PUBLIC REPORT

POA Network Token-Bridge Security Assessment

peppersec

Foreword

Status of a vulnerability or security issue

Clarity is a rare commodity. That is why for the convenience of both the client and the reader, we have introduced a system of marking vulnerabilities and security issues we discover during our security audits.

No issue

Let’s start with an ideal case. If an identified security imperfection bears no impact on the security of our client, we mark it with the label.

✓ Fixed

The fixed security issues get the label that informs those reading our public report that the flaws in question should no longer be worried about.

Addressed

In case a client addresses an issue in another way (e.g., by updating the information in the technical papers and specification) we put a nice tag right in front of it.

Acknowledged

If an issue is planned to be addressed in the future, it gets the tag, and a client clearly sees what is yet to be done.

Although the issues marked “Fixed“ and “Acknowledged“ are no threat, we still list them to provide the most detailed and up-to-date information for the client and the reader.

Severity levels

We also rank the magnitude of the risk a vulnerability or security issue pose. For this purpose, we use 4 “severity levels“ namely:

1. Minor
2. Medium
3. **Major**
4. **Critical**

More details about the ranking system as well as the description of the severity levels can be found in **Appendix 1. Terminology**.

TABLE OF CONTENTS

2. Docker daemon configuration	
2.1. Ensure network traffic is restricted between containers on the default bridge	✓ Fixed
2.2. Enable user namespace support	Acknowledged
2.14. Ensure live restore is Enabled	✓ Fixed
2.18. Ensure containers are restricted from acquiring new privileges	✓ Fixed
4. Container Images and Build File	
4.5. Ensure Content trust for Docker is Enabled	Acknowledged
4.6. Ensure HEALTHCHECK instructions have been added to the container image	Acknowledged
5. Container Runtime	
5.10. - Ensure memory usage for container is limited	✓ Fixed
5.11. - Ensure CPU priority is set appropriately on the container	✓ Fixed
5.12. - Ensure the container’s root filesystem is mounted as read only	Acknowledged
5.14. Ensure ‘on-failure’ container restart policy is set to ‘5’	✓ Fixed
5.28. Ensure PIDs cgroup limit is used	Acknowledged

CONCLUSIONS

APPENDIX 1. TERMINOLOGY

1. Severity

APPENDIX 2. DOCKER BENCH SECURITY OUTPUT

01. Introduction

1 Source code

Object	Location
POA Network Token-bridge. Branch support-erc20-native-#81	#2173d84694270ca032c34fcb1e6dc4e16fe81201
POA consensus. The RewardByBlock smart contract only.	RewardByBlock.sol
POA Bridge smart contracts. Branch erc20-to-native-#79	#9487e38e6adda86feced5984ad449508b343eee9

2 Security assessment methodology

A client's information resources may be affected by the following types of threats:

1. Confidentiality violations with resulting information disclosure. A successful violation of the kind leads to information becoming known to people, who should not have access to it: unauthorized personnel, clients, partners, competitors, and third parties on the whole.
2. Integrity violations and consequent modification or corruption of data leading to changes in its structure or content, and to complete or partial destruction of the data.
3. Availability violation (denial-of-service) causes the inability of a user to access data.

The primary purpose of this security assessment is to estimate the likelihood of the system being exploited as well as the threats above being carried out by an attacker, who belongs to a predefined attacker model.

By "an attacker" we mean an individual adversary or a group of adversaries acting either on their own or in cooperation, whose intentional or unintentional actions may become a dire threat to our clients and their systems.

More details on the security assessment workflow we stick to can be found here.

3 Auditors

1. Alexey Pertsev

02. Summary

Below, you can find a table with all the discovered bugs and security issues listed.

Vulnerability description	Severity
Service doesn’t restart workers after crush	Major
Bad private data storing	Medium
Selecting validators for executeSignatures is not optimal	
RabbitMQ container has unused Management Plugin	
Deployment playbook improvement	
setBlockRewardContract improvement	Minor
The app relies on smart contract immutability	
Block explorer is not going to show RewardByBlock transfers	None
The app doesn’t check Certificate revocation	
Ensure a separate partition for containers has been created	
Ensure auditing is configured for the Docker daemon	
Ensure network traffic is restricted between containers on the default bridge	
Enable user namespace support	
Ensure live restore is Enabled	
Ensure containers are restricted from acquiring new privileges	
Ensure Content trust for Docker is Enabled	
Ensure HEALTHCHECK instructions have been added to the container image	
Ensure memory usage for container is limited	
Ensure CPU priority is set appropriately on the container	
Ensure the container’s root filesystem is mounted as read only	
Ensure ‘on-failure’ container restart policy is set to ‘5’	
Ensure PIDs cgroup limit is used	

03. Incremental audit of POA Bridge Smart Contracts

HomeBridgeErcToNative

1 setBlockRewardContract improvement

Severity: MINOR

The setBlockRewardContract(address _blockReward) method ensures _blockReward address is not equal to zero and that is contract. However, that is not enough to be sure _blockReward is the address of the actual RewardByBlock contract (it may be an address of any contract).

Recomendations:

1. Consider implementing a view call of RewardByBlock, which returns some specific value. For example:

```
function setBlockRewardContract(address _blockReward) public onlyOwner {
        require(IBlockReward(_blockReward).bridgeContract() == this);
        ...
}
```

Status:

✓ Fixed PR82

2 Block explorer is not going to show RewardByBlock transfers

Severity: NONE

Due to the current architecture, the native tokens are minted by a special consensus mechanism via RewardByBlock contract. So there are no transfer transactions that could be viewed at blockchain. Also, events in the reward func are not observable. In this case, the only way to get the confirmation that token minting uses the mintedForAccountInBlock(_ account and the _blockNumber) func. If it is necessary to get all rewarded in a specific block, then AddedReceiver could be used to get a bunch of addresses.

04. POA Network bridge security assessement

General security issues

1 The app doesn't check Certificate revocation

Severity: NONE

In case of the compromise of certificate private key of HOME_RPC_URL/FOREIGN_RPC_URL, an attacker can launch an MITM attack even after certificate revocation (https library does not care about that).

Recommendations:

1. Impact of that attack can be huge, but its likelihood is way too small and can be done by hosting or provider only. That is why there is no issue here.

2 Bad private data storing

Severity: MEDIUM

Token-bridge app uses validator private key to sign transactions to Bridge smart contracts.

1. According to best practices, although secrets should not be stored in a container image, it still is.
2. Also a private key is stored in all watcher containers, but it is not necessary for work. Exception: signature-request watcher uses private key to prepare args for submitSignature function.
3. Also a private key is stored in a host system in the home directory of the user who deployed service by ansible-playbook.

Recommendations:

1. Put the private key in /root directory and set appropriate file permissions.
2. When the bridge service starts workers, pass private key as a parameter for docker-compose to appropriate workers only.
3. Start containers as a non-root user (see options here)

Status:

✓ Fixed PR85 Now all watchers can work without a private key (except signature-request watcher). Other requirements satisfied are in PR38.

3 Service doesn’t restart workers after crush

Severity: MAJOR

The current deployment playbook does not run stable Bridge instance. After some crush workers are not going to be up.

Steps to reproduce. Case 1:

1. Set a private key beginning with 0x.
2. Deploy the service and wait for about 10 minutes, after that some containers will get the `exited` status.

Steps to reproduce. Case 2:

1. Deploy the service as usual.
2. After several hours of expected working, `rabbitMQ` container is going to have status `exit(0)`. Other workers will show message “`Disconnected from amqp Broker`”.

Recommendations:

3. change restart option to `always`.

Status:

✓ Fixed PR116

4 Selecting validators for executeSignatures is not optimal

Severity: MEDIUM

At the time of the security assessment, the last Validator who called `HomeBridge.submitSignature` should also call `ForeignBridge.executeSignatures`, which has real Gas cost for the transaction (to a Foreign network). This scheme means the validator with the biggest network delay (ping to a public RPC) will constantly pay commission. On the other hand, that means that if the Validator goes offline, a transfer will be stopped because token-bridge has no delivery checks by other Validators.

Also, this topic is related to the issue #51 of parity-bridge - Validator should take commission from users for transfers, otherwise destructive actions possible.

Recommendations:

1. Consider implement strict policy for `ForeignBridge.executeSignatures calling`.

Status:

Acknowledged

5 RabbitMQ container has unused Management Plugin

Severity: **MEDIUM**

The current Token-bridge implementation uses RabbitMQ with Management Plugin.
It seems the plugin was used for debugging or developing, but it’s not necessary for production. By default, the plugin is available on the standard management port of 15672, with the default username and password: `guest` / `guest`.

Recommendations:

1. To reduce the attack surface, consider using RabbitMQ without Management Plugin or set a strong password for it.

Status:

✓ Fixed PR116

6 The app relies on smart contract immutability

Severity: **MINOR**

The current token-bridge implementation relies on smart contract immutability and data processing by the smart contract. That means if the smart contract is broken and may be exploited, token-bridge has not much opportunity to prevent this.

Regarding software architecture, we can consider token-bridge and poa-bridge smart contracts as entire app hence token-bridge does not have to special data validation. Moreover, all `watchers` do checks (asserts) before sending the next message to RabbitMQ queue by calling `estimateGas` for the `executeAffirmation` and `executeSignatures` functions.

On the other hand, in production token-bridge and poa-bridge contracts will have middleware in the form of a public node RPC, which may be controlled by a third-party person. Hence all token-bridge checks can be bypassed, and any data can be supplied to the token-bridge app.

Recommendations:

1. Use different RPC for each token-bridge instance to eliminate rogue node problem.

7 Deployment playbook improvement

Severity: **MEDIUM**

The current deployment playbook expects a properly configured host, which has a special user for the Docker management.

Recommendation:

1. Consider changing default user to `root` (or sudoer one) and implementing an explicit creation of Docker user.

Status:

✓ Fixed PR40

Hardening

1 Host Configuration

1.1. Ensure a separate partition for containers has been created

In case of the Token-bridge app, it would be useful to separate the private key and Docker containers (see also paragraph **Bad private data storing**).

Status:

Acknowledged

1.5 - 1.13. Ensure auditing is configured for the Docker daemon

It is recommended to install and configure the `auditd` tools to enable auditing of some of Docker's files, directories, and sockets. It would be useful in case of incident response.

Status:

✓ Fixed **PR40**

2 Docker daemon configuration

2.1. Ensure network traffic is restricted between containers on the default bridge

In case of the Token-bridge app, it would be useful to separate the private key and Docker containers (see also paragraph **Bad private data storing**).

Status:

✓ Fixed **PR116**

2.2. Enable user namespace support

Linux namespaces provide additional isolation for running processes in your containers. User namespace remapping allows processes to run as root in a container while being remapped to a less privileged user on the host. See options “userns-remap”: “default” and recommendations in paragraph **Bad private data storing**.

Status:

Acknowledged

2.14. Ensure live restore is Enabled

Specifying “live-restore”: true in the Docker daemon config, enables containers to go on running when the Docker daemon is not. This improves container uptime during updates of the host system and other stability issues.

Status:

✓ Fixed **PR40**

2.18. Ensure containers are restricted from acquiring new privileges

The “no-new-privileges”: true line in the daemon config prevents privilege escalation from inside of containers. This ensures that containers cannot gain new privileges using setuid or setgid binaries.

Status:

✓ Fixed **PR40**

4 Container Images and Build File

4.5. Ensure Content trust for Docker is Enabled

The app uses RabbitMQ and Redis public docker images, so it may be useful to check signs before running. See the DOCKER_CONTENT_TRUST environment variable.

Status:

Acknowledged

4.6. Ensure HEALTHCHECK instructions have been added to the container image

The HEALTHCHECK instruction could be useful for monitoring. Consider adding health status of all existing bridge instances to **bridge monitor**.

Status:

Acknowledged

5 Container Runtime

5.10. - Ensure memory usage for container is limited

The current configuration does not limit consuming containers memory, which can be dangerous. See **here** for an explanation and useful options.

Status:

✓ Fixed **PR116**

5.11. - Ensure CPU priority is set appropriately on the container

The current configuration does not limit consuming containers CPU, which can be dangerous. See **here** for an explanation and useful options.

Status:

✓ Fixed **PR116**

5.12. - Ensure the container's root filesystem is mounted as read only

You may set `root` filesystem to `read_only` to mitigate possible privilege escalation since the container instance's filesystem cannot be tampered with or written to unless it has explicit read-write permissions on its filesystem folder and directories. See options **here**.

Status:

Acknowledged

5.14. Ensure 'on-failure' container restart policy is set to '5'

Change the **restart option** to `always`. It should help with unexpected container shutdown (**paragraph 3**).

Status:

✓ Fixed **PR116**

5.28. Ensure PIDs cgroup limit is used

Attackers can launch a fork bomb with a single command inside the container. This fork bomb can crash the entire system and requires a restart of the host to make the system operating again. PIDs cgroup `pids_limit: 100` will prevent this kind of attacks by restricting the number of forks that can happen inside a container at a given time. Set the PIDs limit value as appropriate. Incorrect values may leave the containers unusable.

Status:

Acknowledged

Conclusions

The security assessment of POA Network Token-Bridge indicated a set of vulnerabilities of different severity levels. However, most of them pose no threat to the client’s system.

All the discovered security issues and vulnerabilities have been already either fixed completely or acknowledged by the client.

Considering these, we can rate the overall security level of the system as “**High**”.

Appendix 1. Terminology

1 Severity

Severity is the category that described the magnitude of an issue.

		Severity		
Impact	Major	Medium	Major	Critical
	Medium	Minor	Medium	Major
	Minor	None	Minor	Medium
		Minor	Medium	Major
		Likelihood		

MINOR

Minor issues are generally subjective in their nature or potentially associated with the topics like “best practices” or “readability”. As a rule, minor issues do not indicate an actual problem or bug in the code. The maintainers should use their own judgment as to whether addressing these issues will improve the codebase.

MEDIUM

Medium issues are generally objective in their nature but do not represent any actual bugs or security problems. These issues should be addressed unless there is an apparent reason not to.

MAJOR

Major issues are things like bugs or vulnerabilities. These issues may be unexploitable directly or may require a certain condition to arise to be exploited. If unaddressed, these issues are likely to cause problems with the operation of the contract or lead to situations which make the system exploitable.

CRITICAL

Critical issues are directly exploitable bugs or security vulnerabilities. If unaddressed, these issues are likely or guaranteed to cause major problems and ultimately a full failure in the operations of the contract.

Appendix 2. Docker bench security output

```
root@cs363758:~/docker-bench-security# ./docker-bench-security.sh
# ------------------------------------------------------------------------------
# Docker Bench for Security v1.3.4
#
# Docker, Inc. (c) 2015-
#
# Checks for dozens of common best-practices around deploying Docker containers in
production.
# Inspired by the CIS Docker Community Edition Benchmark v1.1.0.
# ------------------------------------------------------------------------------


Initializing Wed Oct 10 14:01:45 MSK 2018



[INFO] 1 - Host Configuration
[WARN] 1.1  - Ensure a separate partition for containers has been created
[NOTE] 1.2  - Ensure the container host has been Hardened
[INFO] 1.3  - Ensure Docker is up to date
[INFO]      * Using 18.06.1, verify is it up to date as deemed necessary
[INFO]      * Your operating system vendor may provide support and security maintenance for
Docker
[INFO] 1.4  - Ensure only trusted users are allowed to control Docker daemon
[INFO]      * docker:x:999:
[WARN] 1.5  - Ensure auditing is configured for the Docker daemon
[WARN] 1.6  - Ensure auditing is configured for Docker files and directories - /var/lib/
docker
[WARN] 1.7  - Ensure auditing is configured for Docker files and directories - /etc/docker
[WARN] 1.8  - Ensure auditing is configured for Docker files and directories - docker.service
[WARN] 1.9  - Ensure auditing is configured for Docker files and directories - docker.socket
[WARN] 1.10 - Ensure auditing is configured for Docker files and directories - /etc/default/
docker
[INFO] 1.11 - Ensure auditing is configured for Docker files and directories - /etc/docker/
daemon.json
```

```
[INFO]      * File not found
[WARN] 1.12 - Ensure auditing is configured for Docker files and directories - /usr/bin/
docker-containerd
[WARN] 1.13 - Ensure auditing is configured for Docker files and directories - /usr/bin/
docker-runc


[INFO] 2 - Docker daemon configuration
[WARN] 2.1  - Ensure network traffic is restricted between containers on the default bridge
[PASS] 2.2  - Ensure the logging level is set to ‘info’
[PASS] 2.3  - Ensure Docker is allowed to make changes to iptables
[PASS] 2.4  - Ensure insecure registries are not used
[PASS] 2.5  - Ensure aufs storage driver is not used
[INFO] 2.6  - Ensure TLS authentication for Docker daemon is configured
[INFO]      * Docker daemon not listening on TCP
[INFO] 2.7  - Ensure the default ulimit is configured appropriately
[INFO]      * Default ulimit doesn’t appear to be set
[WARN] 2.8  - Enable user namespace support
[PASS] 2.9  - Ensure the default cgroup usage has been confirmed
[PASS] 2.10 - Ensure base device size is not changed until needed
[WARN] 2.11 - Ensure that authorization for Docker client commands is enabled
[WARN] 2.12 - Ensure centralized and remote logging is configured
[INFO] 2.13 - Ensure operations on legacy registry (v1) are Disabled (Deprecated)
[WARN] 2.14 - Ensure live restore is Enabled
[WARN] 2.15 - Ensure Userland Proxy is Disabled
[PASS] 2.16 - Ensure daemon-wide custom seccomp profile is applied, if needed
[PASS] 2.17 - Ensure experimental features are avoided in production
[WARN] 2.18 - Ensure containers are restricted from acquiring new privileges


[INFO] 3 - Docker daemon configuration files
[PASS] 3.1  - Ensure that docker.service file ownership is set to root:root
```

```
[PASS] 3.2  - Ensure that docker.service file permissions are set to 644 or more restrictive
[PASS] 3.3  - Ensure that docker.socket file ownership is set to root:root
[PASS] 3.4  - Ensure that docker.socket file permissions are set to 644 or more restrictive
[PASS] 3.5  - Ensure that /etc/docker directory ownership is set to root:root
[PASS] 3.6  - Ensure that /etc/docker directory permissions are set to 755 or more
restrictive
[INFO] 3.7  - Ensure that registry certificate file ownership is set to root:root
[INFO]      * Directory not found
[INFO] 3.8  - Ensure that registry certificate file permissions are set to 444 or more
restrictive
[INFO]      * Directory not found
[INFO] 3.9  - Ensure that TLS CA certificate file ownership is set to root:root
[INFO]      * No TLS CA certificate found
[INFO] 3.10 - Ensure that TLS CA certificate file permissions are set to 444 or more
restrictive
[INFO]      * No TLS CA certificate found
[INFO] 3.11 - Ensure that Docker server certificate file ownership is set to root:root
[INFO]      * No TLS Server certificate found
[INFO] 3.12 - Ensure that Docker server certificate file permissions are set to 444 or more
restrictive
[INFO]      * No TLS Server certificate found
[INFO] 3.13 - Ensure that Docker server certificate key file ownership is set to root:root
[INFO]      * No TLS Key found
[INFO] 3.14 - Ensure that Docker server certificate key file permissions are set to 400
[INFO]      * No TLS Key found
[PASS] 3.15 - Ensure that Docker socket file ownership is set to root:docker
[PASS] 3.16 - Ensure that Docker socket file permissions are set to 660 or more restrictive
[INFO] 3.17 - Ensure that daemon.json file ownership is set to root:root
[INFO]      * File not found
[INFO] 3.18 - Ensure that daemon.json file permissions are set to 644 or more restrictive
[INFO]      * File not found
[PASS] 3.19 - Ensure that /etc/default/docker file ownership is set to root:root
```

```
[PASS] 3.20 - Ensure that /etc/default/docker file permissions are set to 644 or more
restrictive


[INFO] 4 - Container Images and Build File
[WARN] 4.1  - Ensure a user for the container has been created
[WARN]      * Running as root: bridge_bridge_run_5
[WARN]      * Running as root: bridge_bridge_run_4
[WARN]      * Running as root: bridge_bridge_run_3
[WARN]      * Running as root: bridge_bridge_run_2
[WARN]      * Running as root: bridge_bridge_run_1
[WARN]      * Running as root: bridge_redis_1
[WARN]      * Running as root: bridge_rabbit_1
[NOTE] 4.2  - Ensure that containers use trusted base images
[NOTE] 4.3  - Ensure unnecessary packages are not installed in the container
[NOTE] 4.4  - Ensure images are scanned and rebuilt to include security patches
[WARN] 4.5  - Ensure Content trust for Docker is Enabled
[WARN] 4.6  - Ensure HEALTHCHECK instructions have been added to the container image
[WARN]      * No Healthcheck found: [bridge_bridge:latest]
[WARN]      * No Healthcheck found: [redis:latest]
[WARN]      * No Healthcheck found: [rabbitmq:3-management]
[WARN]      * No Healthcheck found: [node:8]
[INFO] 4.7  - Ensure update instructions are not use alone in the Dockerfile
[INFO]      * Update instruction found: [bridge_bridge:latest]
[INFO]      * Update instruction found: [rabbitmq:3-management]
[INFO]      * Update instruction found: [node:8]
[NOTE] 4.8  - Ensure setuid and setgid permissions are removed in the images
[INFO] 4.9  - Ensure COPY is used instead of ADD in Dockerfile
[INFO]      * ADD in image history: [bridge_bridge:latest]
[INFO]      * ADD in image history: [redis:latest]
[INFO]      * ADD in image history: [rabbitmq:3-management]
[INFO]      * ADD in image history: [node:8]
```

```
[NOTE] 4.10 - Ensure secrets are not stored in Dockerfiles
[NOTE] 4.11 - Ensure verified packages are only Installed


[INFO] 5  - Container Runtime
[PASS] 5.1  - Ensure AppArmor Profile is Enabled
[WARN] 5.2  - Ensure SELinux security options are set, if applicable
[WARN]      * No SecurityOptions Found: bridge_bridge_run_5
[WARN]      * No SecurityOptions Found: bridge_bridge_run_4
[WARN]      * No SecurityOptions Found: bridge_bridge_run_3
[WARN]      * No SecurityOptions Found: bridge_bridge_run_2
[WARN]      * No SecurityOptions Found: bridge_bridge_run_1
[WARN]      * No SecurityOptions Found: bridge_redis_1
[WARN]      * No SecurityOptions Found: bridge_rabbit_1
[PASS] 5.3  - Ensure Linux Kernel Capabilities are restricted within containers
[PASS] 5.4  - Ensure privileged containers are not used
[PASS] 5.5  - Ensure sensitive host system directories are not mounted on containers
[PASS] 5.6  - Ensure ssh is not run within containers
[PASS] 5.7  - Ensure privileged ports are not mapped within containers
[NOTE] 5.8  - Ensure only needed ports are open on the container
[PASS] 5.9  - Ensure the host’s network namespace is not shared
[WARN] 5.10 - Ensure memory usage for container is limited
[WARN]      * Container running without memory restrictions: bridge_bridge_run_5
[WARN]      * Container running without memory restrictions: bridge_bridge_run_4
[WARN]      * Container running without memory restrictions: bridge_bridge_run_3
[WARN]      * Container running without memory restrictions: bridge_bridge_run_2
[WARN]      * Container running without memory restrictions: bridge_bridge_run_1
[WARN]      * Container running without memory restrictions: bridge_redis_1
[WARN]      * Container running without memory restrictions: bridge_rabbit_1
[WARN] 5.11 - Ensure CPU priority is set appropriately on the container
[WARN]      * Container running without CPU restrictions: bridge_bridge_run_5
[WARN]      * Container running without CPU restrictions: bridge_bridge_run_4
```

```
[WARN]      * Container running without CPU restrictions: bridge_bridge_run_3
[WARN]      * Container running without CPU restrictions: bridge_bridge_run_2
[WARN]      * Container running without CPU restrictions: bridge_bridge_run_1
[WARN]      * Container running without CPU restrictions: bridge_redis_1
[WARN]      * Container running without CPU restrictions: bridge_rabbit_1
[WARN] 5.12 - Ensure the container’s root filesystem is mounted as read only
[WARN]      * Container running with root FS mounted R/W: bridge_bridge_run_5
[WARN]      * Container running with root FS mounted R/W: bridge_bridge_run_4
[WARN]      * Container running with root FS mounted R/W: bridge_bridge_run_3
[WARN]      * Container running with root FS mounted R/W: bridge_bridge_run_2
[WARN]      * Container running with root FS mounted R/W: bridge_bridge_run_1
[WARN]      * Container running with root FS mounted R/W: bridge_redis_1
[WARN]      * Container running with root FS mounted R/W: bridge_rabbit_1
[PASS] 5.13 -  Ensure incoming container traffic is binded to a specific host interface
[WARN] 5.14 - Ensure ‘on-failure’ container restart policy is set to ‘5’
[WARN]      * MaximumRetryCount is not set to 5: bridge_bridge_run_5
[WARN]      * MaximumRetryCount is not set to 5: bridge_bridge_run_4
[WARN]      * MaximumRetryCount is not set to 5: bridge_bridge_run_3
[WARN]      * MaximumRetryCount is not set to 5: bridge_bridge_run_2
[WARN]      * MaximumRetryCount is not set to 5: bridge_bridge_run_1
[WARN]      * MaximumRetryCount is not set to 5: bridge_redis_1
[WARN]      * MaximumRetryCount is not set to 5: bridge_rabbit_1
[PASS] 5.15 - Ensure the host’s process namespace is not shared
[PASS] 5.16 - Ensure the host’s IPC namespace is not shared
[PASS] 5.17 - Ensure host devices are not directly exposed to containers
[INFO] 5.18 - Ensure the default ulimit is overwritten at runtime, only if needed
[INFO]      * Container no default ulimit override: bridge_bridge_run_5
[INFO]      * Container no default ulimit override: bridge_bridge_run_4
[INFO]      * Container no default ulimit override: bridge_bridge_run_3
[INFO]      * Container no default ulimit override: bridge_bridge_run_2
[INFO]      * Container no default ulimit override: bridge_bridge_run_1
[INFO]      * Container no default ulimit override: bridge_redis_1
```

```
[INFO]       * Container no default ulimit override: bridge_rabbit_1
[PASS] 5.19 - Ensure mount propagation mode is not set to shared
[PASS] 5.20 - Ensure the host’s UTS namespace is not shared
[PASS] 5.21 - Ensure the default seccomp profile is not Disabled
[NOTE] 5.22 - Ensure docker exec commands are not used with privileged option
[NOTE] 5.23 - Ensure docker exec commands are not used with user option
[PASS] 5.24 - Ensure cgroup usage is confirmed
[WARN] 5.25 - Ensure the container is restricted from acquiring additional privileges
[WARN]       * Privileges not restricted: bridge_bridge_run_5
[WARN]       * Privileges not restricted: bridge_bridge_run_4
[WARN]       * Privileges not restricted: bridge_bridge_run_3
[WARN]       * Privileges not restricted: bridge_bridge_run_2
[WARN]       * Privileges not restricted: bridge_bridge_run_1
[WARN]       * Privileges not restricted: bridge_redis_1
[WARN]       * Privileges not restricted: bridge_rabbit_1
[WARN] 5.26 - Ensure container health is checked at runtime
[WARN]       * Health check not set: bridge_bridge_run_5
[WARN]       * Health check not set: bridge_bridge_run_4
[WARN]       * Health check not set: bridge_bridge_run_3
[WARN]       * Health check not set: bridge_bridge_run_2
[WARN]       * Health check not set: bridge_bridge_run_1
[WARN]       * Health check not set: bridge_redis_1
[WARN]       * Health check not set: bridge_rabbit_1
[INFO] 5.27 - Ensure docker commands always get the latest version of the image
[WARN] 5.28 - Ensure PIDs cgroup limit is used
[WARN]       * PIDs limit not set: bridge_bridge_run_5
[WARN]       * PIDs limit not set: bridge_bridge_run_4
[WARN]       * PIDs limit not set: bridge_bridge_run_3
[WARN]       * PIDs limit not set: bridge_bridge_run_2
[WARN]       * PIDs limit not set: bridge_bridge_run_1
[WARN]       * PIDs limit not set: bridge_redis_1
[WARN]       * PIDs limit not set: bridge_rabbit_1
```

```
[PASS] 5.29 - Ensure Docker’s default bridge docker0 is not used
[PASS] 5.30 - Ensure the host’s user namespaces is not shared
[PASS] 5.31 - Ensure the Docker socket is not mounted inside any containers


[INFO] 6 - Docker Security Operations
[INFO] 6.1  - Avoid image sprawl
[INFO]      * There are currently: 4 images
[INFO] 6.2  - Avoid container sprawl
[INFO]      * There are currently a total of 14 containers, with 7 of them currently
running


[INFO] 7 - Docker Swarm Configuration
[PASS] 7.1  - Ensure swarm mode is not Enabled, if not needed
[PASS] 7.2  - Ensure the minimum number of manager nodes have been created in a swarm
(Swarm mode not enabled)
[PASS] 7.3  - Ensure swarm services are binded to a specific host interface (Swarm mode not
enabled)
[PASS] 7.5  - Ensure Docker’s secret management commands are used for managing secrets in a
Swarm cluster (Swarm mode not enabled)
[PASS] 7.6  - Ensure swarm manager is run in auto-lock mode (Swarm mode not enabled)
[PASS] 7.7  - Ensure swarm manager auto-lock key is rotated periodically (Swarm mode not
enabled)
[PASS] 7.8  - Ensure node certificates are rotated as appropriate (Swarm mode not enabled)
[PASS] 7.9  - Ensure CA certificates are rotated as appropriate (Swarm mode not enabled)
[PASS] 7.10 - Ensure management plane traffic has been separated from data plane traffic
(Swarm mode not enabled)

[INFO] Checks: 104
[INFO] Score: 17
```

About Us

Worried about the security of your project? You're on the right way! The second step is to find a team of seasoned cybersecurity experts who will make it impenetrable. And you've just come to the right place.

PepperSec is a group of whitehat hackers seasoned by many-year experience and have a deep understanding of the modern Internet technologies. We're ready to battle for the security of your project.

LET'S KEEP IN TOUCH

peppersec.com

hello@peppersec.com

Github