AXIOM JDK

Axiom Trusted Images 25 LTS Руководство администратора

Axiom Trusted Images | Октябрь 2026

Copyright © 2019-2026 Все права защищены АО "АКСИОМ" (АКСИОМ)

Программное обеспечение АКСИОМ содержит программное обеспечение с открытым исходным кодом. Дополнительная информация о коде сторонних разработчиков доступна на сайте https://axiomjdk.ru/third_party_licenses. Для дополнительной информации о том, как получить копию исходного кода, можно обратиться по адресу info@axiomjdk.ru.

ДАННАЯ ИНФОРМАЦИЯ МОЖЕТ ИЗМЕНЯТЬСЯ БЕЗ ПРЕДВАРИТЕЛЬНОГО УВЕДОМЛЕНИЯ. АКСИОМ ПРЕДОСТАВЛЯЕТ ПРОГРАММНОЕ ОБЕСПЕЧЕНИЕ "КАК ЕСТЬ" БЕЗ КАКИХ-ЛИБО ГАРАНТИЙ, АКСИОМ ПРЯМО ОТКАЗЫВАЕТСЯ ОТ ВСЕХ ПОДРАЗУМЕВАЕМЫХ ГАРАНТИЙ, ВКЛЮЧАЯ, НО НЕ ОГРАНИЧИВАЯСЬ ПОДРАЗУМЕВАЕМЫМИ ГАРАНТИЯМИ ТОВАРНОЙ ПРИГОДНОСТИ И ПРИГОДНОСТИ ДЛЯ ОПРЕДЕЛЕННОЙ ЦЕЛИ.

АКСИОМ НИ ПРИ КАКИХ ОБСТОЯТЕЛЬСТВАХ НЕ НЕСЕТ ОТВЕТСТВЕННОСТИ ЗА ЛЮБЫЕ КОСВЕННЫЕ, СЛУЧАЙНЫЕ, СПЕЦИАЛЬНЫЕ, ШТРАФНЫЕ ИЛИ КОСВЕННЫЕ УБЫТКИ, ИЛИ УБЫТКИ ОТ ПОТЕРИ ПРИБЫЛИ, ДОХОДА, ДАННЫХ ИЛИ ИСПОЛЬЗОВАНИЯ ДАННЫХ, ПОНЕСЕННЫЕ ВАМИ ИЛИ ЛЮБОЙ ТРЕТЬЕЙ СТОРОНОЙ, БУДЬ ТО В РЕЗУЛЬТАТЕ ДЕЙСТВИЯ ДОГОВОРА ИЛИ ДЕЛИКТА, ДАЖЕ ЕСЛИ АКСИОМ БЫЛО ПРЕДУПРЕЖДЕНО О ВОЗМОЖНОСТИ ТАКИХ УБЫТКОВ.

Использование любого программного продукта АКСИОМ регулируется соответствующим лицензионным соглашением, которое никоим образом не изменяется условиями данного уведомления. Программные продукты и фирменные наименования: Axiom JDK, Axiom JDK Pro, Axiom Runtime Container Pro, Axiom Linux, Libercat, Libercat Certified и АКСИОМ принадлежат АКСИОМ и их использование допускается только с разрешения правообладателя.

Товарный знак Linux® используется в соответствии с сублицензией от Linux Foundation, эксклюзивного лицензиата Линуса Торвальдса, владельца знака на всемирной основе. Java и OpenJDK являются товарными знаками или зарегистрированными товарными знаками компании Oracle и/или ее аффилированных лиц. Другие торговые марки являются собственностью их соответствующих владельцев и используются только в целях идентификации.

Содержание

1. Введение 5

1.1. Axiom Trusted Images 25 LTS в производственной среде 5

2. Системные требования и настройка среды 6

2.1. Системные требования 6

2.2. Программные требования 6

3. Проверка подписей 7

4. Выгрузка SBOM 8

5. Перечень образов 9

5.1. 25-trusted-base 9

5.2. 25-trusted-axiom-runtime-container-pro 9

5.3. 25-trusted-axiom-runtime-container-certified 9

5.4. 25-trusted-axiom-runtime-container-express 9

5.5. 25-trusted-axiom-native-image-kit-container 10

5.6. 25-trusted-gcc 10

5.7. 25-trusted-go 11

5.8. 25-trusted-python 12

5.9. 25-trusted-nginx 12

5.10. 25-trusted-node 12

1. Введение

Настоящий документ содержит сведения по настройке и администрированию программного обеспечения Axiom Trusted Images 25 LTS. Данный документ предназначен для системных администраторов, разработчиков и команд по внедрению.

1.1. Axiom Trusted Images 25 LTS в производственной среде

Axiom Trusted Images 25 LTS — это набор готовых контейнеров с образами на базе ОС Axiom Linux. Образы могут включать Axiom JDK Pro, Axiom NIK Pro, интерпретатор языка Python, компиляторы GCC и Go, среду выполнения NodeJS и другие утилиты и библиотеки. См. Перечень образов.

Все образы поставляются в версии `latest`, что означает использование последних версий компонентов (например Axiom JDK) и libc библиотеки musl. Если часть `:<тег образа>` опущена, она заменяется на `:latest`, то есть команды работают с тегом `latest`.

2. Системные требования и настройка среды

2.1. Системные требования

Для работы с образами Axiom Trusted Images 25 LTS нужны:

- Процессор с 64-битной архитектурой aarch64, x86_64 и поддержкой виртуализации.
- Оперативная память: обратитесь к документации вашей ОС. Как правило, необходимый минимум - 4 ГБ RAM.
- Операционная система:
  - Linux (рекомендуемая среда; поддерживаются основные дистрибутивы: Ubuntu, CentOS, Debian и т.д.);
  - Windows (с WSL 2 или Hyper-V);
  - macOS (с Hypervisor.framework или через Docker Desktop).
- Место на диске: достаточно свободного пространства для хранения образов и контейнеров (зависит от количества и размера образов).

2.2. Программные требования

Установленный Docker Engine, Docker Desktop или утилита командной строки Podman.

3. Проверка подписей

Все поставляемые образы подписаны ключом АО «АКСИОМ» с использованием программы cosign.

Для проверки подписи необходимо выполнить следующие шаги:

1. Скачайте SSL сертификат с ключом

```
curl -O https://download.axiomjdk.ru/ssl-certificates/axiom-jsc-image-signing.crt
```

2. Скачайте корневой сертификат к нему

```
curl -O https://download.axiomjdk.ru/ssl-certificates/axiom-jsc-ca.crt
```

3. Проверьте, что сертификат с ключом действительно выпущен корневым сертификатом

```
openssl verify -CAfile axiom-jsc-ca.crt -show_chain axiom-jsc-image-signing.crt
```

4. Извлеките публичный ключ из сертификата

```
openssl x509 -in axiom-jsc-image-signing.crt -noout -pubkey -out public.key
```

5. Проверьте подпись образа

```
cosign verify --key public.key --private-infrastructure=true <идентификатор образа>
```

4. Выгрузка SBOM

К каждому образу прикреплены 2 файла SBOM в форматах:

- CycloneDX 1.6
- SPDX 2.3

Эти файлы содержат подписанные списки компонент, из которых состоит образ.

Для выгрузки SBOM потребуется программа `cosign` и публичный ключ из раздела Проверка подписей.

Для получения файла в формате CycloneDX выполните команду:

```
cosign verify-attestation \
    --key public.key \
    --type cyclonedx \
    <идентификатор образа> | jq -r '.payload' | base64 -d | jq '.predicate'
```

Для получения файла в формате SPDX выполните команду:

```
cosign verify-attestation \
    --key public.key \
    --type spdx \
    <идентификатор образа> | jq -r '.payload' | base64 -d | jq '.predicate'
```

5. Перечень образов

5.1. 25-trusted-base

Это базовый образ с минимальным набором компонент ОС для запуска статических слинкованных программ, либо программ, зависящих только от стандартной библиотеки C.

Примеры использования этого образа приведены в последующих разделах.

5.2. 25-trusted-axiom-runtime-container-pro

Данный образ содержит Axiom JDK Pro 27.

Ниже представлен пример Dockerfile, в котором Java приложение компилируется в образе 25-trusted-axiom-runtime-container-pro:

```
FROM <...>/25-trusted-axiom-runtime-container-pro:latest AS builder

COPY <<'EOF' HelloAxiom.java
public class HelloAxiom {
    public static void main(String[] args) {
        System.out.println("Hello Axiom!");
    }
}
EOF
```

5.3. 25-trusted-axiom-runtime-container-certified

Данный образ поставляется с Axiom JDK Certified 21.

Работа с образом 25-trusted-axiom-runtime-container-certified осуществляется аналогично образу 25-trusted-axiom-runtime-container-pro.

5.4. 25-trusted-axiom-runtime-container-express

Репозиторий содержит образ с Axiom JDK Express 11.

Работа с образом 25-trusted-axiom-runtime-container-express осуществляется аналогично образу

25-trusted-axiom-runtime-container-pro.

5.5. 25-trusted-axiom-native-image-kit-container

Репозиторий содержит образ с Axiom NIK 25.

Ниже представлен пример Dockerfile, в котором Java приложение с помощью NIK компилируется в бинарный файл, который затем запускается в базовом образе:

```
FROM <...>/25-trusted-axiom-native-image-kit-container:latest AS builder

COPY <<'EOF' HelloAxiom.java
public class HelloAxiom {
    public static void main(String[] args) {
        System.out.println("Hello Axiom!");
    }
}
EOF

RUN javac HelloAxiom.java
RUN native-image --no-fallback -o hello-axiom HelloAxiom

FROM <...>/25-trusted-base:latest

COPY --from=builder /home/appuser/hello-axiom .

ENTRYPOINT [ "/home/appuser/hello-axiom" ]
```

5.6. 25-trusted-gcc

Репозиторий содержит образ с инструментами для C/C++ разработки на базе GCC 14.3.

Ниже представлен пример Dockerfile со сборкой C приложения с помощью образа 25-trusted-gcc с последующим запуском в базовом образе:

```
FROM <...>/25-trusted-gcc:latest AS builder

COPY <<'EOF' hello_axiom.c
#include <stdio.h>

int main(int argc, char *argv[]) {
    printf("Hello Axiom!\n");
```

```
}
EOF

RUN gcc -Wall -o hello-axiom hello_axiom.c

FROM <...>/25-trusted-base:latest

COPY --from=builder /home/appuser/hello-axiom .

ENTRYPOINT [ "/home/appuser/hello-axiom" ]
```

5.7. 25-trusted-go

Репозиторий содержит образ с установленным компилятором Go и вспомогательными утилитами и библиотеками. Образ предназначен для сборки и отладки программ написанных на Go.

Образ поставляется только с версией Go, поддерживаемой разработчиками языка Go (https://go.dev).

Ниже представлен пример Dockerfile со сборкой Go приложения с помощью образа 25-trusted-go с последующим запуском в базовом образе:

```
FROM <...>/25-trusted-go:latest AS builder

COPY <<'EOF' hello_axiom.go
package main

import "fmt"

func main() {
    fmt.Println("Hello Axiom")
}
EOF

RUN go build -o hello-axiom hello_axiom.go

FROM <...>/25-trusted-base:latest

COPY --from=builder /home/appuser/hello-axiom .

ENTRYPOINT [ "/home/appuser/hello-axiom" ]
```

5.8. 25-trusted-python

Репозиторий содержит образ с Python 3.12 и базовыми Python утилитами (pip, setuptools, wheel).

Пример Dockerfile с запуском приложения в образе 25-trusted-python:

```
FROM <...>/25-trusted-python:latest

COPY <<'EOF' hello_axiom.py
if __name__ == "__main__":
    print("Hello Axiom")
EOF

CMD [ "hello_axiom.py" ]
```

5.9. 25-trusted-nginx

Репозиторий содержит образ с веб-сервером Nginx и содержит минимальный набор пакетов, необходимых для работы `nginx`. Не содержит никаких динамических модулей `nginx`.

Следующий пример кода демонстрирует Dockerfile, который создает образ для размещения простого статического контента:

```
FROM 25-trusted-nginx:latest
COPY static-html-directory /var/lib/nginx/html
```

Выполните следующие команды в директории, где находится директория `static-html-directory`, чтобы собрать и запустить контейнер:

```
$ docker build -t my-nginx .
$ docker run -d -p 8080:8080 my-nginx
```

После этого вы сможете получить доступ к http://localhost:8080.

5.10. 25-trusted-node

Репозиторий содержит образы с Node.js и базовыми утилитами (npm):

Образ поставляется только с версией Node.js, поддерживаемой разработчиками Node.js (https://nodejs.org).

Для корректной обработки сигналов ОС рекомендуется в Dockerfile запускать процесс `node` не напрямую, а используя прослойку `/usr/sbin/tini`, которая уже предустановлена во всех образах.

Пример Dockerfile с запуском приложения в образе 25-trusted-node:

```
FROM <...>/25-trusted-node:latest

COPY <<'EOF' hello_axiom.js
import { createServer } from 'node:http';

const server = createServer((req, res) => {
  res.writeHead(200, { 'Content-Type': 'text/plain' });
  res.end('Hello Axiom!\n');
});

server.listen(5000, '0.0.0.0', () => {
  console.log('Listening on 0.0.0.0:5000');
});
EOF

ENTRYPOINT [ "/usr/sbin/tini", "--"]
CMD [ "node", "hello_axiom.js"]
```

AXIOM JDK

Axiom Trusted Images 25 LTS
Руководство администратора