фундамент безопасности сертифицированной по высшим классам защиты и уровням доверия операционной системы (ОС) Astra Linux, обеспечивающий наряду с другими механизмами, включая замкнутую программную среду (ЗПС), защиту привилегированных процессов ОС, целостность исполняемых и конфигурационных системных файлов и каталогов ОС, а также пользовательских данных. Использование МКЦ направлено на защиту от вирусов (например, «шифровальщиков»), от эксплуатации многих типовых уязвимостей программного обеспечения (ПО) ОС семейства Linux, в том числе приводящих к атакам нарушителя с правами суперпользователя root. При этом внедрение механизма МКЦ поверх штатного для ОС семейства Linux дискреционного управления доступом представляет существенные трудности, часто требует разработки технологий и сценариев согласованного с ним применения системного и прикладного ПО. В этой связи автором предлагается адаптированная к МКЦ технология контейнерной виртуализации, которая позволяет создавать для недоверенного потенциально «опасного» ПО такого, как браузеры или офисные пакеты, своеобразные «песочницы», где оно изолируется от остального ПО.

## Адаптированная контейнерная виртуализация

Вопросы разработки таких «песочниц» для потенциально «опасного» ПО исследовались в работе [1], посвященной анализу безопасности контейнерной виртуализации в ОС семейства Linux. При этом предлагалось изолировать «опасное» ПО с использованием штатных механизмов ОС таких, как namespaces и cgroups, а также seccomp [2]. Кроме того, в работе [3] исследовались возможности применения для изоляции ПО, имеющегося в пакете безопасности SELinux (LSM-модуле), механизма мандатного управления доступом, предотвращая с его помощью несанкционированный доступ к системным компонентам ОС при наличии уязвимостей, связанных с контейнерами.

В соответствии с документом NIST SP 800-190 «Application Container Security Guide» выделяются следующие виды уязвимостей в контейнерах:

- Уязвимости в ПО контейнера: уязвимость, позволяющая выполнить атаку типа «побег из контейнера» и получить привилегированный доступ на хостовой ОС.
- Неограниченный сетевой доступ из контейнеров: при компрометации контейнера и отсутствии должных мер ограничения сетевого контейнерного трафика, нарушители могут производить сетевые атаки, направленные на соседние контейнеры или сам хост.
- Небезопасные конфигурации контейнеров: при отсутствии разграничения на выполнение высокопривилегированных команд из контейнеров хостовая ОС и «соседние» контейнеры могут быть скомпрометированы.
- Уязвимости в контейнеризованных приложениях: при обнаружении уязвимости в приложении, которое было размещено в контейнере, последствия от ее эксплуатации могут включать в себя компрометацию хостовой ОС или других контейнеров.

Наличие МКЦ в ОС Astra Linux позволяет снизить вероятность реализации этих атак за счет запуска контейнеров на промежуточном неиерархическом уровне целостности или на отрицательных линейных уровнях целостности. В этом случае недоверенное ПО, даже если подвергнется атаке нарушителя или заражению вирусом, не будет представлять опасности для всей остальной системы.

Например, для защиты доверенного ПО, недоверенное ПО запускается в контейнере-«песочнице» на неиерархическом уровне целостности «Виртуализация»

0x00000002, а само доверенное ПО – на максимальном уровне целостности «Высокий», используемом в ОС Astra Linux по умолчанию – 0x0000003F.

Для предотвращения атак между контейнерами могут применяться линейные уровни целостности, например, -128 и -10, что позволяет выполнить запуск этих контейнеров на разных уровнях, защищая контейнер с большим уровнем целостности (0x00000002:-10) от атаки из контейнера с меньшим уровнем целостности (0x00000002:-128), так как при попытках получения соответствующего доступа на запись из второго контейнера это будет запрещено механизмом МКЦ. При этом изоляция второго контейнера от первого не осуществляется, т. е. предполагается, что в первом контейнере функционирует менее «опасное» ПО, чем во втором.

Вместе с тем далеко не всегда нарушитель нацелен на захват управления над ОС в целом. Иногда ему достаточно зашифровать данные непривилегированного пользователя, чтобы достигнуть экономических целей своей атаки. Ранее при таком сценарии при запуске контейнера в сессии непривилегированного пользователя и реализации атаки вида «побег из контейнера» нарушитель мог получить доступ ко всем данным (файлам или каталогам) соответствующего пользователя, так как они имеют тот же неиерархический уровень целостности «Низкий» (0x00000000:0). Однако, если запускать контейнер, управление над которым может потенциально захватить нарушитель, на пониженном линейном уровне целостности (например, на 0x00000000:-128), то механизм МКЦ не позволит нарушителю получить доступ на запись к данным, находящимся на уровне целостности «Низкий» 0x00000000:0. Таким образом, существенно снижается риск успешного выполнения атак вирусов-«шифровальщиков» при захвате нарушителем контейнера, функционирующего от имени учетной записи непривилегированного пользователя.

**Заключение**

Рассмотренная технология является примером того, как важна при использовании контейнерной виртуализация основанная на применении механизма МКЦ многоуровневая защита. При этом можно отметить, что рассматриваемая технология базируется на использовании механизма МКЦ, который в отличие от многих других механизмов защиты (например, замкнутая программная среда или блокировка интерпретаторов) работает по умолчанию, начиная с основного режима защиты ОС Astra Linux. При этом замкнутая программная среда или блокировка интерпретаторов ограничивают состав разрешенного для использования ПО, а МКЦ – нет.

При использовании контейнерной виртуализации могут появляться дополнительные уязвимости, в том числе в составе компонент ядра ОС. Однако для нарушителя проэксплуатировать эти уязвимости будет гораздо сложнее, когда контейнер функционирует на отрицательном линейном уровне целостности. При этом с учетом изоляции контейнеров средствами ОС для выявления аномального «поведения» атакованных нарушителем контейнеров в перспективе возможно использование адаптированных к МКЦ методов машинного обучения.

**Список литературы**

[1]. Wan Z., Lo D., Xia X., L. Cai. Practical and effective sandboxing for Linux containers / Empir Software Eng, 2019, Vol. 24, pp. 4034–4070.

[2]. N. Lopes, R. Martins, M.E. Correia, S. Serrano, F. Nunes. Container Hardening Through Automated Seccomp Profiling // In Proceedings of the 2020 6th International Workshop on Container Technologies and Container Clouds, 2020, pp. 31–36.

[3]. J.-A. Kabbe. Security analysis of Docker containers in a production environment // Norwegian University of Science and Technology, 2017, 91 p.