# Secure Coding Refuelled

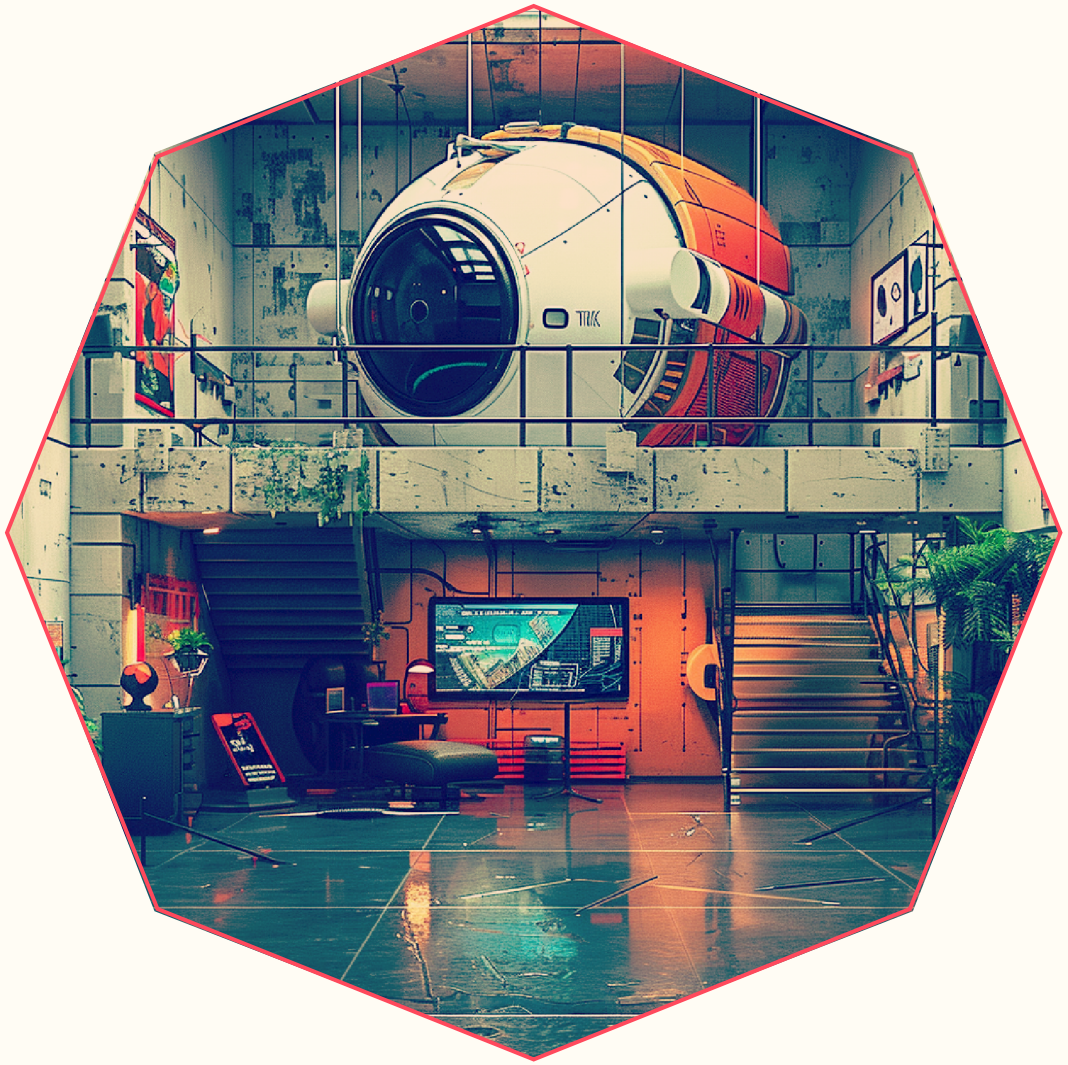
## IBM Adversary Services Lunch & Learn

Ruben Boonen

IBM

# What is IBM Adversary Services?

- **Red Team**, servicing external clients
- Strategic, **objective based**
  - Long running (**~3 months**)
  - High-impact & **covert**
  - Advanced, mature clients
- whoami?
  - CNE Capability Development Lead
  - Windows Systems Programming
  - Vulnerability research Windows Userland/Kernel, Android Userland (**and anything really**)
  - Reverse engineering
  - **Full-stack-random** developer



# Why do bugs matter?

Bugs on the **kill-chain**

## External

The external attack surface includes **apps** and **services**

## On-Host

Corporate endpoints have **software stacks** installed, often with custom components (e.g. .NET services)

## Internal

**Custom** and **commercial apps** and **services** (e.g. Jira, ESXi, Jenkins)

## Attacker expanding access

**Gradually attackers expand access**, including using vulnerabilities

## Client Sensitive Data

**High-privilege access** to business sensitive data

# Show me the vulnerabilities!

- Software vulnerabilities can be hard to understand
- How did you **select bugs**?
  - **Recent**
  - Target audience is especially interested in **web applications**
  - **Relatively low complexity** Root-Cause Analysis (**RCA**)
- We have **severe time limitations**
  - Full RCA and exploitation analysis could takes days or weeks
  - **Rapid fire**

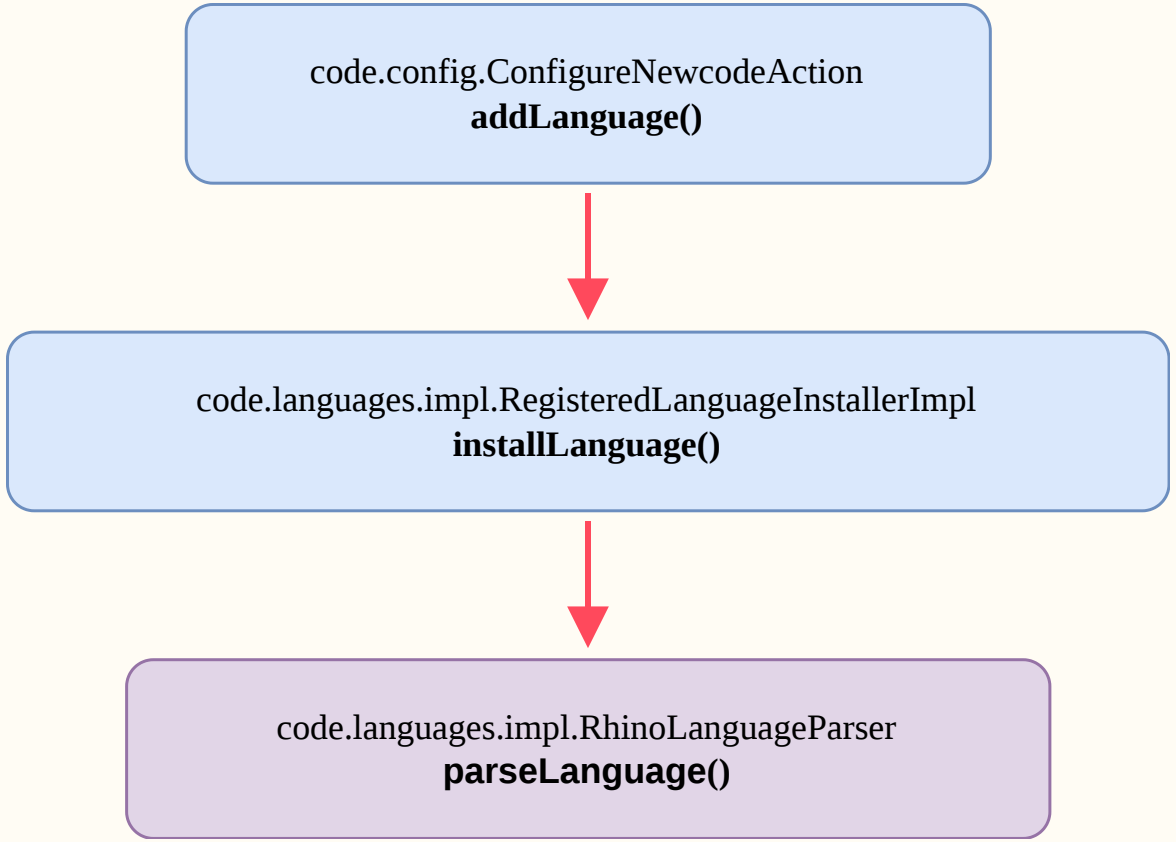
# CVE-2024-21683

Atlassian Confluence

Authenticated RCE

# Vulnerability Background

- Confluence has many default plugins, including **Code Macro**
  - Allows users, **with appropriate permissions**, to upload **new language** definitions for **syntax highlighting**
  - https://github.com/syntaxhighlighter/syntaxhighlighter
- Content is uploaded as a **JavaScript file**
  - Expected to follow **specific definition formatting**
- Very **cool**, very **business**
  - For sure our users are **only experts** that will create **properly formatted definition files**

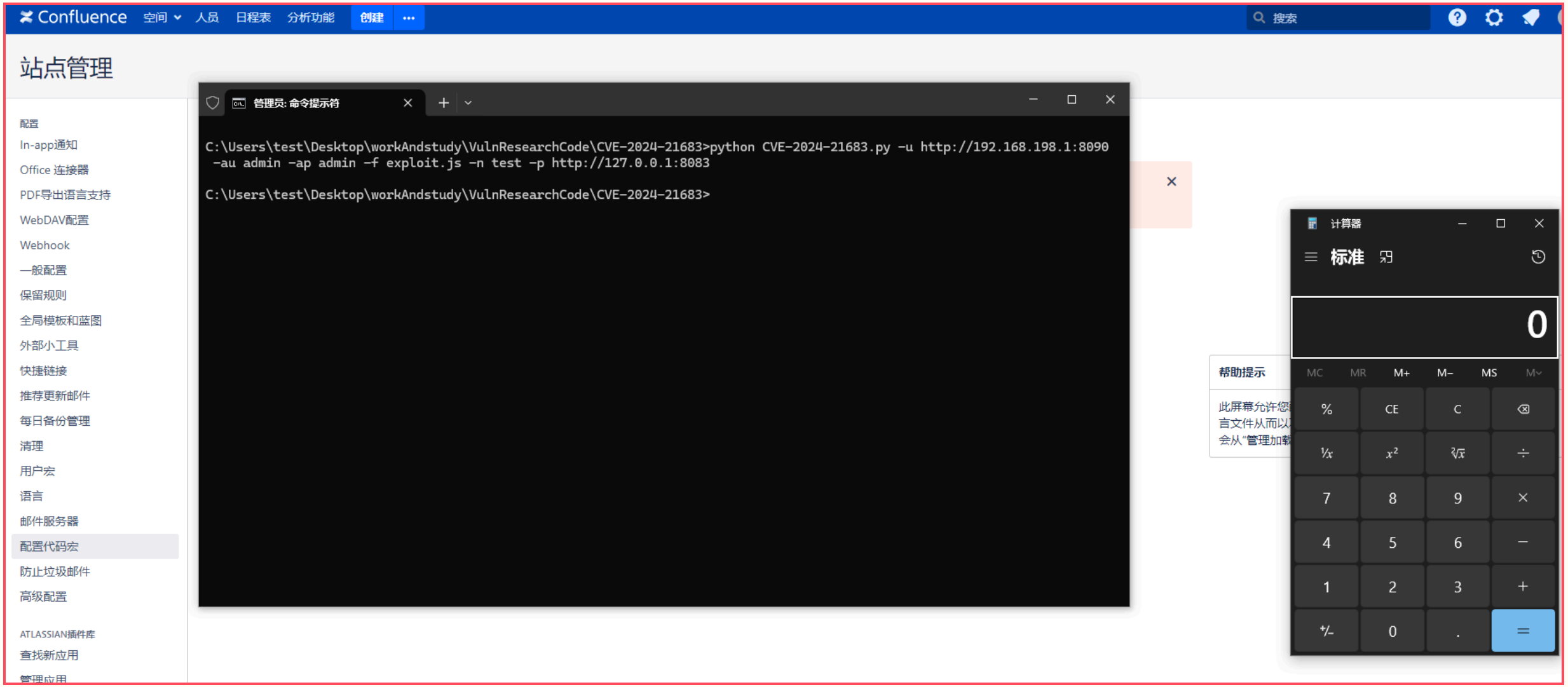


# What is the problem?

- **parseLanguage** creates a scriptable object using **initStandardObjects**
  - https://javadoc.io/doc/org.mozilla/rhino/1.7.7.2/org/mozilla/javascript/Context.html
- The user-controlled JavaScript file is **evaluated** without further filtering, this lets us **instantiate Java Objects**
  - **ProcessBuilder**, **Exec**, **...**

| | |
|---|---|
| `ScriptableObject` | `initSafeStandardObjects()`<br>Initialize the standard objects, leaving out those that offer access directly to Java classes. |
| `Scriptable` | `initSafeStandardObjects(ScriptableObject scope)`<br>Initialize the standard objects, leaving out those that offer access directly to Java classes. |
| `ScriptableObject` | `initSafeStandardObjects(ScriptableObject scope, boolean sealed)`<br>Initialize the standard objects, leaving out those that offer access directly to Java classes. |
| `ScriptableObject` | `initStandardObjects()`<br>Initialize the standard objects. |
| `Scriptable` | `initStandardObjects(ScriptableObject scope)`<br>Initialize the standard objects. |
| `ScriptableObject` | `initStandardObjects(ScriptableObject scope, boolean sealed)`<br>Initialize the standard objects. |

```
new java.lang.Runtime.getRuntime().exec("Not good!")
```




@W01fh4cker

https://github.com/W01fh4cker/CVE-2024-21683-RCE

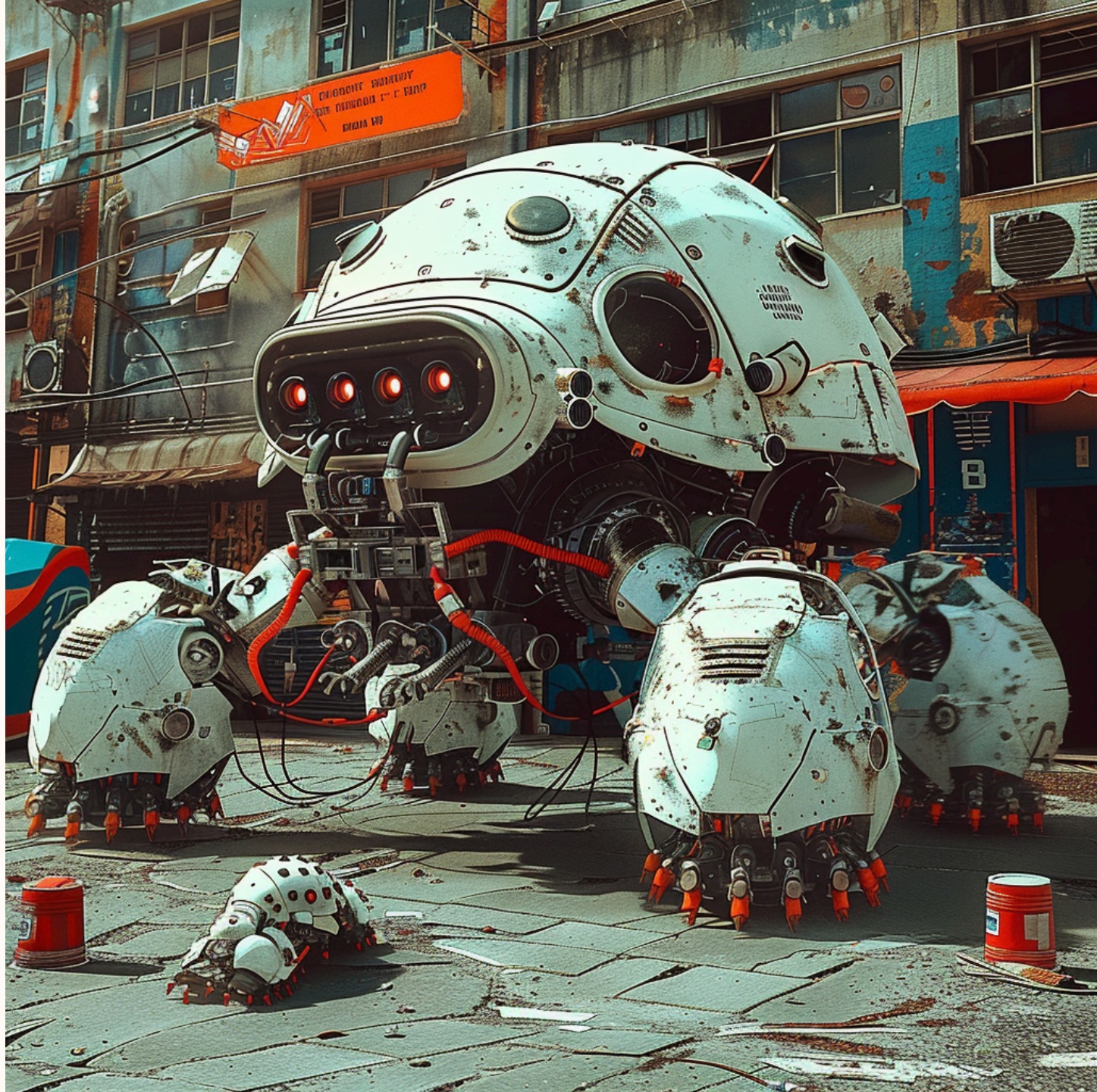
# Lessons learned

- Do not trust **user input**
  - **Users** can and will **break your assumptions** by accident
  - Attackers will look for **flawed developer assumptions**
- Carefully consider **execution sinks**
  - What does this function call into?
  - What is the sink doing with my inputs?
- Use **safer alternatives** where possible
  - Do you need all the functionality you are bringing into the application?
  - Is there a way to **restrict code functionality**?
  - e.g. **initSafeStandardObjects**

## CVE-2022-2915

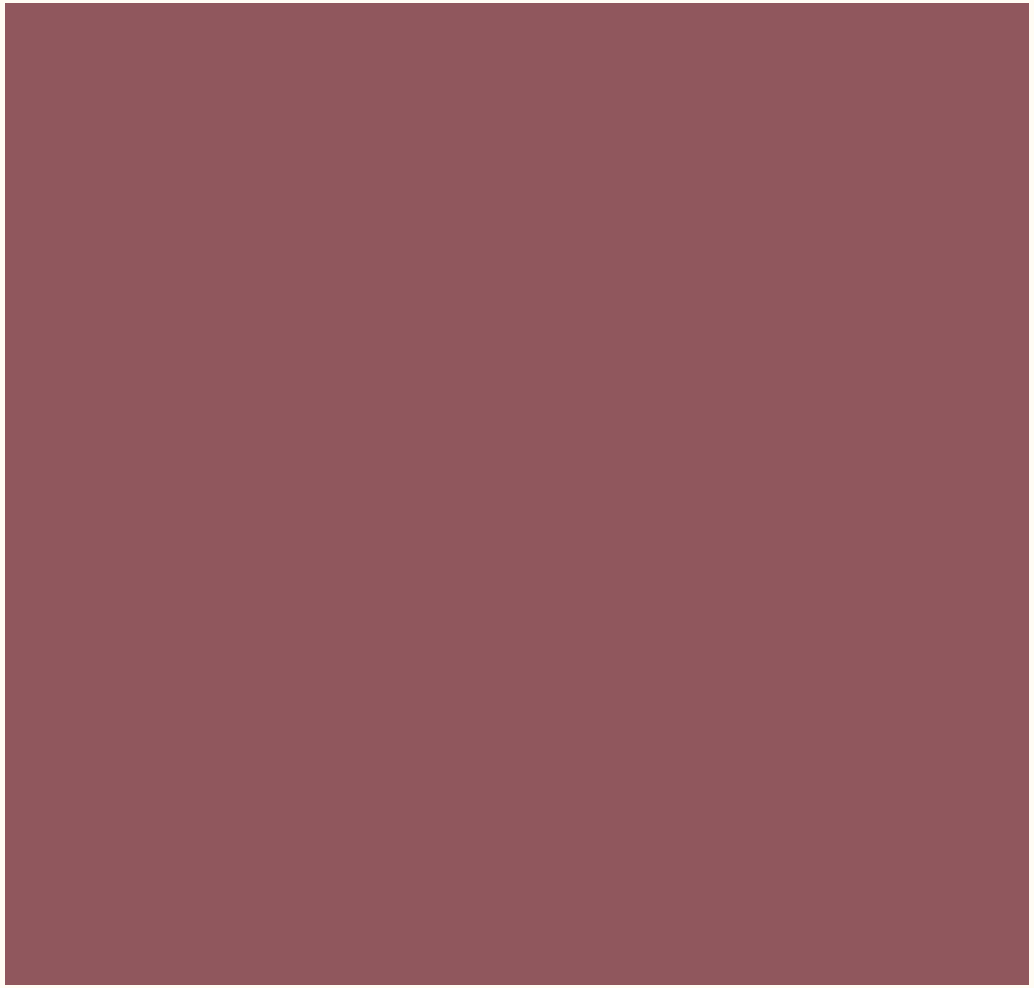
SonicWall Secure Mobile Access (SMA) Gateway

Authenticated RCE



# Vulnerability Background

- **SonicWall** does many things
  - Web Application Firewall (**WAF**) and **VPN**
  - Service integration: SAML, Citrix XenDesktop, Microsoft AD, etc
  - Like **Cisco**, **Fortinet**, **Palo Alto**, etc
- These types of devices have had **shockingly bad** internal development **quality**
  - Many **0days** identified **In-The-Wild** (ITW)
  - An attacker sitting on a gateway, DIY OS, no monitoring or IR capabilities, this is **very scary**
- Common Gateway Interface (**CGI**)
  - **SonicWall** has a **cgi-bin endpoint** called



# What is the problem?

Very easy to understand, **sloppy development** leading to **memory corruption**.

Not a problem for normal users, but **attackers** can get **heap Out-Of-Bound** (OOB) **write**.

# Lessons learned

- Exploitation [redacted]
- What did we learn?
  - Be careful when you write code and **keep design choices consistent**
  - Be especially careful with **unsafe memory access**
  - Where possible, **migrate future development to memory safe languages**
    - https://media.defense.gov/2022/Nov/10/2003112742/-1/-1/0/CSI_SOFTWARE_MEMORY_SAFETY.PDF

## CVE-2024-30169

Zeus Z3 SME-01 H.264 video encoder & decoder

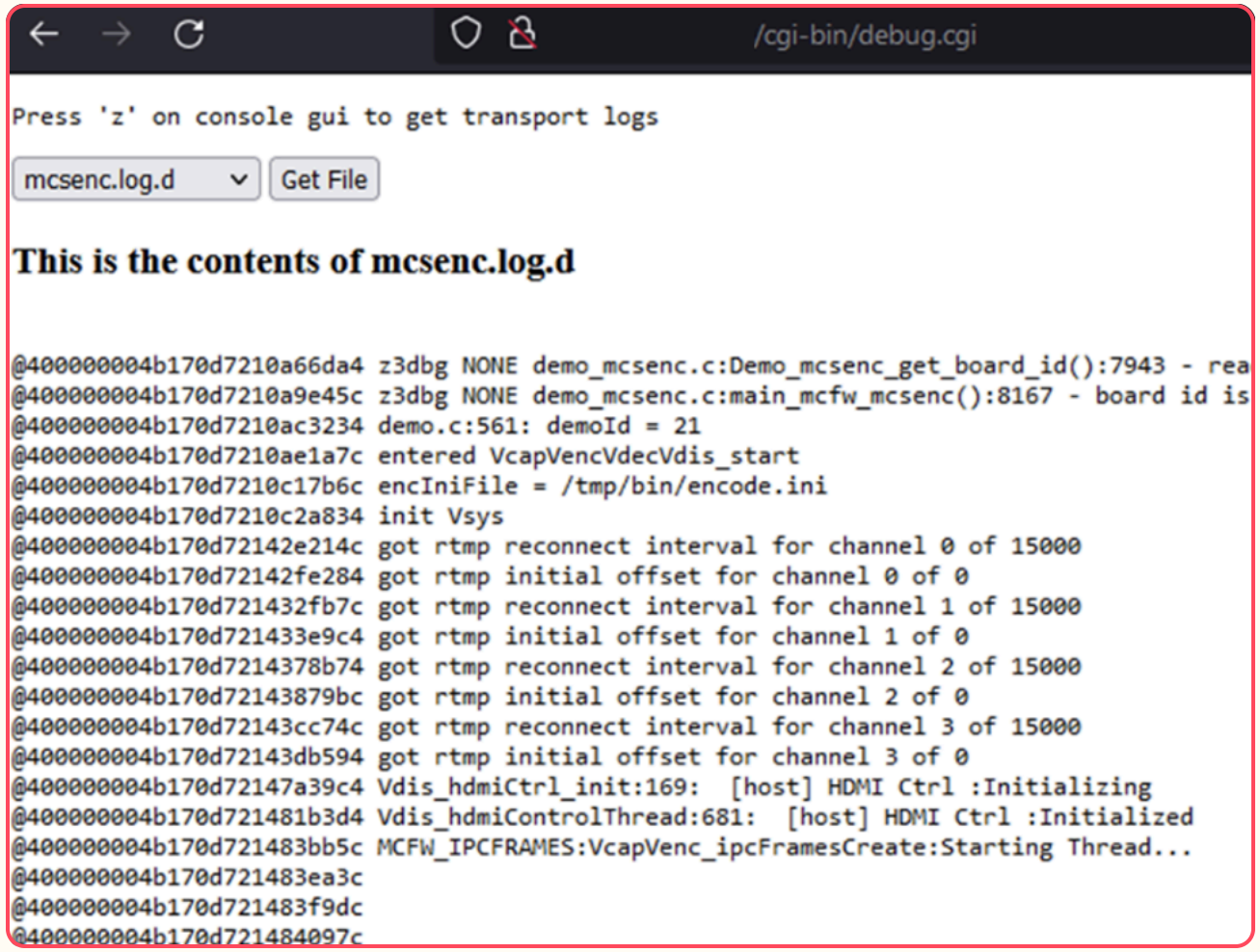
Unauthenticated RCE

# Vulnerability Background

- Found by one our awesome colleagues, **Todd Hastings**
- Actually this was part of **multiple CVE disclosures**
  - Get more information from the **IBM** Security Intelligence **blog post**
  - Very **thoughtful discussion** about the risks of attackers taking over **low-visibility IOT devices**
  - https://securityintelligence.com/x-force/iot-exploitation-during-security-engagements/
- Physical and virtual appliances often make **terrible** design decisions

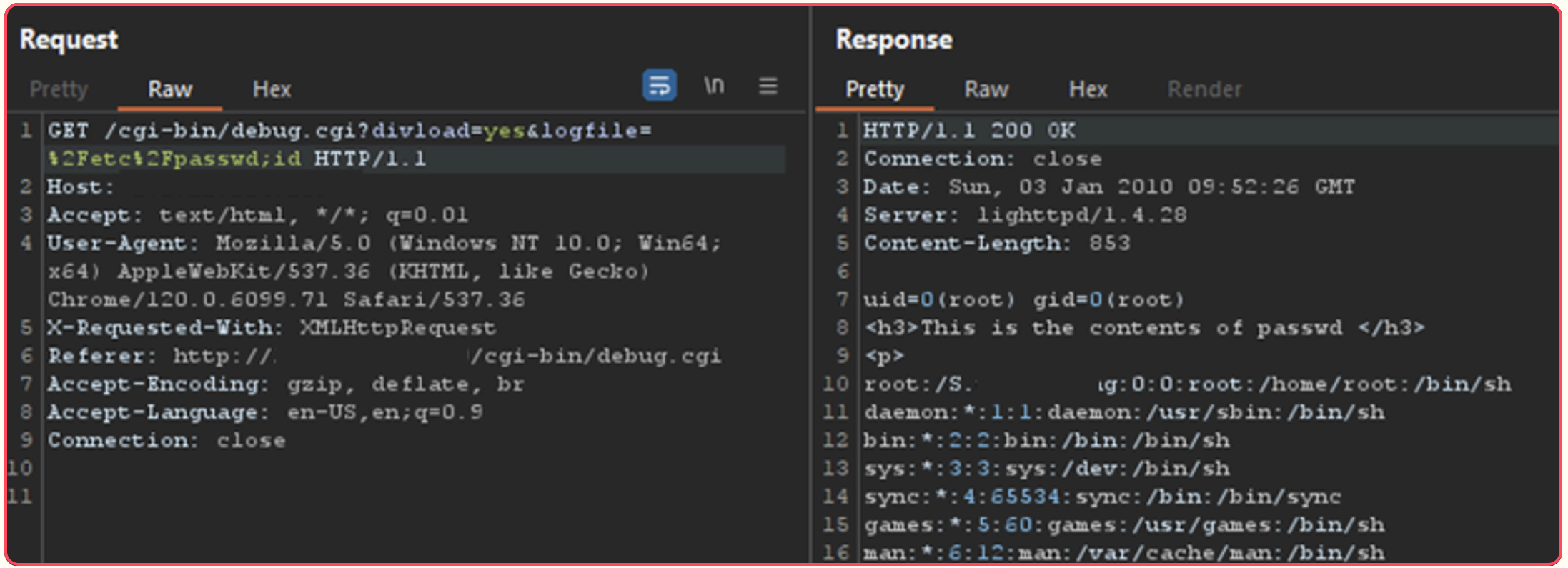
# What is the problem?

Very easy to understand, the **debug cgi-bin endpoint** is **available unauthenticated** and the web request will **accept any system path** (**or command**).



# Lessons learned

**Don't** do that!



# Questions?