LONGTERM

PrimaBlock Security Assessment

July, 2018

Overview

During July, 2018 Longterm Security, Inc. reviewed PrimaBlock's technology stack and software practices to discover security vulnerabilities. Testing encompassed the smart contracts, the frontend, backend software services, and other infrastructure. We are pleased to endorse PrimaBlock's security practices and software security quality.

Best Practices

Attack Model

We see two areas of threats to PrimaBlock. The first area of threat comes down to attacks against the users of PrimaBlock, both against pool creators as well as funders. Scammers and fraudsters may try to impersonate organizations to fund fraudulent pools. In their tactics, attackers could attempt to abuse social media, social media integrations, or attempt UI redressing attacks through flaws in PrimaBlock's website. A more devastating attack would be scamsters using XSS to trick users into making malicious transfers.

Best Practice Recommendations

Key Threat Vectors for Threat Type #1 (Attacks Against Customers):

Social Engineering/Phishing/Scammers

The scammer issue is largely one that lies outside of PrimaBlock's technology and has to do with user's awareness and care as much as anything else. The new social media integrations do provide a new avenue for impersonating others, but they also provide ways for creators to securely verify their accounts on social media platforms such as Telegram, Twitter, and LinkedIn which helps other users check the veracity of pools.

Continue to:

- Provide two-factor authentication to users. This is a strong defense against account hijacks.
- Educate users through the website, FAQ, and educational videos on the youtube channel to keep users aware of risks and scammer tactics.
- Avoid including third-party scripts and resources on the website which could be used for UI redressing
    - Limit the amount of dynamic HTML from the blog to confined areas on the website
- Provide a way to cancel and shut down scam pools in progress on the platform.
- Use DKIM or SPF policies on the primablock.com domain to prevent e-mail phishing

XSS

The best mitigation against Cross-Site-Scripting is to apply a good Content Security Policy. With Angular2 in practice, the strongest CSP protections of disallowing inline scripting may not be practical. On the other hand, Angular2, when used correctly, makes XSS very unlikely barring an Angular2 Sanitizer/Browser flaw. Currently PrimaBlock observes best practices here.

Continue to:

- Avoid using **bypassSecurityTrustHtml** / **bypassSecurityTrustScript** angular helpers
- Avoid generating Angular Templates, Raw HTML/web content server-side to display to users on the primablock domain
- Validate content types for user-generated uploads to prevent unexpected XSS from certain content types, such as Flash applets and SVG images.

CSRF

Another common client-side attack is Cross-Site-Request-Forgery. Currently PrimaBlock eliminates this risk by using the Gorilla CSRF package to protect the API routes in accounts. For the presale pool websocket service, an origin check prevents CSRF issues there.

Continue to:

- Check the Origin for WebSocket Services
- Use Gorilla CSRF for Go's route handling for APIs

- Never have stateful effects in GET requests, limit them to POST/PUT/DELETE requests.

Potential Areas for Improvement:

- Verify new APIs have adequate CSRF protection.

Key Threat Vectors for Threat Type #2 (Attacks Against PrimaBlock):

The second threat area we consider is about attacks against PrimaBlock's infrastructure and staff. Even when using technology secure from vulnerabilities, social engineering and other tricks can be used to gain entry into an organization. In the cryptocurrency space, attacks against staff and infrastructure can be especially devastating if attackers gain access to credentials for financial cryptocurrency transactions.

Social Engineering/Phishing

Someone could attempt to break into PrimaBlock infrastructure without any technological payloads, but by impersonation and other phishing tricks. For example, an attacker could attempt to gain entry into an account for a third-party email provider or dns-host.

Continue to:

- Require two-factor authentication for all infrastructure access
- Harden G-Suite settings
- Use password managers

Advanced, Targeted Attacks against PrimaBlock and the Ethereum Community

As an extension to the above, consider the risk to PrimaBlock against a well-funded adversary. If PrimaBlock staff is sent a malicious link with this exploit, or a common ethereum developer page is targeted with such an exploit, consider the fallout and how to defend against it.

Continue to

- Isolate critical infrastructure credentials from day-to-day operations and especially from personal equipment.
- Isolate server-side components further so that if an attacker gains access to one system they don't have the keys to the kingdom

Areas For Improvement:

- About twice an year, spend an afternoon going through a simulated incident response scenario. The simulation can be a pen and paper exercise where the DevOps team tries to uncover a hypothetical attack. Throughout the process it will engage critical thinking viewpoints that will uncover potential areas that need better security controls and monitoring.

Weaknesses in Backend Services

We continue to see great coding practices and sane design decisions that prevent many common flaws and issues.

Continue to

- Sanitize inputs and use validators wherever possible for structured data
- Use cryptographically secure identifiers and codes that are not guessable or enumerable
- Use standard system libraries for parsing inputs
- Use parameterized statements to prevent SQL injection

Backdoors and Flaws With Third-Party Software

Third-party NPM dependencies could be targeted to universally attack ethereum developers, affecting PrimaBlock also. The NPM community is starting to make improvements in this area with signing, however this is still a notable risk for the Ethereum DApp community.

Continue to

- Avoid using docker containers from unverified parties
- Keep dependencies up to date to get security patches

Smart Contract Flaws

After spending extensive amounts of time auditing the smart contracts we are confident that they are largely immune to vulnerabilities. We are pleased to see the use of SafeMath to eliminate integer handling flaws.

Continue to

- Use SafeMath
- Perform code reviews when introducing changes, even if minor

Vulnerability Tracking Document

Findings Guide

Impact Rating	Explanation	Examples
Critical	A catastrophically severe vulnerability with a demonstrable exploitation path, requiring little or no interaction from the victim nor special privileges to exploit, leading to unauthorized code execution or data access.	Pre-authenticated remote code execution, SQL Injection from public APIs
High	A severe vulnerability, possibly requiring some user interaction from the victim or special privileges to exploit, leading to unauthorized code execution or data access.	Reflected Cross Site Scripting (XSS), CSRF, use-after-free, memory trespass errors
Medium	An important vulnerability, allowing some unauthorized behavior or violating some security assumption, that could lead to partial data access without authorization, or social engineering attacks	Uninitialized memory disclosure, clickjacking
Low	A vulnerability that may not be feasible to exploit in practice or would provide little benefit to real attackers, or a vulnerability that is not useful to attackers without other contributing factors	Information leaks about software versions
Informational	These can include best practice recommendations, security hardening tips, or additional information how a system works	Missing CSP Headers, Compiler Security Flags, a failed exploitation attempt

Findings Summary

Issue	Severity	Status	Title
LTS-1	Informational	Fixed	Potentially dangerous use of request-promise-native
LTS-2	Informational	Fixed	Usage of sendgrid links for official emails
LTS-3	Medium	Fixed	Exposed portainer instance
LTS-4	Medium	Fixed	Twitter OAuth1 Profile CSRF

LTS-1 Potentially dangerous use of request-promise-native

Informational

Classification

CWE-918

Description

The code in presale-pool-frontend/server contains many references to request-promise-native which is a HTTP request library. This is used by a number of API endpoints to make requests to the internal http://${config.accountsService}/ server.

While we did not find an exploitable SSRF issue in the code using this library, the code pattern that is used might create an exploitable SSRF issue in the future when new functionality is added.

Remediation

Employ a whitelist filter on allowed characters. There is currently no filtering which allows dangerous characters such as ("/", ".", "%") to be passed on to internal services.

Reproduction steps, additional notes

N/A

LTS-2 Usage of sendgrid links for official emails

Informational

Classification

N/A

Description

When confirming an account or requesting a password reset, the confirmation links will go through sendgrid and look like the following:

https://u7863111.ct.sendgrid.net/wf/click?upn=ghRy6-2FoHI....

When clicking this link, the sendgrid.net host will respond with a HTTP 302 redirect:

```
< HTTP/1.1 302 Found
< Server: nginx
< Date: Mon, 30 Jul 2018 16:07:09 GMT
< Content-Type: text/html; charset=utf-8
< Transfer-Encoding: chunked
< Connection: keep-alive
< Location: https://rinkeby.primablock.com/forgot#eyJhbGciOiJSUzU...
```

There are two risks associated with using this third-party service. The first risk is that it opens up the possibility for attackers to create legitimate-looking phishing emails since they can create their own sendgrid account and create links at will. This makes it harder to detect a phishing attempt at first glance.

The second risk is that letting sendgrid handle emails for forgotten passwords means that they are trusted to reset the password for any account. A determined attacker could find the email address for a pool administrator, compromise sendgrid, reset the password for the targeted user and then intercept the password reset link.

Remediation

If the forgotten password links would instead link to primablock.com directly, it would be easier to detect phishing attempts.

Consider not using a third-party service for password reset emails.

Reproduction steps, additional notes

Accounts with 2FA enabled are safe against the reset attack since a 2FA code is still required to login after resetting the password.

LTS-3 Exposed portainer instance

Medium

Classification

CWE-183

Description

A vulnerability was addressed in June which would allow executing arbitrary code in docker containers, without proper authentication: CVE-2018-12678

“Portainer before 1.18.0 supports unauthenticated requests to the websocket endpoint with an unvalidated id query parameter for the /websocket/exec endpoint, which allows remote attackers to bypass intended access restrictions or conduct SSRF attacks.
“

Mitigating Factors

It is likely that to actually run commands on the containers, an attacker would also need to know a docker container id. It is improbable for a remote attacker to guess these ids. We have not investigated portainer further.

This is a testing server, so the impact on production was also likely limited.

Remediation

If this service is not needed it should be removed. Portainer should be updated and access to it should be restricted to a SSO solution such as Google IAP or behind a VPN such was Wireguard.

Reproduction steps, additional notes

N/A

LTS-4 Twitter OAuth 1 Profile CSRF

Medium

Classification

CWE-287

Description

The Twitter OAuth 1.0 integration was susceptible to a hijacking vulnerability. A malicious attacker could start an OAuth verification with their twitter account, and then send the link to a victim signed into PrimaBlock. The victim would then unintentionally have their PrimaBlock profile linked with the attacker’s profile.

Mitigating Factors

This is a social engineering attack that requires a victim to be signed into PrimaBlock and to not have a twitter account linked to their profile already.

Remediation

This issue was remediated during the course of the engagement by adding additional state to the OAuth 1.0 negotiation

Reproduction steps, additional notes

N/A