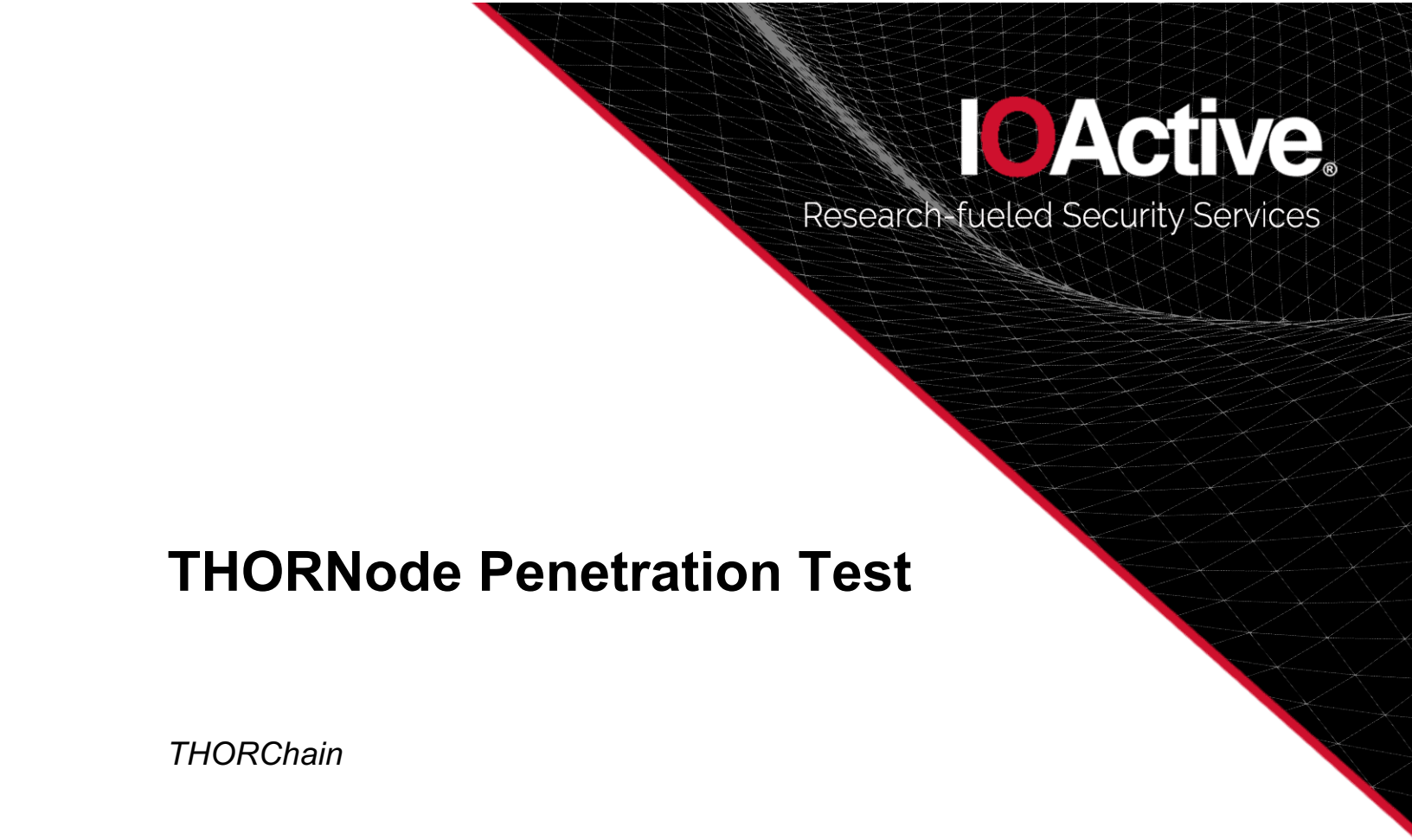
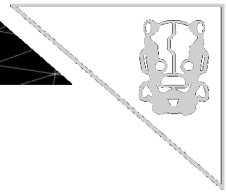


THORNode Penetration Test

THORChain

IOActive Europe Limited
Fifth Floor, 120 Charing Cross Road
London, WC2H 0JR
United Kingdom

UK Office: +44(0)20 7240 5223
UK Direct: +44(0)20 7836 8115



Document Management

Document Information

Client Name	THORChain
Project Name	THORNode Penetration Test
Project Start Date	2020-09-30
Project End Date	2020-10-14

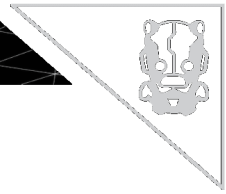
Document Revision Information

Date	Version	Author	Revision Details
2020-10-19	1.0	Matt Suiche	Initial Version
2020-10-19	1.1	IOActive	Tech Pubs QA
2020-10-19	1.2	Gary van Blerk	Peer Review
2020-10-21	1.3	Matt Suiche	Additions
2020-11-15	1.4	IOActive	Edits
2020-11-16	1.5	IOActive	Tech Pubs QA
2020-11-16	1.6	Scott Headington	Added Remediation Information
2020-11-16	Final	Scott Headington	Final report version

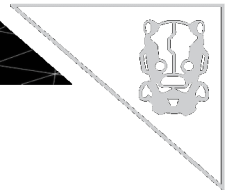
Project Contacts

IOActive Europe Limited

Name	Title	Contact Information
Matt Suiche	Security Consultant	matt.suiche@ioactive.com
Scott Headington	Managing Director of Services	scott.headington@ioactive.com
Melissa Van Hooser	Engagement Manager	melissa.vanhooser@ioactive.com



Contents



Executive Summary

THORChain engaged IOActive Europe Limited (IOActive) to assess the security threats and risks associated with its THORNode software.

THORChain is a liquidity protocol based on Tendermint and Cosmos-SDK and utilising Threshold Signature Schemes (TSS). THORNodes sync and run the network. Block explorers can connect to a THORNode to display the underlying blockchain data. THORNodes can either be validators or light nodes that simply sync and display data, which allows a node to join and service the THORChain network.

Project Description

One IOActive consultant assessed the THORNode software from the 30th of September to the 14th of October 2020. The goal of this assessment was to determine if the private keys housed on a node could be compromised by an attacker.

The consultant reviewed the following in-scope assets:

- Terraform scripts for deploying a THORNode cluster
- Helm Charts for deploying and managing a THORNode cluster
- The run-time configuration and attack surface of a local THORNode instance deployed by the consultant
- A canonical THORNode deployment provided by THORChain

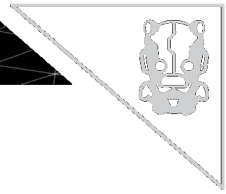
The consultant observed the following major strengths:

- Solid documentation
- The design based on Cosmos-SDK and Tendermint
- Ease of deployment

IOActive was not successful in compromising any of the keys during the assessment; however, it is our opinion that a motivated attacker with more time and resources may be able to achieve this goal by focusing on one of three attack vectors as described in the Technical Summary section.

Next Steps

IOActive strongly recommends addressing the attack vectors presented in this report to improve the security posture of the THORNode software. Once THORChain has addressed the findings, IOActive further recommends performing remediation validation testing to confirm they are properly fixed.



Technical Summary

Project Scope

IOActive's assessment supported THORChain's stated goal of determining if one or more of the following private keys could be compromised on a THORNode:

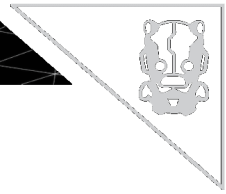
- Cosmos key
- Yggdrasil encrypted private key for funds
- TSS shared key used for signing transactions

The consultant conducted a grey-box security assessment, including analysing the infrastructure-as-code (Terraform, Helm), reviewing traffic analysis, and other testing techniques. The consultant:

- Reviewed the Terraform scripts for deploying a THORNode cluster: `https://gitlab.com/thorchain/devops/cluster-launcher`
- Reviewed the Helm Charts for deploying and managing THORNode: `https://gitlab.com/thorchain/devops/node-launcher`
- Analysed the run-time configuration and attack surface of a local THORNode instance deployed by the consultant
- Completed an infrastructure penetration test against a canonical THORNode deployment provided by THORChain

The following were out of scope for this engagement:

- An extensive review of either the THORNode or Cosmos codebase
- Reviewing the hosting or cloud-provider configuration and infrastructure (e.g. AWS security group configuration, S3 bucket security, etc.)



Project Approach

IOActive selected attack vectors based on the ATT&CK-like threat matrix for Kubernetes designed by the Microsoft Azure Security Center[1] team and shown in Figure 1.

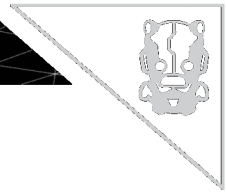
Initial Access	Execution	Persistence	Privilege Escalation	Defense Evasion	Credential Access	Discovery	Lateral Movement	Impact
Using Cloud credentials	Exec into container	Backdoor container	Privileged container	Clear container logs	List K8S secrets	Access the K8S API server	Access cloud resources	Data Destruction
Compromised images in registry	bash/cmd inside container	Writable hostPath mount	Cluster-admin binding	Delete K8S events	Mount service principal	Access Kubelet API	Container service account	Resource Hijacking
Kubeconfig file	New container	Kubernetes CronJob	hostPath mount	Pod / container name similarity	Access container service account	Network mapping	Cluster internal networking	Denial of service
Application vulnerability	Application exploit (RCE)		Access cloud resources	Connect from Proxy server	Applications credentials in configuration files	Access Kubernetes dashboard	Applications credentials in configuration files	
Exposed Dashboard	SSH server running inside container					Instance Metadata API	Writable volume mounts on the host	
							Access Kubernetes dashboard	
							Access tiller endpoint	

Figure 1. Attack Matrix

For this assignment, IOActive only focused on initial tactics for compromise (listed under "Initial Access"), where an attacker focuses on gaining access to a resource in containerized environments. There are five main techniques which can be leveraged by an attacker to gain access:

- Cloud credentials
- Compromised images in the registry
- The Keubeconfig file
- Exploiting an application vulnerability
- Using an exposed dashboard

[1] https://www.microsoft.com/security/blog/2020/04/02/attack-matrix-kubernetes/



Attack Vectors

Cloud Credentials

THORChain provides two cloud providers options: AWS and DigitalOcean. Compromised cloud credentials can lead to a Kubernetes cluster takeover.

An attacker could attempt to steal credentials directly from the node operator team. This almost happened to Coinbase employees in 2019 when they were targeted by two Firefox 0-day exploits via phishing emails.[2] We recommend node operators conduct internal training with their employees to raise awareness of spear-phishing attacks and impersonation over social media, which could be leveraged to target specific employees.

IOActive recommends that all the cloud operators have two-factor authentication (2FA) enabled for accessing the cloud provider hosting the cluster, be vigilant against such attacks, and provide phishing training to cloud operators.

Compromised Images (Supply-chain Attacks)

Running a compromised image in a cluster can compromise the cluster. Attackers who gain access to a private registry can plant their own compromised images in the registry. The latter can then be pulled by a user. In addition, users often use untrusted images from public registries that may be malicious.

Building images based on untrusted base images can also lead to similar results.

Supply chain attacks have been on the rise against open-source projects. Although Node.js (npm) and Python (PyPI) repositories have been the most commonly targeted, it is possible for Go applications and repositories to also be targeted. Repositories managed by individuals should be considered high-risk, for instance, the two following packages:

- The AWS terraform script `cluster-launcher/aws/provider.tf` relies on a package that is maintained by an individual `gavinbunney/kubectl`.
- The bitcoin-daemon `node-launcher/bitcoin-daemon/values.yaml` relies on a packaged maintained by an individual `ruimarinho/bitcoin-core`.

IOActive recommends that everyone with write access to `registry.gitlab.com/thorchain` have 2FA enabled.

[2] https://blog.coinbase.com/responding-to-firefox-0-days-in-the-wild-d9c85a57f15b

Remediation Note: As of 11/16/20 IOActive validated that THORChain remediated the risk from the two above noted packages by removing them from the code base and making the appropriate replacements where applicable. This can be shown in the following GitHub commits:

https://gitlab.com/thorchain/devops/node-launcher/-/merge_requests/91

https://gitlab.com/thorchain/devops/node-launcher/-/merge_requests/21

Kubeconfig File

The Kubernetes configuration (kubeconfig) file, contains details about Kubernetes clusters including their location and credentials.

Similar to stealing cloud credentials, node operators could be the target of elaborate attacks aimed at gaining access to their machines. This would give an attacker access to the Kubernetes configuration file in order to gain access to the clusters.

If an attacker managed to compromised the cloud credentials, they could access this configuration file.

IOActive recommends node operators install and maintain anti-malware protection on their systems, and be vigilant against insider threats, where another malicious user accesses the node operator's machine and steals this file.

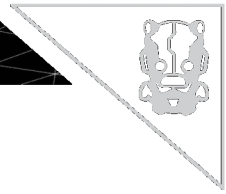
Application Vulnerability

Running a public-facing vulnerable application in a cluster can enable initial access to the cluster. A container that runs an application that is vulnerable to remote code execution (RCE) vulnerability could be exploited.

Kubernetes nodes could be compromised directly by exploiting a Linux RCE vulnerability. An attacker who achieved RCE could access the content of the Docker containers and install a rootkit, similar to the Drovorub malware recently reported by the NSA/FBI.[3] Once the node is compromised, the attacker could copy the files under `~/.thord/*` and `~/.thorcli/*`.

IOActive recommends continuously checking for and installing the latest version of vendor-supplied software. We also recommend running Linux Kernel 3.7 or later in order to take full advantage of kernel signing enforcement to prevent the loading of unsigned kernel modules by attackers.

[3] https://www.fbi.gov/news/pressrel/press-releases/nsa-and-fbi-expose-russian-previously-undisclosed-malware-drovorub-in-cybersecurity-advisory



If possible, UEFI Secure Boot should be enabled; however, this may be challenging on AWS. To date, only Google Cloud Platform provides access to this feature with shielded VMs.

Another consideration for more privacy when using the secrets/keys would be to use SGX-type enclaves. This would prevent attackers who compromise the system from reading the memory of certain applications. AWS came up with its own model called AWS Nitro Enclaves,[4] and Azure announced that it will provide early access to that feature on October 14, 2020.[5]

Exposed Dashboard

The Kubernetes dashboard is a web-based user interface that enables monitoring and managing a Kubernetes cluster.

IOActive recommends never publicly exposing this dashboard.

[4] https://aws.amazon.com/ec2/nitro/nitro-enclaves/

[5] https://azure.microsoft.com/en-us/blog/azure-and-intel-commit-to-delivering-next-generation-confidential-computing/