Fine penetration tests for fine websites

Dr.-Ing. Mario Heiderich, Cure53
Bielefelder Str. 14
D 10709 Berlin
cure53.de · mario@cure53.de

Pentest-Report Windscribe VPN Server 11.-12.2022

Cure53, Dr.-Ing. M. Heiderich, Dipl.-Inf. G. Kopf, B.Sc. M. Münch, F. Westendorf

Index

Dr.-Ing. Mario Heiderich, Cure53
Bielefelder Str. 14
D 10709 Berlin
cure53.de · mario@cure53.de

Introduction

"Windscribe encrypts your browsing activity, blocks ads, and unblocks entertainment content"

From https://windscribe.com/

This report details the scope, results, and conclusory summaries of a penetration test and source code audit against the Windscribe Limited in-memory VPN server image and post-provisioning VPN stack. The work was requested by Windscribe Limited in August 2022 and initiated by Cure53 in late November and early December 2022, namely in CW48 and CW49. A total of fifteen days were invested to reach the coverage expected for this project. The testing conducted for this audit was divided into two distinct Work Packages (WPs) for ease of execution, as follows:

- **WP1**: Source-code-assisted penetration tests against in-memory VPN server image
- **WP2**: Source-code-assisted penetration tests against post-provisioning VPN stack

Cure53 was provided with sources, assisting documentation, detailed insights into the targeted primary focus areas, and any alternative means of access required to ensure a smooth test completion. For this purpose, the methodology chosen was white box and a team consisting of four senior testers was assigned to the project's preparation, execution, and finalization. All preparatory actions were completed in November 2022, namely in CW47, to ensure the assessments could proceed without hindrance or delay.

Communications were facilitated via a dedicated, shared Slack channel deployed to combine the workspaces of Windscribe and Cure53, thereby creating an optimal collaborative working environment. All participatory personnel from both parties were invited to partake throughout the test preparations and discussions. In light of this, communications proceeded smoothly on the whole. The scope was well-prepared and transparent, no noteworthy roadblocks were encountered throughout testing, and cross-team queries remained minimal as a result. The Windscribe team delivered excellent test preparation and assisted the Cure53 team in every respect to procure maximum coverage and depth levels for this exercise.

Cure53 gave frequent status updates concerning the test and any related findings, whilst simultaneously offering prompt queries and receiving efficient, effective answers from the maintainers. Pertinent finding specifics were shared and discussed in the aforementioned Slack channel during the active testing phase, which proved highly useful for problem-solving and awareness purposes.

Dr.-Ing. Mario Heiderich, Cure53
Bielefelder Str. 14
D 10709 Berlin
cure53.de · mario@cure53.de

Concerning the findings, the Cure53 team achieved strong coverage over the WP1 and WP2 scope items, identifying a total of eleven. Four of the findings were categorized as security vulnerabilities, whilst the remaining seven were deemed general weaknesses with lower exploitation potential.

Generally speaking, the total yield of findings is relatively high, which would typically garner cause for concern regarding the Windscribe VPN server and stack's security offering. In this particular case, however, the fact that the majority of the discovered issues were merely considered general weaknesses or hardening advice - rather than tangible security flaws - ultimately negates this negativity to invoke a relatively favorable assessment on the whole. Nevertheless, four identified vulnerabilities were allocated a *High* severity rating and should be dealt with at the earliest possible convenience to avoid any inherent risk of attacker exploitation. Additionally, Cure53 would like to underline that the majority of issues - including the aforementioned *High* severity vulnerabilities - were identified during the reviews against the in-memory VPN server image, as denoted under WP1. This provides ample evidence of the requirement to instill stronger security measures in this area specifically.

All in all, the testing team did not detect any vulnerability worthy of a *Critical* severity marker during this engagement, and observed evidence that the Windscribe team has already implemented a host of best practices and security features to avoid numerous common vulnerabilities. Nevertheless, the Windscribe VPN server and stack's security posture would certainly benefit from hardening improvements. Cure53 strongly recommends adhering to all guidance offered in this report to elevate the components in scope to a first-rate standard from a security viewpoint.

The report will now shed more light on the scope and testing setup as well as provide a comprehensive breakdown of the available materials. Subsequently, the report will list all findings identified in chronological order, starting with the detected vulnerabilities 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 Windscribe in-memory VPN server image and post-provisioning VPN stack, giving high-level hardening advice where applicable.

Dr.-Ing. Mario Heiderich, Cure53
Bielefelder Str. 14
D 10709 Berlin
cure53.de · mario@cure53.de

Scope

- **Penetration Tests & Audits Against Windscribe's In-Memory VPN Server & Deployment**
  - **WP1**: Source-code-assisted pentests against in-memory VPN server image
    - **Hostnames:**
      - *yyz-h01.stg.windscribe.io*
      - *us-013.windscribe.dev*
    - **In-scope aspects:**
      - The Windscribe in-memory VPN server image
      - The base OS template, to be deployed using Quartermaster.
    - **Key focus areas:**
      - Tests covering secure system setup
      - Tests covering server setup
      - Tests covering secret management
      - Tests covering storage, deployment, post-deployment clean-up
  - **WP2**: Source-code-assisted pentests against post-provisioning VPN stack
    - **Hostnames:**
      - *see WP1*
    - **In-scope aspects:**
      - The actual server stack
      - Runtime after provisioning and deployment are completed
    - **Key focus areas:**
      - Tests covering robustness issues
      - Tests covering memory-safety issues
      - Tests covering server setup
      - Tests covering secret management
      - Tests covering secret storage
      - Tests covering logic issues
      - Tests covering potential access-control bypasses
  - **VPN access was granted for Cure53**
  - **SSH access was provided to the following hosts:**
    - *ws-admin@yyz-h01.stg.windscribe.io* and
    - *ws-admin @us-013.windscribe.dev*
  - **Test-supporting material was shared with Cure53**
  - **All relevant sources were shared with Cure53**

Dr.-Ing. Mario Heiderich, Cure53
Bielefelder Str. 14
D 10709 Berlin
cure53.de · mario@cure53.de

Identified Vulnerabilities

The following section lists all vulnerabilities and implementation issues identified throughout the testing period. Please note that findings are listed in chronological order rather than by their degree of severity and impact. The aforementioned severity rank is offered in brackets following the title heading for each vulnerability. Furthermore, each vulnerability is given a unique identifier (e.g., *WIN-01-001*) to facilitate any future follow-up correspondence.

WIN-01-003 WP1: Directory Traversal in OpenVPN Authentication *(High)*

***Fix note**: This issue has been mitigated by the Windscribe team and fix-verified by Cure53.*

During the analysis of the deployed OpenVPN configuration, the observation was made that authentication is performed using an OpenVPN plugin. The actual authentication functionality is implemented in the marvin service. This implementation exposes a directory traversal issue when attempting to create an OpenVPN configuration file for the respective user. The following code excerpt highlights the present behavior:

Affected file:
marvin/auth.go

Affected code:

```
// Ensure authentication succeeded
if !connectResponse.Success {
incrementAbuseCounter(ctx, clientIP, "openvpn")
return ErrBadConnectAuth
}

// If authentication succeeded, write the config
configFilePath := mConfig.OpenVPN.ControlFilePrefix + "/" + protocol +
      "/" + username
cf, err := os.OpenFile(configFilePath, os.O_WRONLY|os.O_CREATE|
      os.O_TRUNC, 0644)
if err != nil {
return fmt.Errorf("failed to create the config file: %s", err)
}
defer cf.Close()
```

Here, one can observe that the code concatenates the authenticated user's name into a file path, which it then attempts to write to. An attacker could provide a username such as *../../../../tmp/foo* in order to write to another file.

Dr.-Ing. Mario Heiderich, Cure53
Bielefelder Str. 14
D 10709 Berlin
cure53.de · mario@cure53.de

Notably, in order to instigate a successful attack, gimp mode must be activated for the respective node, which ensures that all authentication attempts are accepted regardless of the validity of the used credentials.

Furthermore, the established system hardening may render this issue challenging to exploit; an attacker may not simply be able to overwrite an arbitrary file such as */etc/passwd* via this technique. However, testing could not completely rule out the possibility of an attacker forcing an undesirable state upon the system by writing to an accessible file such as */var/run/keydb/keydb.sock*.

To mitigate this issue, Cure53 advises rejecting all usernames containing directory separators.

WIN-01-006 WP1: Locally-Bound Services Reachable Via Roundabout *(High)*

***Fix note**: This issue has been mitigated by the Windscribe team and fix-verified by Cure53.*

During the assessment of the roundabout proxy, the observation was made that the proxy does not prevent connections to services running on the respective node in general. Whilst restrictions have been established to block direct connections to addresses such as *127.0.0.1*, other IP addresses - such as the node's primary public IP address - can still be reached. In combination with the findings detailed in tickets WIN-01-007 and WIN-01-011, this can facilitate a condition whereby external attackers can leverage the roundabout proxy to establish TCP connections to services bound either on *0.0.0.0* or on one of the node's public IP addresses.

Services only bound to *127.0.0.1* cannot be reached. Reachable services, however, include those deemed critical, such as meeseeks and ssh. Whilst both require authentication credentials or signatures, this behavior is considered a clear violation of hardening best practices, since attackers in possession of access credentials - via social engineering, phishing, or similar - can open a command shell on the affected node.

Affected file:
roundabout/state/common.go

Affected code:

```
func IsPrivateIP(ip net.IP) bool {
   if ip == nil {
       return true
   }
   if ip.IsLoopback() || ip.IsLinkLocalUnicast() || ip.IsLinkLocalMulticast() {
       return true
```

Dr.-Ing. Mario Heiderich, Cure53
Bielefelder Str. 14
D 10709 Berlin
cure53.de · mario@cure53.de

```
    }

    for _, block := range PrivateIPBlocks {
        if block.Contains(ip) {
        return true
        }
    }
    return false
}
```

The function offered above is utilized to determine whether an IP address falls within the range of hosts that roundabout would not permit a connection for. However, this process lacks a check that would otherwise ensure the node's public IP addresses cannot be used.

The following PoC illustrates the present issue. Notably, the PoC assumes that gimp mode is active for the host:

> *Each VPN Node can be thought of as a military unit, consisting of the entire stack of services running on the node. Under normal operation, permission requests and status updates are passed up the chain of command, to Matt-Daemon on the Distributed Core Machines. Instructions and privileged information are passed down the chain of command.*
>
> *VPN Nodes are capable of operating autonomously with no communication with central command. This is called gimp mode.*
>
> *In this mode, a VPN Node will allow almost any connections, perhaps with a few safety checks which the node could validate locally. Once the Node re-establishes connection with a Distributed Core Machine, inauthentic connections are found and terminated.*

The commands offered below are executable from a remote host and are not allow-listed in the node's firewall ruleset.

PoC:

```
intersolar greg λ openssl s_client -host us-013.windscribe.dev -port 443
CONNECTED(00000003)
depth=2 C = US, ST = New Jersey, L = Jersey City, O = The USERTRUST Network, CN
= USERTrust RSA Certification Authority

[...]
CONNECT 181.215.52.101:22 HTTP/1.1
Host: 127.0.0.1
```

Dr.-Ing. Mario Heiderich, Cure53
Bielefelder Str. 14
D 10709 Berlin
cure53.de · mario@cure53.de

```
Proxy-Authorization: Basic dGVzdF9zcXVpZDpjdXJlNTM=
Ws-Grp: 127.0.0.1

HTTP/1.1 200 OK
Server: nghttpx

SSH-2.0-OpenSSH_8.2p1 Ubuntu-4ubuntu0.5
foobar
Invalid SSH identification string.
```

To mitigate this issue, Cure53 advises restricting the addresses the roundabout proxy can connect to and ensuring any IP addresses belonging to the respective node itself are blocked. Furthermore, roundabout should be prevented from using *127.0.0.1* as a source address.

WIN-01-007 WP1: Lax Firewall Rules for Local IPs *(High)*

Whilst reviewing the node's firewall configuration, the discovery was made that the ruleset is lax when handling traffic originating from a local IP address, such as *127.0.0.1*. For this source IP address, the firewall will accept all *INPUT* traffic, regardless of the destination IP, port number, or interface.

The following node excerpt demonstrates the present issue:

```
   32  8014 ACCEPT  udp  --  *    *      0.0.0.0/0           181.215.52.102
       udp dpt:500
   34  8918 ACCEPT  udp  --  *    *      0.0.0.0/0           181.215.52.101
       udp dpt:500
1519K   99M ACCEPT  all  --  *    *      127.0.0.0/8         0.0.0.0/0
 212K   10M ACCEPT  icmp --  *    *      0.0.0.0/0           0.0.0.0/0
       icmptype 8 state NEW,RELATED,ESTABLISHED
  163 17193 ACCEPT  udp  --  *    *      0.0.0.0/0           181.215.52.101
       udp dpt:51820
```

Henceforth, an attacker could exploit this behavior by leveraging the behavior described in ticket WIN-01-006, for instance.

To mitigate this issue, Cure53 advises preventing local IP addresses from accessing services they do not strictly require access to, including meeseeks and ssh.

Dr.-Ing. Mario Heiderich, Cure53
Bielefelder Str. 14
D 10709 Berlin
cure53.de · mario@cure53.de

WIN-01-011 WP1: Roundabout Accepts Test Credentials *(High)*

Whilst analyzing the roundabout implementation, the observation was made that the code includes a path utilized for testing purposes. In this code path, roundabout will accept an HTTP header named *Ws-Grp*, which specifies the source IP address used when establishing a connection. This code path is only triggered if the used username constitutes *test_squid*. The ability to select an outgoing IP address enables attackers to use an IP address such as *127.0.0.1* and exploit the lax firewall configuration described in ticket WIN-01-007, which in turn may ultimately facilitate a scenario such as that detailed in ticket WIN-01-006.

Whilst this may be considered a minor issue since the attacker must hold access to *test_squid* credentials, Cure53 would like to underline that this requirement will be bypassable if gimp mode is active for the node. In this case, an attacker can use any password as long as the *test_squid* user has not been authenticated with the system previously.

Affected file:
roundabout/cmd/windscribe.go

Affected code:

```
    // allow wanIP override for testing
    if clientConfig.Username == "test_squid" && proxyCtx.Req.Header.Get("Ws-Grp")
        != "" {
        newWanIP := proxyCtx.Req.Header.Get("Ws-Grp")
        logs.Logger.Debugln(ctx, fmt.Sprintf("overriding exit IP to %v",
newWanIP))
        proxyCtx.ForwardProxySourceIP = newWanIP
    }
```

To mitigate this issue, Cure53 advises removing the test code path entirely from production builds. If this is infeasible, one could alternatively restrict access to pertinent user accounts such as *test_squid* when gimp mode is active in the node, and excluding related usernames from any code path that does not require the user to fully authenticate.

Dr.-Ing. Mario Heiderich, Cure53
Bielefelder Str. 14
D 10709 Berlin
cure53.de · mario@cure53.de

Miscellaneous Issues

This section covers any and all noteworthy findings that did not lead to an exploit but might 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 way to be called. Conclusively, while a vulnerability is present, an exploit might not always be possible.

WIN-01-001 WP1: Lack of Certificate Verification *(Low)*

During the static analysis of the *node-kernel-initrd* package, the observation was made that a script downloads the *kernel* and *initrd* binaries from an internal host without verifying the server's certificate. This may allow an attacker in a privileged network position to provide malicious binaries that ultimately culminate on a production host.

Affected file:
node-kernel-initrd/prepare.sh

Affected code:

```
# Download kernel and initrd
wget -nc --no-check-certificate --user-agent="iPXE/123" ${ENDPOINT}/kernel -O
/boot/kernel
wget -nc --no-check-certificate --user-agent="iPXE/123" ${ENDPOINT}/initrd -O
/boot/initrd
```

To mitigate this issue, Cure53 advises evaluating the possibility to download the respective fragments only from servers that allow for successful verification of the provided certificates.

WIN-01-002 WP2: Quartermaster MD5 Usage For Auth Hash Generation *(Info)*

Whilst reviewing *quartermaster*, MD5 usage was identified for the purpose of client authentication hash generation. Generally speaking, MD5 usage is considered a negative practice since the current era deems it broken by modern cryptography standards. MD5 should be avoided due to inherent weaknesses, such as a lack of collision resistance and susceptibility to hash length extension attacks. Even though this particular use-case does not directly incur a vulnerability, the testing team deemed creating this ticket pertinent for completion purposes.

Notably, the utilized construction of *MD5(secret + time)* may indicate potential for reuse of deprecated tokens. Besides the actual token, the respective endpoint also accepts a parameter entitled *time*, which contains the timestamp used to generate the token.

Dr.-Ing. Mario Heiderich, Cure53
Bielefelder Str. 14
D 10709 Berlin
cure53.de · mario@cure53.de

The source code for this service was not provided for review; however, one should note that token reuse may be possible if it fails to check that the provided timestamp is sufficiently recent.

Affected file:
quartermaster/windscribe.go

Affected code:
```
func wsFetchServerData(hostname string) (*wsServerDataType, error) {
   var (
       wsAPIHost   = config.WS.APIHost
       wsAPISecret = config.WS.APISecret
   )

   time := fmt.Sprintf("%d", timestampUnix())
   clientAuthHash := hashMD5(wsAPISecret + time)
```

To mitigate this issue, Cure53 advises retracting support for MD5 and alternatively leveraging a modern hash function, such as SHA3.

WIN-01-004 WP1: OpenVPN Management Interface Usage *(Low)*

Whilst examining the deployed OpenVPN configuration, the observation was made that the OpenVPN management interface is activated and bound to a local IP address. Whilst this behavior does not represent a directly exploitable vulnerability, the node's overall attack surface is unnecessarily increased as a result. The management interface does not appear to be used by any locally-running services and hence should be disabled. The code excerpt offered below highlights the present issue:

Affected file:
node-assets/root/opt/openvpn/etc/server_tcp.conf

Affected code:
```
management 127.0.0.1 55591
server {{ cidrip(networks.openvpn_tcp, 0) }}
{{ cidrmask(networks.openvpn_tcp) }}
ifconfig {{ cidrip(networks.openvpn_tcp, 1) }} {{ cidrmask(networks.openvpn_tcp)
}}

topology subnet
```

Furthermore, Cure53 advises not binding to a TCP port in case this functionality is required at a later date, for which a Unix domain socket is considered the more optimal approach.

Dr.-Ing. Mario Heiderich, Cure53
Bielefelder Str. 14
D 10709 Berlin
cure53.de · mario@cure53.de

WIN-01-005 WP2: String Concatenation to Build Command String *(Info)*

During the static analysis, the testing team observed the presence of a common negative practice, specifically usage of string concatenation to build the final command string executed via a system shell. This antipattern is particularly prone to command injection, which may in turn facilitate compromise of the host system. Furthermore, the resulting command is executed via *ssh* and includes sensitive data that ultimately may culminate in a log or history file.

Affected file:
quartermaster/api_controllers_deploy.go

Affected code:

```
// Running neuralzyer via an ssh session.
   neuralyzerRunCmd := fmt.Sprintf(`echo '%s' | sudo -S '%s' --httpTimeout='%d'
--hostname='%s' --initrd='%s' --kernel='%s' --rootfs='%s' --dns='%s' --role-
id='%s' --secret-id='%s' %s`,
       sshPassword, …)
   if stdout, stderr, err := sshCmd(sshNodeClient, neuralyzerRunCmd); err != nil
{
```

To mitigate this issue, Cure53 advises using *execve* family library functions rather than executing commands via system shells. Additionally, parameterization in conjunction with input validation should be utilized, with usage of permitted arguments only enforced.

WIN-01-008 WP1: Node Software Surplus to Requirement *(Low)*

Whilst investigating the nodes using SSH, the observation was made that software packages are installed, which appears unnecessary for service operation purposes. For instance, *git* and *php* were confirmed installed on the analyzed node instance, which are arguably avoidable attack surfaces. A non-exhaustive list of packages worthy of consideration and potential removal is offered below:

Avoidable packages:

- *git*
- *php*
- *telnet*
- *s3cmd*
- *rsync*
- *ntfs-3g*
- *gcc*
- *fuse*

Dr.-Ing. Mario Heiderich, Cure53
Bielefelder Str. 14
D 10709 Berlin
cure53.de · mario@cure53.de

To mitigate this issue, Cure53 generally advises removing unnecessary software packages and therefore ensuring airtight defense-in-depth.

WIN-01-009 WP1: Disabled SSH Host Key Verification *(Medium)*

During the code audit of the shell scripts residing in *node-assets/scripts/*, the observation was made that the *users_windscribe.sh* bash script sets *StrictHostKeyChecking=no* in the *ssh* configuration file. This renders SSH connections vulnerable to Man-in-the-Middle attacks, since the SSH client will no longer verify the server's public key.

Affected file:
node-assets/scripts/users/users_windscribe.sh

Affected code:
```
cat > $USERHOME/.ssh/config <<- EOF
Host *
UserKnownHostsFile=/dev/null
StrictHostKeyChecking=no
EOF
```

To mitigate this issue, Cure53 recommends removing *StrictHostKeyChecking* from the config, which implicitly sets it to `ask` again. To avoid asking whether a certain public key or fingerprint is correct, all known host keys can be placed into *.ssh/known_hosts* by default and during deployment, for example.

WIN-01-010 WP1: Firewall Rules Applied Non-Atomically *(Low)*

Whilst reviewing the firewall ruleset configuration, the observation was made that the nodes' firewall rules are not applied in an atomic fashion by using *iptables-restore*. Rather, rules are applied one by one and sequentially using the *iptables* binary.

Affected file:
node-assets/scripts/network/firewall_ipv4.sh

Affected code:
```
#!/usr/bin/env bash

set -o pipefail

export IPT=/sbin/iptables

# See firewall.sh in `node-packer` for initialization of firewall

# Strongswan
```

Dr.-Ing. Mario Heiderich, Cure53
Bielefelder Str. 14
D 10709 Berlin
cure53.de · mario@cure53.de

```
## DNS TCP/UDP
${IPT} -w -A INPUT -s {{ networks.strongswan }} -d
{{ cidrip(networks.openvpn_tcp, 1) }}/32 -p tcp -m tcp --dport 53 -j ACCEPT
${IPT} -w -A INPUT -s {{ networks.strongswan }} -d
{{ cidrip(networks.openvpn_tcp, 1) }}/32 -p udp -m udp --dport 53 -j ACCEPT
${IPT} -w -A INPUT -s {{ networks.strongswan }} -d
{{ cidrip(networks.openvpn_udp, 1) }}/32 -p tcp -m tcp --dport 53 -j ACCEPT
```

This is considered a negative practice, since the firewall will be rendered in an intermediate state between deploying individual rules. For instance, the first rule will take effect before the second rule has been established, which may ease external system access during this window of opportunity between applying the firewall rules.

To mitigate this issue, Cure53 recommends applying firewall rules atomically by relying on *iptables-restore*, for instance.

Dr.-Ing. Mario Heiderich, Cure53
Bielefelder Str. 14
D 10709 Berlin
cure53.de · mario@cure53.de

Conclusions

The impressions gained during this report - which details and extrapolates on all findings identified during the CW48 and CW49 testing against the Windscribe in-memory VPN server image and post-provisioning VPN stack by the Cure53 team - will now be discussed at length.

To pass opening commentary, the available documentation and implementation was favorably received by the testing team. One can certainly argue that the robust software quality in turn facilitates a strong security foundation on the whole.

The general approach to leverage an in-memory method for node server provisioning helps to mitigate a number of post-exploitation scenarios, whereby attackers could otherwise attempt to obtain persistence on a node. However, Cure53 would like to underline that this inherently renders forensic analyses of possibly compromised nodes more challenging to achieve. In light of this, logging and monitoring should be considered crucial aspects of the entire process.

Evidently, the systems were designed with security in mind, considering that commonly-found issues were reliably mitigated and (in most cases) security-related components were constructed using modern primitives. Furthermore, the deployment of a memory-managed language completely negates a host of vulnerability classes that often proliferate in software systems of this nature, such as memory corruption.

The testing team also positively acknowledged the application of system hardening and documentation regarding the system's security properties. For instance, the use of systemd's *ReadWritePaths* configuration option greatly reduces any potential exploitability of filesystem access weaknesses, such as that described in ticket WIN-01-003. However, a noteworthy proportion of issues were identified following the completion of this audit, which in combination facilitated the ability to reach network services such as SSH, which by default should not be externally reachable.

The fact that an attacker can leverage a number of individual issues in tandem serves to underline the essential requirement for defense-in-depth. Even issues that appear benign or inexploitable should be addressed in order to prevent attackers from abusing multiple simultaneously and forming an exploit chain.

Dr.-Ing. Mario Heiderich, Cure53
Bielefelder Str. 14
D 10709 Berlin
cure53.de · mario@cure53.de

A plethora of minor issues were attributed to the integration of typical negative practices, an area that the developer team should focus on for follow-up improvement. Single design decisions remain questionable, including the detection of a period whereby unauthenticated requests are proxied via roundabout or gimp mode.

Regarding gimp mode specifically, one must stipulate that this mode's raison d'etre is to ensure nodes remain in an operational state, even in cases whereby they cannot connect to the central infrastructure and hence cannot sufficiently perform tasks such as user authentication. This is a worthwhile objective from both a user experience and security resilience viewpoint, but can also grant attackers compromise opportunities, as confirmed by the issue described in ticket WIN-01-011. In light of this, the developer team should reconsider whether this approach remains optimal for the components in scope. In relation to user experience, this mode could benefit from further restriction by enforcing that the node only accepts connections from user accounts that have successfully authenticated with the node previously, which would subsequently locally cache their credentials.

Additionally, the fact that one of the test systems was redeployed and did not match the provided source code is worthy of mention. In this regard, a vulnerability was identified on the system itself that was not however present in the provided source code. The auditing team was consequently required to allocate resources toward analyzing an issue that has already been mitigated through removal of the related functionality.

During this redeployment, gimp mode was inadvertently activated on the system and therefore accepted all authentication attempts, regardless of the validity of the used credentials. This incurred unnecessary confusion when analyzing the finding detailed in ticket WIN-01-006.

To conclude, whilst the testing team was able to unveil an array of vulnerabilities and general weaknesses, none resulted in a full system compromise or represented inherent design deficiencies. As a result, Cure53 is pleased to confirm that subsequent hardening and resolution of all tickets outlined in this report will comprehensively deliver the desired security properties for the analyzed system.

Cure53 would like to thank Yegor Sak, Alex Elisenko, Connie Lukawski, Konnor Klashinsky, Mark Ulicki, and all other participatory personnel from the Windscribe team for their excellent project coordination, support, and assistance, both before and during this assignment.