

Welcome



“From Zero to SSRF to RCE and back again”

- Bsides Basingstoke (2022-07-15) ~ Presented by Thomas Cope

Hello World!



- Thomas Cope (<https://tomcope.com>)
- Chief Security Officer (CSO) at Qush Security
- Working in Cyber Security ~9 years
- MSc in Software and Systems Security
- CISSP
- Part time Security Researcher (CVE-2020-5014 + CVE-2021-29707)



AWS Certified
Solutions Architect – Associate
Amazon Web Services
Training and Certification



CompTIA Security+
ce Certification
CompTIA



Certified Information Systems Security Professional (CISSP)
(ISC)²



Security and Privacy by Design Foundations
IBM



Think Like a Hacker
IBM



Government Insights and Solutions (Silver)
IBM



IBM Mentor
IBM



Docker Essentials: A Developer Introduction
IBM



IBM Recognized Teacher/ Educator
IBM



IBM Security Essentials for Architects
IBM



Get started with Kubernetes and IBM Cloud Container Service
IBM



Cloud Security Architect and Engineer Fundamentals
IBM

Qush Security

<https://www.qush.com>



- “Enterprise Grade AI Powered Human Centered DLP Solution”
- EDR Capability
- Extensive policy platform
 - “If user in dev team uploads source code to their personal gdrive on a work computer then block the upload and display a notification with a link to the AUP where my must provide a business reason to perform another upload”



Instant protection online and offline

Cloud delivered, with simple
setup, out-of-the-box policies
& ML



Minimize the occurrence of false positives

Tailored behavioral learning
and reinforced deep context



Industry leading privacy protection

Anonymization mode for
privacy & security compliance



Secure by design

Certified product, state-of-
the-art technology & practices



Fully scalable

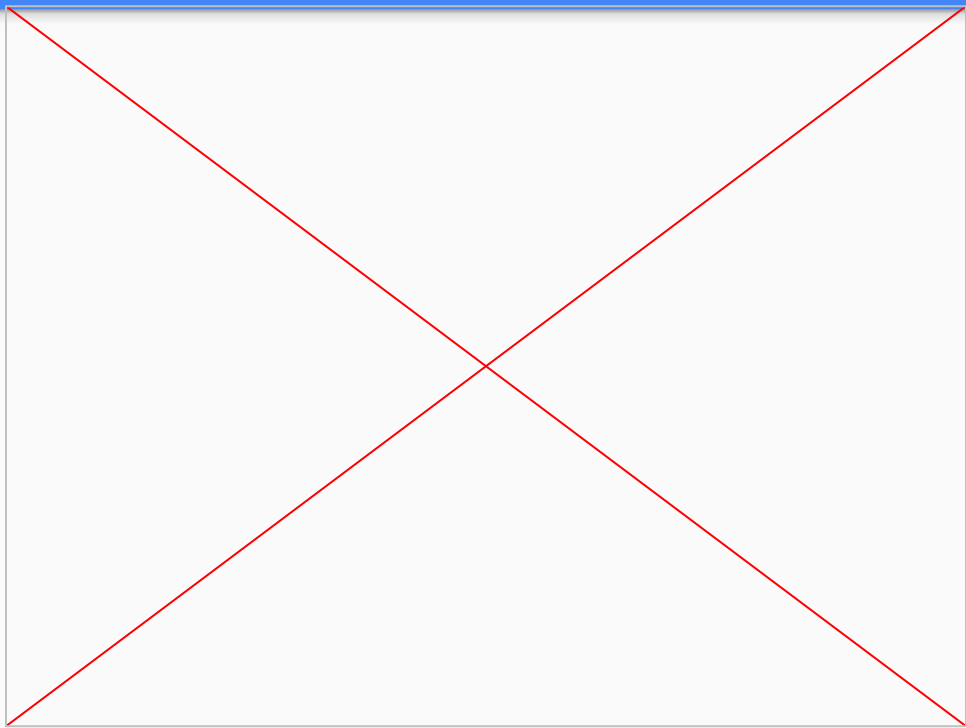
Massively scalable to 100k+
agents with
full control



Very low footprint

Powerful API integrations.
Minimal memory & CPU
required

Qush - Reveal



Enough about me, Lets begin!

What is up with the title?

Security Research
Process

Server Side Request
Forgery

Remote Code Execution

“From Zero to SSRF to RCE
and back again”

Real world impact

Security Research
datapower-redis-rce-exploit
(CVE-2020-5014)

Ethical Hacking Journey - Attackers Mindset

> Methodology



- Information Discovery – researching the target
- Target Scanning – identifying potential entry points
- Vulnerability Assessment – looking for weaknesses
- Exploiting Weakness – gaining access
- Privilege Escalation – increasing privileges for full access
- Retaining Access – maintaining control
- Covering Tracks – hiding evidence of intrusion



What differences in methodology might there be between a real attack and an authorised penetration test?

Information Discovery - What are you going to hack?

> Methodology



- Information Discovery – researching the target
- Target Scanning – identifying potential entry points
- Vulnerability Assessment – looking for weaknesses
- Exploiting Weakness – gaining access
- Privilege Escalation – increasing privileges for full access
- Retaining Access – maintaining control
- Covering Tracks – hiding evidence of intrusion

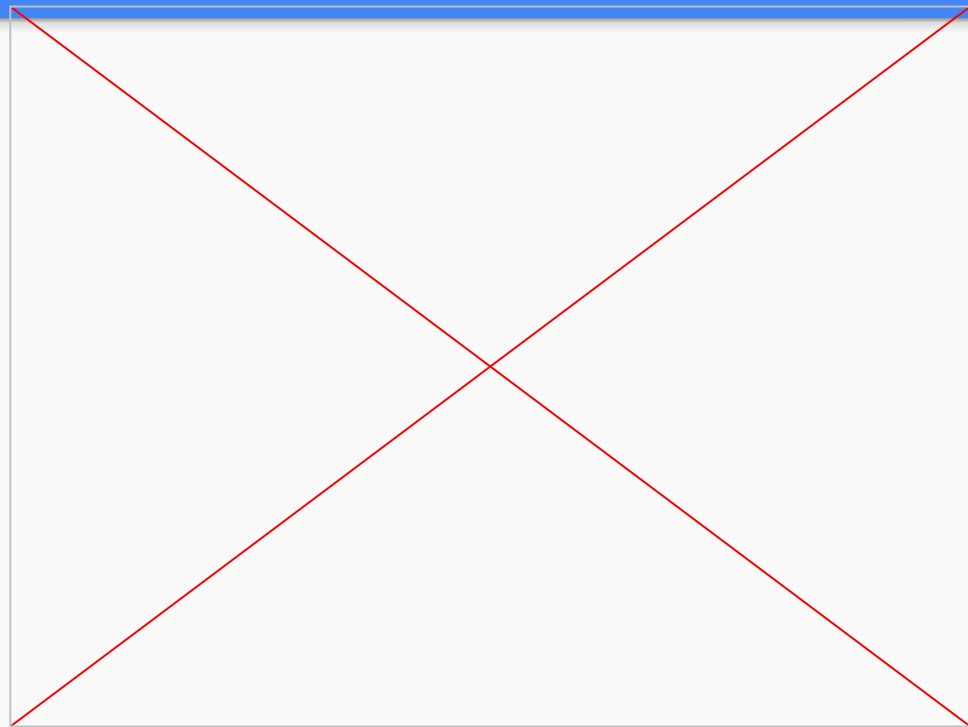


What differences in methodology might there be between a real attack and an authorised penetration test?

- Pick something fun!
 - Cool and enjoyable
 - Relevant to your current role
- Easy Access to support, images and documentation
- Real world impact
- Has a reporting program (HackerOne)
- Reward
 - Monies \$£\$£\$£\$£
 - Credit (CVE)
 - Hall of Fame - Google
 - For Fun
 - Gray hat side (Selling exploits to brokers)
 - Not Recommended

I picked - IBM Datapower

- Pick something fun!
 - Cool and enjoyable
 - Relevant to your current role
- Easy Access to support, Images and documentation
- Real world impact
- Has a reporting program (HackerOne)
- Reward
 - Monies \$£\$£\$£\$£
 - Credit (CVE)
 - Hall of Fame - Google
 - For Fun
 - Gray hat side (Selling exploits to brokers)
 - Not Recommended



IBM Datapower - Explained

- Hardened by design
 - No Shell access
 - Can't access the underlying operating system
 - Holds really important data like encryption keys
- Docker version allows you access to the inner workings
 - Physical Box is more common
 - Used in banks to process transactions at high speed
- If you forget the password you need to send it back to the factory
- Only access it via the GUI + Restricted management interfaces
- **Designed to be a really tough nut to crack**

Making Progress...

> Methodology



- Information Discovery – researching the target
- Target Scanning – identifying potential entry points
- Vulnerability Assessment – looking for weaknesses
- Exploiting Weakness – gaining access
- Privilege Escalation – increasing privileges for full access
- Retaining Access – maintaining control
- Covering Tracks – hiding evidence of intrusion



What differences in methodology might there be between a real attack and an authorised penetration test?

Setup

- Download
- Configure
- Look at the webgui
- “admin” “admin”
 - Shodan search???? - Scan the internet
 - Break into unconfigured docker containers (with approval from system owner, if your doing a pentest)

Target Scanning

- Break apart the container and have a look inside, see if we can find a good target!
 - Anyone know the docker commands?
 - What's running?
 - What's on the file system?

We have a few Options...

- Reverse engineer "drouter" (ghidra, Ida Pro) - String Search
- Web scanning of the WebGUI
- Telnet?
- SSH?
- **Redis looks like a good target**

Making Progress...

> Methodology



- Information Discovery – researching the target
- Target Scanning – identifying potential entry point
- Vulnerability Assessment – looking for weaknesses
- Exploiting Weakness – gaining access
- Privilege Escalation – increasing privileges for full access
- Retaining Access – maintaining control
- Covering Tracks – hiding evidence of intrusion



What differences in methodology might there be between a real attack and an authorised penetration test?

Vulnerability Assessment

Is Redis Vulnerable?

...

...

...

...

Let's find out!

(127.0.0.1:16379)

Vulnerability Assessment

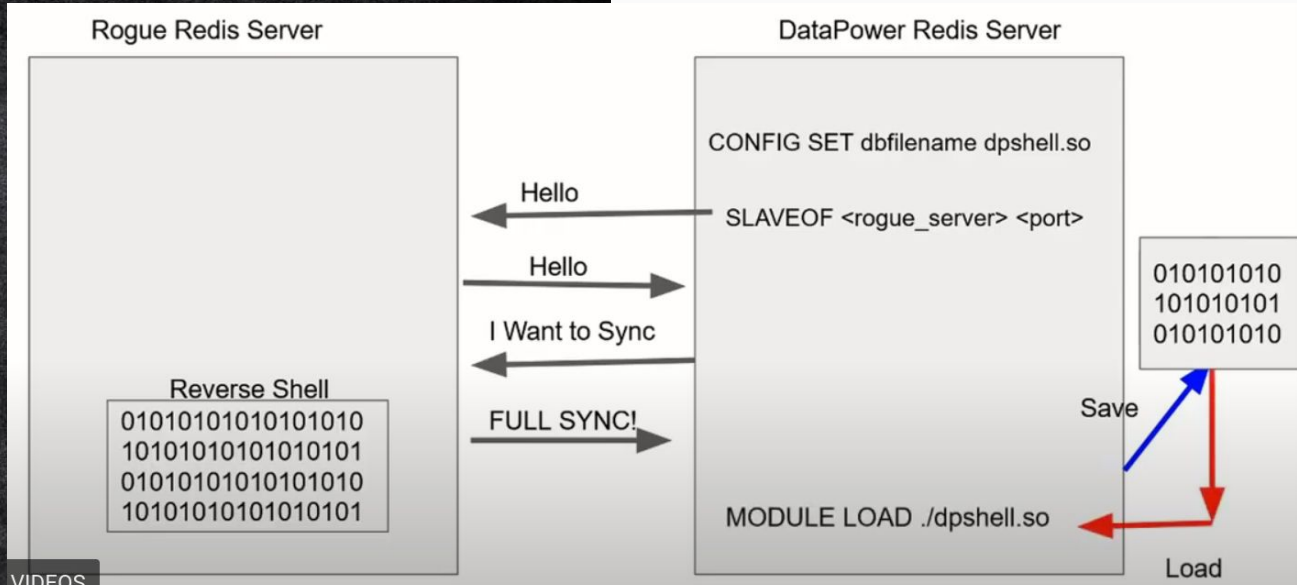
- Info Command
- Password needed
- Hard Coded Password
- Is it hard coded???
- Google Redis exploits

Vulnerability Assessment - Redis RCE

2³
EDITION

Redis post-exploitation

Pavel Toporkov



VIDEOS

2018.ZERONIGHTS.ORG

Rogue Redis Exploit

- Crazy Powerful exploit
- Any Redis server on the internet
- **Instant Remote Code Execution!**
- Redis introduced “Protected Mode” by default to help prevent this
 - Defaults to only listening on localhost
 - Redis Security guide recommends disabling the use of the “module” command
- Do people follow any of this advice or use the “Protected Mode”?
- If you Look at Shodan right now -> You'll find some Vulnerable ones

Vulnerability Assessment

Redis looks like a good target! But..... How to talk to it?

```
/opt/ibm/datapower/root/dp-redis-server 127.0.0.1:16379  
/opt/ibm/datapower/root/dp-redis-sentinel 127.0.0.1:26379 [se
```

127.0.0.1 - Localhost - No external connections
-> Not protected mode - Datapower team hardening

Making Progress...

> Methodology



- Information Discovery – researching the target
- Target Scanning – identifying potential entry point
- Vulnerability Assessment – looking for weaknesses
- Exploiting weakness – gaining access
- Privilege Escalation – increasing privileges for full access
- Retaining Access – maintaining control
- Covering Tracks – hiding evidence of intrusion



What differences in methodology might there be between a real attack and an authorised penetration test?

Exploiting Weakness

We need a SSRF!

<https://portswigger.net/web-security/ssrf>

What is SSRF?

Server Side Request Forgery

“A Web Security vulnerability that allows an attacker to induce the server-side application to make HTTP requests to an arbitrary domain of the attacker's choosing”

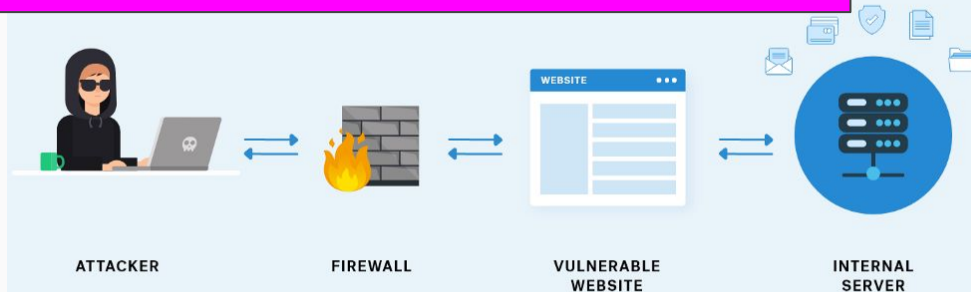
- Connecting to itself
- Internal Services
- Third Party Systems

A Scary class of exploits

Quick Example

`https://app.com/api/v1/webhook?name=test?addr=127.0.0.1`

Woops! That not allowed! Should only talk to Websites on the internet! Not itself! Or how about 192.168.0.12 or 10.10.100.12



You put your Database / Redis / Internal systems of a different network, blocked off from the internet but your own application is tricked into sending malicious requests by an attacker

Possible Impacts

- Used as a hopping point to attack another company.
 - Request would originate from a different - Reputation Damage / DDOS
- Access to internal APIs / Services
 - Database endpoints
 - Authentication services
 - **Redis**
- Cloud Metadata
 - **Potential access to service accounts credentials**
 - Could then be used to access restricted cloud buckets / databases
- K8s
 - Access to K8s APIs, Secrets, Config Maps
 - Service accounts for further access to K8s

In the Wild - Exploiting JIRA (CVE-2017-9506)

```
/plugins/servlet/oauth/users/icon-uri?consumerUri=https://www.attacker.com
```

AWS - IAM role will leak AWS key

```
http://169.254.169.254/latest/meta-data/
```

Docker - List Containers

```
http://127.0.0.1:2375/v1.24/containers/json
```

Kubernetes ETCD - Can contain API keys and internal ip and ports

```
http://127.0.0.1:2379/v2/keys/?recursive=true
```

Exploiting Weakness

Hmmm... What's this?

Not secure | <https://127.0.0.1:9090/webguiapp/soapBox>

DataPower Gateway

Send a Test Message

Request

URL:

Request Headers:

Header Name	Value	
<input type="text"/>	<input type="text"/>	<input type="button" value="Add"/>

Request Body:

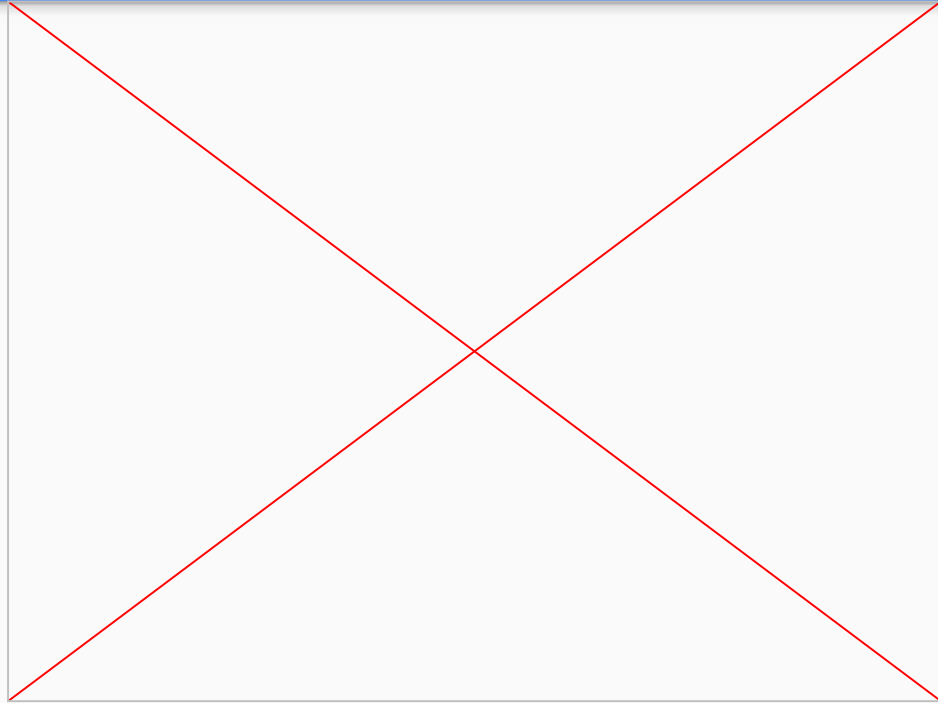
Response

Response Code:

Response Headers:

Response Body:

Exploiting Weakness



Making Progress...

> Methodology



- Information Discovery – researching the target
- Target Scanning – identifying potential entry point
- Vulnerability Assessment – looking for weaknesses
- Exploiting Weakness – gaining access
- Privilege Escalation – increasing privileges for full access
- Retaining Access – maintaining control
- Covering Tracks – hiding evidence of intrusion



What differences in methodology might there be between a real attack and an authorised penetration test?

Next Steps

- Privilege Escalation
 - Redis already runs with high privilege
 - No Principle of least privilege :(
- Retaining Access
 - RCE - easy to add a persistent backdoor / reverse shell
- Covering Tracks
 - Can we make it more sneaky?

Let's Write the Exploit!

- Golang
- Standard Library
- ~2 Hours

Let's Run It!

- Demo Time!

It works!

- Zero to SSRF to RCE!
- The Scope
 - Any Datapower
 - Anywhere on the internet
 - Only need any form of login (even the lowest level account)
 - Gain full root access, bypass the restricted environment and run a persistent remote shell
 - Extract encryption keys that were never meant to leave the device

CVE Score - 8.8 (High)

<https://www.first.org/cvss/calculator/3.0#CVSS:3.0/AV:N/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:H>

Base Score

8.8
(High)

Attack Vector (AV)

Network (N)

Adjacent (A)

Local (L)

Physical (P)

Attack Complexity (AC)

Low (L)

High (H)

Privileges Required (PR)

None (N)

Low (L)

High (H)

User Interaction (UI)

None (N)

Required (R)

Scope (S)

Unchanged (U)

Changed (C)

Confidentiality (C)

None (N)

Low (L)

High (H)

Integrity (I)

None (N)

Low (L)

High (H)

Availability (A)

None (N)


Low (L)

High (H)

Let's Report It!

hackerone

SOLUTIONS ▾PRODUCTS ▾PARTNERS ▾COMPANY ▾HACKERS ▾RESOURCES ▾



IBM
<http://ibm.com> · @IBM

Reports resolved
3276

Assets in scope
3

[Submit report](#)

Vulnerability Disclosure Program
Launched on Sep 2018
Managed by HackerOne

[Policy](#) [Hacktivity](#) [Thanks](#) [Updates \(2\)](#)

Policy

IBM recognizes how important the security community is in keeping our products, offerings, services and websites safe for our customers and users. We thank you in advance for your contributions to our vulnerability disclosure program. Vulnerability reports submitted via this program will be handled by IBM's global Product Security Incident Response Team (PSIRT). This team will coordinate with other IBM teams to investigate, and if needed, identify the appropriate response plan. Maintaining communication between all involved parties, both internal and external, is a key component of our vulnerability response process.

Response Efficiency

10 hrs
Average time to first response

2 months
Average time to resolution

1234567

tom@Awesome420 | cpu: 37% | wifi: 4GEE-WiFi-0566-2.4GHz (73%) | no lan | bat: 100% | mem: 5.2 GiB / 9.4 GiB --> 2022-05-18 22:57:45 <--

Let's Report It!

> Timeline

- 21st October 2020 = Initial Report filed via Hackerone
- 22nd October 2020 = IBM Confirmed receipt of report
- 4th November 2020 = Asked for a Update
- 10th November 2020 = Initial Review complete → Report moved to Triaged
- 7th December 2020 = Asked for a Update
- 9th December 2020 = IBM are still investigating
- 6th January 2021 = Asked for a Update
- 13th January 2021 = Confirmed valid vulnerability and are working with product teams to develop a fix
- 21st January 2021 = Asked for Acknowledgment
- 22nd January 2021 = Confirmed public acknowledgement with my name - "Thomas Cope"
- 25th January 2021 = Confirmed Acknowledgment and a CVE number would be issued
- 8th February 2021 = Asked for a Update
- 9th February 2021 = IBM are working on a Fix
- 5th March 2021 = Asked for a Update
- 8th March 2021 = IBM confirmed Fix and provided link to public security bulletin

Total time elapsed between initial report and resolution = 139 days (4 months, 16 days)

<https://tomcope.com/exploit/2020/10/21/ibm-datapower-exploit-cve-2020-5014.html>

All Worth it! - Fixed and CVE!

Security Bulletin: IBM DataPower Gateway vulnerable to an RCE attack (CVE-2020-5014)

Security Bulletin

Summary

IBM has addressed the relevant CVE

Vulnerability Details

CVEID: [CVE-2020-5014](#)

DESCRIPTION: IBM DataPower Gateway could allow a local attacker with administrative privileges to execute arbitrary code on the system using a server-side request forgery attack.

CVSS Base score: 6.7

CVSS Temporal Score: See: <https://exchange.xforce.ibmcloud.com/vulnerabilities/193247> for the current score.

CVSS Vector: (CVSS:3.0/AV:L/AC:L/PR:H/UI:N/S:U/C:H/I:H/A:H)

Affected Products and Versions

Affected Product(s)	Version(s)
IBM DataPower Gateway	10.0.0.0-10.0.1.1
IBM DataPower Gateway	2018.4.1.0- 2018.4.1.1.1

All Worth it! - Fixed and CVE!

The screenshot shows the CVE Details website interface. At the top, there's a navigation bar with links like 'CVE List', 'CNAs', 'WGs', 'Board', 'About', and 'News & Blog'. Below this is a search bar and a 'Search CVE List' button. A notice banner states: 'NOTICE: Transition to the all-new CVE website'. The main content area displays details for CVE-2020-5014, including a description of an SSRF vulnerability in IBM DataPower and a list of references. A large, semi-transparent watermark is overlaid diagonally across the page, reading: 'Acknowledgement This vulnerability was reported to IBM by Thomas Cope'.

HOME > CVE > CVE-2020-5014

CVE-ID
CVE-2020-5014

Description
IBM DataPower is vulnerable to a server-side request forgery (SSRF) attack, which allows an attacker to execute arbitrary code on the system using a server-side request forgery attack. IBM X-Force ID: 193247.

References
Note: [References](#)

- CONFIRM: <https://www.ibm.com/support/pages/node/6426789>
- URL: <https://www.ibm.com/support/pages/node/6426789>
- XF: <https://exchange.xforce.ibmcloud.com/vulnerabilities/193247>

[Printer-Friendly View](#)

The fix

If you are looking for a side project...

The Real World...

Qush Impacted - Customer Pentest

- PSIRT 416 - Aware webhooks can make requests against internal components
- PSIRT 417 - Cyber internal IP blacklist not configured
- PSIRT 418 - Aware webhooks can make requests against cloud components
- PSIRT 420 - Video vulnerable to various SSRF attacks

4 Security Issues!

Pentest

Timestamp	Score	Policy	Description
20/04/2021, 04:36	Critical	nothing to see here	This sensor was not created by a webhook.
20/04/2021, 04:36	Critical	nothing to see here	This sensor was not created by a webhook.
20/04/2021, 04:36	Critical	nothing to see here	This sensor was not created by a webhook.
20/04/2021, 04:36	Critical	nothing to see here	This sensor was not created by a webhook.

Threat Modeling Identified the issue -> Created library to handle checks (with tests + code review) -> Implemented the library -> BUT

On Closer inspection library was not integrated correctly. Integration tests are important!

The Solution

- Following Guidance from:
https://cheatsheetseries.owasp.org/cheatsheets/Server_Side_Request_Forgery_Prevention_Cheat_Sheet.html
 - -> Defence in Depth <-
 - Whitelist domain accessible by the service - Done in code
 - Network protections - K8s Network Policy + Firewalls
 - Proactive monitoring of network traffic
 - Isolation of services
 - Require authentication to all backend services

The Solution - Custom Library

```
8
9 func main() {
10     log.Println(v...: "Amazing Webhook Test")
11     c, err := internalip.NewHttpClient( rule: "local")
12     if err != nil {
13         log.Fatal(err.Error())
14     }
15     r, err := c.Get( url: "https://example.com")
16     if err != nil {
17         log.Fatal(err.Error())
18     }
19     defer r.Body.Close()
20     if r.StatusCode != http.StatusOK {
21         log.Fatal(r.Status)
22     }
23     log.Println(v...: "Internet test OK")
24     r, err = c.Get( url: "http://127.0.0.1")
25     if err != nil {
26         log.Println(v...: "URL is not allowed, this is a expected error:")
27         log.Fatal(err.Error())
28     }
29     log.Fatal(v...: "Ow no :(, the request should have failed")
30 }
31
main()
```

The Solution

```
2021/05/24 10:22:16 Amazing example using the 'internalip' HTTP Client
2021/05/24 10:22:17 Internet test OK
2021/05/24 10:22:17 URL is not allowed, this is a expected error:
2021/05/24 10:22:17 Get "http://127.0.0.1/": 127.0.0.1 failed to resolve into any permitted hostnames
exit status 1
```

My Personal Security Research helped
prepare me for handling the incident at
work

Before I Close out...

Now Hiring!



Graduate Software Engineer careers

- Test Automation
- Cloud architecture / infrastructure
- Security product development
- Backend or Frontend
- Machine Learning
- Data Pipeline
- Kernel layer OS systems
- Go (Golang)
- C
- C++
- C#
- Python
- Elm
- Typescript
- Kafka
- Kubernetes
- Docker
- Flink
- Spark
- GKE
- GCP
- Postgres
- ML

Thankyou! Any Questions?



<https://www.qush.com/careers>

Qush