Fine penetration tests for fine websites

Dr.-Ing. Mario Heiderich, Cure53
Wilmersdorfer Str. 106
D 10629 Berlin
cure53.de · mario@cure53.de

Pentest-Report Psiphon Conduit Integration Codebase 04-05.2024

Cure53, Dr.-Ing. M. Heiderich, MSc. H. Moesl-Canaval, P. Einkemmer

Index

Dr.-Ing. Mario Heiderich, Cure53
Wilmersdorfer Str. 106
D 10629 Berlin
cure53.de · mario@cure53.de

Introduction

"Psiphon Inc. is a company based in Toronto, producing open-source multi-platform software that helps over 3 million people every week connect to content on the Internet. We're a team focused on delivering the best software we can, introducing new products regularly and making sure we develop to the needs of our constantly growing global audience."

From https://www.psiphon.ca/en/about.html

This report (identifiable as the acronym PSI-07) outlines the results of a penetration test and source code audit against the integration of the Psiphon Conduit feature utilized in the Psiphon core codebase.

Psiphon Inc. contacted Cure53 with the initial proposal in January 2024, stipulating the overall requirements and goals. Once the finer details and budget (eleven work days) had been agreed upon, the evaluation itself was scheduled for late April and early May 2024, specifically CW18 through CW19. The work was structured into a single Work Package (WP), defined as *WP1: White-box pentests & audits against Psiphon Conduit Integration codebase*. Notably, Cure53 has been tasked with assessing the Psiphon Conduit library on one previous occasion, as it was included in the scope of an assignment held in June 2023 (see *PSI-06*).

A white-box pentesting methodology was selected for this exercise, hence full source code access was provided, as well test-supporting documentation and other miscellaneous facets. A team of three senior testers, selected for their proficiency with auditing features of this nature, were assigned to the assignment's preparation, execution, and finalization stages.

All preparatory initiatives were completed in advance for a seamless start, specifically in CW17 April 2024. For communications, a dedicated and shared Slack channel was established to combine all active personnel from Psiphon and Cure53 into a collaborative forum. Discussions and queries were fairly minimal, the scope benefited from comprehensive preparation, and no noteworthy roadblocks were encountered during the test. Periodic status updates were provided to keep the customer informed regarding the general test progress and interesting discoveries. Live reporting was also conducted using the aforementioned Slack channel.

With regards to the findings, Cure53's decent coverage over the scope items only yielded a single finding classified as a common fault with lower exploitation potential. The overall lack of vulnerabilities attests to the performant defensive capabilities of the Psiphon Conduit feature.

Dr.-Ing. Mario Heiderich, Cure53
Wilmersdorfer Str. 106
D 10629 Berlin
cure53.de · mario@cure53.de

However, one must emphasize that the inspected feature proved compositionally complex. Due to the limited documentation that was primarily accessible via source code comments, the test team needed additional time to familiarize with the infrastructure landscape, henceforth reflecting the effectiveness of the security measures implemented by the Psiphon team.

To caveat this, the complexity of the Psiphon Conduit integration's software, architecture, and components necessitates recurring in-depth security assessments. Frequent changes can introduce unintended ramifications and systemic security issues, while broadly scoped audits oftentimes lack the focused depth needed to unearth embedded flaws. Moving forward, Cure53 recommends performing regular and targeted evaluations to proactively identify and address all plausible vulnerabilities and threats.

The report will now shed more light on the scope and testing setup as well as provide a comprehensive breakdown of the available materials. This will be followed by a chapter outlining the test methodology, which serves to provide greater clarity on the techniques applied and coverage achieved throughout this audit. Subsequently, the report will list all findings identified in chronological order, starting with the detected vulnerabilities (albeit none were found) and followed by the general weaknesses unearthed. Each finding will be accompanied by a technical description and Proof of Concepts (PoCs) where applicable, plus any relevant mitigatory or preventative advice to action.

In summation, the report will finalize with a conclusion in which the Cure53 team will elaborate on the impressions gained toward the general security posture of the attributes in focus, giving high-level hardening advice where applicable.

Dr.-Ing. Mario Heiderich, Cure53
Wilmersdorfer Str. 106
D 10629 Berlin
cure53.de · mario@cure53.de

Scope

- **Pentests & audits against Psiphon Conduit Integration codebase**
  - **WP1:** White-box pentests & audits against Psiphon Conduit Integration codebase
    - **Source code:**
      - URL:
        - https://github.com/Psiphon-Labs/psiphon-tunnel-core/tree/inproxy
      - Integration commit:
        - https://github.com/Psiphon-Labs/psiphon-tunnel-core/commit/53dab5608a42c9e0d569e1f2d0f526b8b9885379
    - **Documentation:**
      - https://psiphon.slack.com/archives/C0166KQR8G5/p1685540595300579
      - https://github.com/Psiphon-Labs/psiphon-tunnel-core?tab=readme-ov-file#technical-summary
  - **Test-supporting material was shared with Cure53**
  - **All relevant sources were shared with Cure53**

Dr.-Ing. Mario Heiderich, Cure53
Wilmersdorfer Str. 106
D 10629 Berlin
cure53.de · mario@cure53.de

Test Methodology

This section documents the testing methodology applied by Cure53 during this project and discusses the resulting coverage, shedding light on how various components were examined. Further clarification concerning areas of investigation subjected to deep-dive assessment is offered, considering the absence of any security vulnerabilities and only one miscellaneous issue detected.

Psiphon's inproxy/Conduit library has already been reviewed by Cure53 during a previous assessment performed back in June 2023. The inproxy enables third-party, ephemeral proxies in order to aid Psiphon client connections to the Psiphon network. The feature itself is inspired by Tor's Snowflake pluggable transport.

The Psiphon network comprises clients within censored regions, brokers, proxies, and Psiphon servers. Clients offer default (but updatable) broker connection settings. The purpose of the broker is to locate the optimal proxy for a client based on compartment IDs and their geolocation. Clients communicate with proxies using WebRTC DataChannels over DTLs, while interactions with brokers leverage domain fronting techniques that encapsulate a Noise protocol session. Proxies serve to relay either TCP or UDP flows from the client to the server, which are contained within Psiphon tunnel protocols. Once a client tunnel to a Psiphon server is established, brokers send secure session packets to the server via the client. These packets are layered on top of client-broker and client-server communications, effectively concealing any direct traffic from brokers to the servers.

This complex landscape and the limited corresponding documentation (available only via the source code and comments) necessitated extended familiarization time for the Cure53 team. This should not be interpreted as a justification for the general lack of vulnerabilities detected. Rather, it reflects the effective hardening and security measures implemented by the Psiphon team in the Psiphon Conduit integration.

The passages below offer insight regarding the key objectives and degree of coverage for the technical procedures:

- The testers commenced proceeding with a review of the prior pentest report (PSI-06), reassessing all previously identified issues related to the Conduit library. All but one were fix-verified by Cure53, the outlier being *PSI-06-004*, which is related to DPI via randomized Hellos in DTLs. Thus, the audit team inspected the latest commit[1] within the inproxy branch of the psiphon-tunnel-core GitHub repository to verify whether the modifications implemented since the last audit contain a fix for the aforementioned outstanding issue. This investigation confirmed that the flaw had indeed been resolved.

[1] https://github.com/Psiphon-Labs/psiphon-tunnel-core/commit/da8f91026e020abe597859dd6[...]ba7b

Dr.-Ing. Mario Heiderich, Cure53
Wilmersdorfer Str. 106
D 10629 Berlin
cure53.de · mario@cure53.de

- This security audit's core focus was placed on the integration of the already audited inproxy feature into the Psiphon core codebase. For this purpose, the testing team performed a thorough inspection of the commit[2], which was related to the inproxy integration into *psiphon-tunnel-core*. This commit significantly altered the codebase, modifying 96 source files with approximately 13,000 additions and 4,000 deletions.
- To evaluate all updates applied to the inproxy feature since the last audit, a comparison was made between the integration commit previously reviewed and the most recent iteration. This examination revealed that the inproxy-related code underwent minor refactoring, with the majority of changes considered non-security-relevant.
- Moreover, the customer also shared builds of the *psiphon-tunnel-core* to dynamically test the inproxy feature in tandem with respective configuration files for running a client or proxy. Initial attempts using the *psiphon-tunnel-core* binary/CLI and the provided configuration files were not successful and resulted in an error. Following consultation with the customer, a new build of the *psiphon-tunnel-core* binary was provided, which operated as expected and paved the way for a successful tunnel connection. For this configuration to function, the new binary using the provided proxy configuration had to be executed within another network than the alternative using the provided client configuration.
- The Psiphon application was vetted for outdated and vulnerable Golang dependencies using tools such as *gosec*[3] and *govulncheck*[4]. However, no issues were identified in this area and all dependencies were updated to the latest possible versions.
- Next, Cure53 concentrated on the new *InproxyBrokerClientManager* component and associated elements. This manager is integral since it facilitates the relay system that enables dials to brokers. Central to this system is the creation of a *NewInproxyBrokerClientInstance*, which creates a link between a broker client and its specific dial parameters. These parameters ensure successful connections to a Psiphon server from within censored regions and hence play a critical role in the security architecture. The configurations, which include information such as compartment IDs, broker specifications, domain fronting settings, and other parameters, are designed to bypass censorship mechanisms. To ascertain the security efficacy of these settings, the testing team rigorously investigated the plausibility of altering the configurations via unauthorized or malicious actors at runtime. However, these activities yielded no vulnerable pathways.
- On both the proxy- and client-side, the same *InproxyBrokerClientManager* instance handles multiple clients and extensively leverages locking mechanisms. As proxies are fundamental to the Psiphon network, the potential for Denial of Service (DoS) attacks by malicious clients on a proxy was examined, though this aspect was deemed risk averse.

[2] https://github.com/Psiphon-Labs/psiphon-tunnel-core/commit/53dab5608a42c9e0d569e1f2d[...]85379
[3] https://github.com/securego/gosec
[4] https://go.dev/blog/govulncheck

Dr.-Ing. Mario Heiderich, Cure53
Wilmersdorfer Str. 106
D 10629 Berlin
cure53.de · mario@cure53.de

- The configuration of WebRTC connections from clients to proxies was extensively appraised. Various features, including NAT discovery, STUN, and port mapping, were found to be activated or deactivated based on configured probabilities, demonstrating diverse client behaviors. The testers experimented with different settings, including DTLS randomization and traffic shaping, though no erroneous behaviors were witnessed.
- Ample testing capacity was also granted to the broker feature enhancements. The meek server component of Psiphon was specifically adapted to enable configuration of a broker based on geographic IP location and tactical data. This adaptation was designed to optimize connectivity strategies by leveraging location-specific information to circumvent regional censorship barriers. The enhancements observed in this area during the audit primarily included the capability to fetch tactical configurations based on the geographic IP data of clients and proxies, alongside certain modifications to the standard Psiphon handshake procedure to accommodate these alterations. Despite diligent efforts, no detrimental security practices were observed.
- Lastly, the audit team positively acknowledged that a signed reset session token was integrated into the Psiphon server, which assures session expiry and hence neutralizes potential replay attacks. The implementation and cryptographic construct of the signed reset token were also systematically analyzed, which was verified as sufficiently secured upon inspection.

Dr.-Ing. Mario Heiderich, Cure53
Wilmersdorfer Str. 106
D 10629 Berlin
cure53.de · mario@cure53.de

Miscellaneous Issues

This section covers any and all noteworthy findings that did not incur an exploit but may assist an attacker in successfully achieving malicious objectives in the future. Most of these results are vulnerable code snippets that did not provide an easy method by which to be called. Conclusively, while a vulnerability is present, an exploit may not always be possible.

PSI-07-001 WP1: Constant WebRTC DataChannel label *(Info)*

***Note**: The issue was fixed and the fix was verified by Cure53 who had access to the diff.*

While reviewing Psiphon's inproxy feature, the audit team confirmed that both the clients and proxies communicate using the WebRTC protocol via its data channels. WebRTC utilizes multiple obfuscation techniques to enhance security, such as randomizing the DTLS Client/ServerHello and shaping traffic. This involves integrating random padding and decoy messages to the data channel flows.

This approach lays a solid groundwork for obfuscation. However, the initialization of the WebRTC data channel includes a constant label, which is consistently transmitted within the packets of the data channel. This persistent labeling could potentially represent a weak point in the communication obfuscation procedure, since pattern recognition and tracking may be facilitated.

Affected file:
psiphon-tunnel-core/psiphon/common/inproxy/webrtc.go

Affected code:

```
func newWebRTCConn(...) {
	[...]

	if isOffer {

		dataChannelInit := &webrtc.DataChannelInit{}
		if !config.ReliableTransport {
			ordered := false
			dataChannelInit.Ordered = &ordered
			maxRetransmits := uint16(0)
			dataChannelInit.MaxRetransmits = &maxRetransmits
		}

		// TODO: randomize length?
		dataChannelLabel := "in-proxy-data-channel"

		dataChannel, err := peerConnection.CreateDataChannel(
			dataChannelLabel, dataChannelInit)
```

Dr.-Ing. Mario Heiderich, Cure53
Wilmersdorfer Str. 106
D 10629 Berlin
cure53.de · mario@cure53.de

```
                if err != nil {
                        return nil, nil, nil, errors.Trace(err)
                }

                conn.setDataChannel(dataChannel)
        }

        [...]
}
```

Although this situation does not immediately jeopardize the detection of the obfuscation measures, Cure53 recommends altering the current approach and additionally using dynamic labels. This would further enhance security and decrease the predictability of the communication patterns, therefore complicating censorship efforts.

Dr.-Ing. Mario Heiderich, Cure53
Wilmersdorfer Str. 106
D 10629 Berlin
cure53.de · mario@cure53.de

Conclusions

In context, Cure53 conducted an in-depth code audit of the Conduit feature with a team of eight senior pentesters back in 2023, while this current Q4 2024 test iteration was handled by three consultants and targeted the integration of the Conduit feature into the Psiphon core codebase.

Cross-organizational communications were frequent, as enabled through the use of a private Slack channel. The discussions were generally productive and the developers provided assistance when necessary.

The commit integrating the feature in scope was intricate, involving approximately 13,000 code additions and 4,000 deletions. Furthermore, the Conduit feature has remained in continuous development over the past year, resulting in an altogether complex codebase written in Golang and consisting of approximately 1.1 million lines of code. This presented challenges in understanding its integration into the Psiphon environment due to the absence of high-level documentation; hence, the audit team had to solely rely on source code comments for guidance.

Due to the fact that not all issues identified in the 2023 audit had been immediately resolved at the time, one of the test team's first evaluation procedures was to confirm that all previously identified issues had been addressed.

To summarize, no vulnerabilities were located and only one minor issue was identified during this assignment. Consequently, a dedicated Test Methodology section has been included in this report in order to provide detailed information regarding the plethora of checks and tests conducted. This hopefully offers ample insight into the rigorous examination processes employed to ensure the widest possible testing coverage.

The Psiphon maintainers distributed a binary file along with configuration files, which the Cure53 testers utilized to establish a proxy and various clients for dynamic testing purposes. The central elements of the Conduit integration were identified in the code as the *InproxyBrokerClientManager* and *NewInproxyBrokerClientInstance* structures, which represent the primary modifications. These Golang compositions were meticulously audited for race conditions and configuration weaknesses pertaining to inproxy.

During the 2023 audit, the critically significant dial parameters were honed in on for careful investigation, which confirmed that the parameters in question are generated, stored, and distributed securely within the integration commit. Elsewhere, the code was analyzed for potential DoS vectors within the in-proxy application, though no such compromise opportunities were observed.

Dr.-Ing. Mario Heiderich, Cure53
Wilmersdorfer Str. 106
D 10629 Berlin
cure53.de · mario@cure53.de

Given the inherent risks that supply chain vulnerabilities can introduce to any contemporary software system, an extensive analysis of all software dependencies was carried out. This check confirmed that none of the dependencies were outdated or vulnerable at the time of testing, ensuring that the system's integrity was maintained and reducing potential security threats originating from external dependencies.

The only minor deficiency identified within the allotted time frame pertained to the fact that the WebRTC DataChannels established for inproxy communication use constant channel labels, as discussed in ticket PSI-07-001.

In conclusion, the significant complexity of the Psiphon code base increased the challenge of fully comprehending Conduit's operationality and integration into Psiphon. Furthermore, the testers required substantial time and resources to familiarize with the scope in question. Despite these difficulties, sufficient evidence that the integration prioritized high-level security was observed, reflecting a robust approach to safeguarding the system.

In general, the Psiphon Conduit integration would certainly profit from recurring security assessments that mirror the aims and coverage of previously strategized projects. The difficulty of evaluating such a complex framework of software, architecture stack, and associated components is irrefutable. Moreover, alterations within one application area can evoke undesirable impact elsewhere, which expands the risk of repeated, emerging, and system-wide security problems. Conducting supplementary security evaluations would certainly promote comprehensive and in-depth analyses, while detailed investigations (rather than broad and sometimes surface-level audits) would prove highly beneficial toward accurately determining the scope's security posture.

Cure53 would like to thank Adam Kruger and Rod Hynes from the Psiphon Inc. team for their excellent project coordination, support, and assistance, both before and during this assignment.