Help To Self-Help

How Developers Can Test Their Code Without Being Cyber Security Experts

Who am I?

Principal and Founder at River Security

Certified SANS Instructor

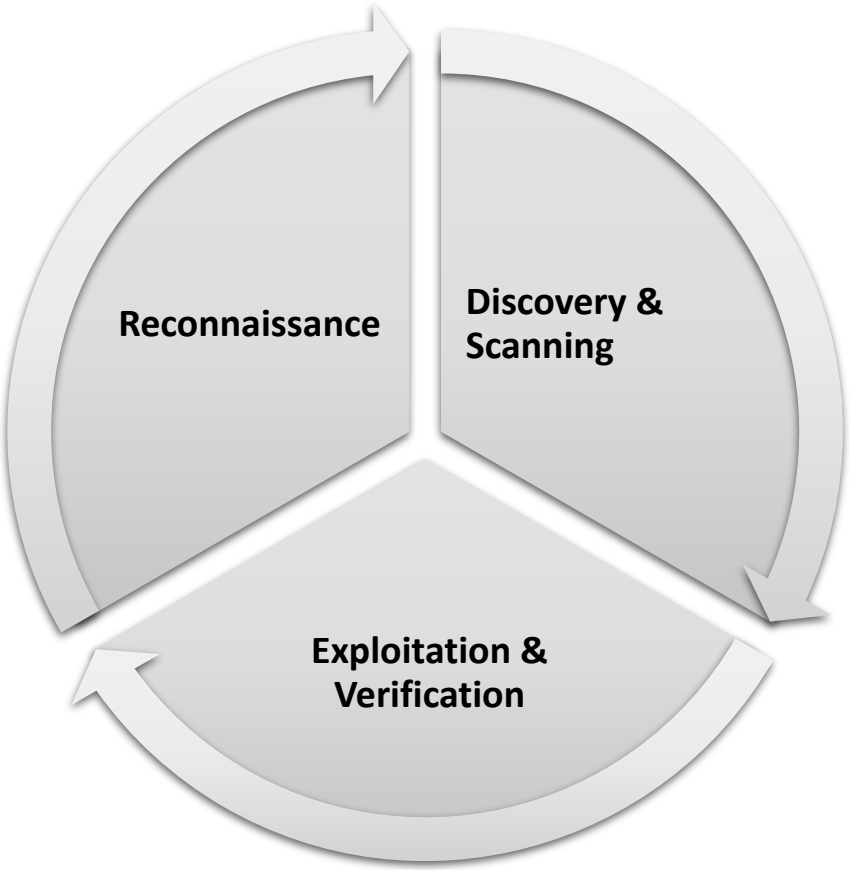
I show how criminals break-in,
and I help throw them back out…

Specialize in Continuous Attack Surface Management and Always-On Penetration Testing

Online presence: https://into.bio/chrisdale

GCIH GIAC Certified Incident Handler
GPEN GIAC Certified Penetration Tester
GSLC GIAC Security Leadership
GIAC Mobile Device Security Analyst
GDAT GIAC Defending Advanced Adversaries
GCTI GIAC Cyber Threat Intelligence
GCFA GIAC Certified Forensic Analyst

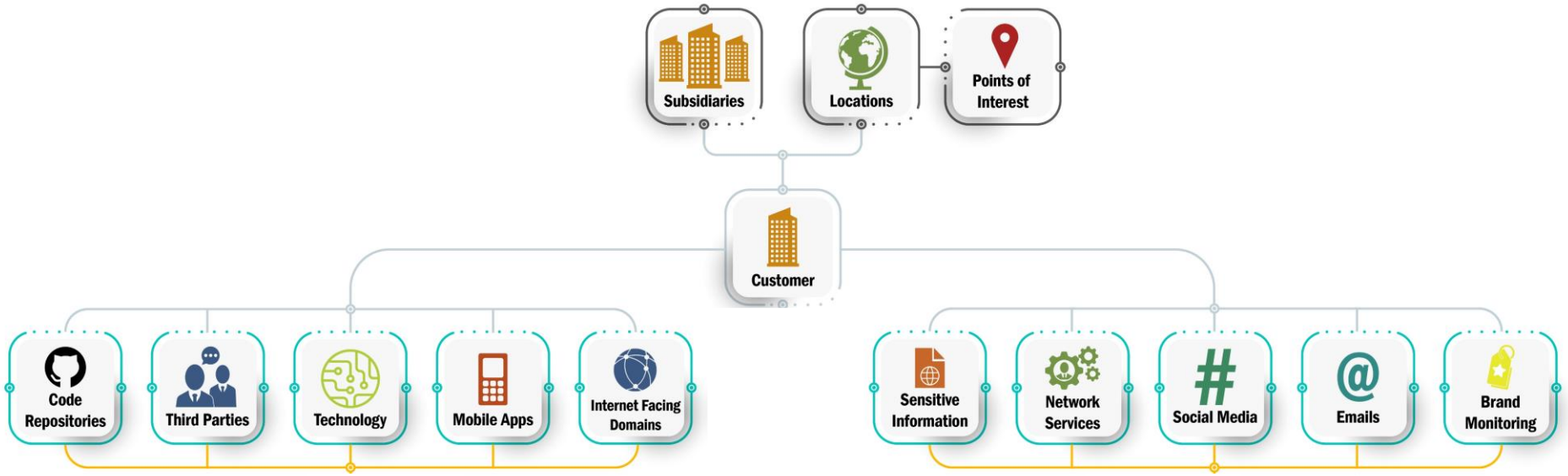
High Level Pentest Methodology



Discovery Process is Essential

- Threat Actors will spend a significant time on reconnaissance, scanning and discovery
- There are opportunities in the current, but developing issues are of uttermost importance
  - New CVE's for existing infrastructure
  - New domains being provisioned, configured and developed
  - New services available, i.e. ports opened
  - Credentials breached
  - Files uploaded to Internet exposed services
  - Changes to repositories, mobile applications and more
  - Deployment of cloud resources

From the outside – threat actors are always-on



Example: MVP of Web Application 1/3

Map browsable and unlinked attack surface

- Browse the entire application, discover all browsable content
- Utilize Content Discovery on all interesting places
- For JavaScript, extract file paths and references
- Discover if application changes based on unlinked parameters
  - Headers, Cookies, GET and POST
- Build that site-map and see if you can get a good grasp on all the application logic available

Example: MVP of Web Application 2/3

Map browsable and unlinked attack surface

- For functionality such as e.g. *?action=showUser&id=123* , try fuzzing the verb with words like:
  - Add, delete, update and so on…
  - Useful wordlists:
    - Server-side variable names
    - Form field values
    - Form Field names
- Browse entire application with Collaborator Everywhere turned on

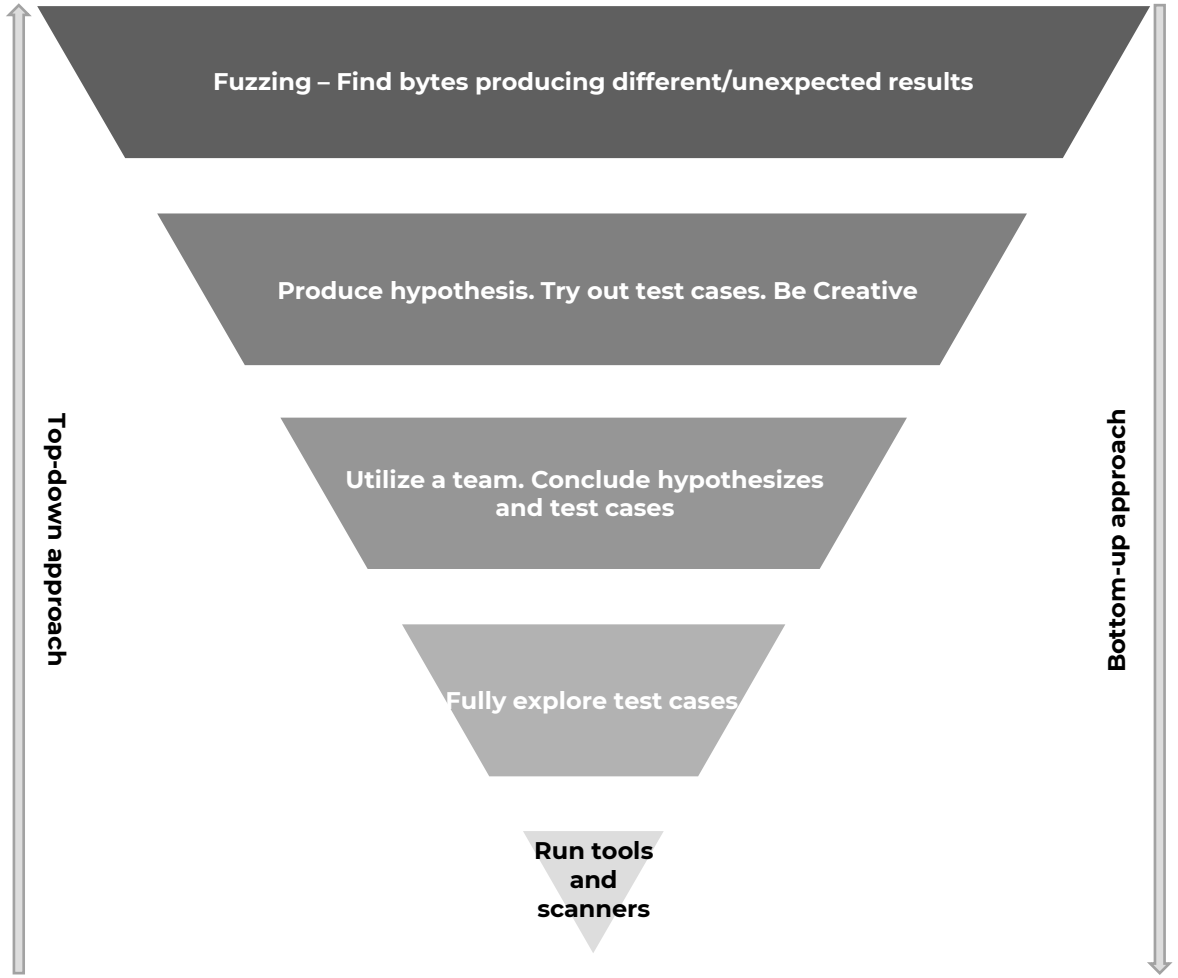
Example: MVP of Web Application 3/3

Utilize your Penetration Testing knowledge to find vulnerabilities

- For-each script and functionality discovered:
  - Determine properly how the functionality works and try related use-cases and hypothesis
  - Send to Fuzzer Tool and define input parameters for Fuzzing
    - Fuzz all bytes %00 through %FF
    - For each discrepancy, add on
  - Send to Vulnerability Scanner and let it do its work.
    - Backslash Powered Scanner and other extensions will also aid here.
  - Finally, back to Intruder and fuzz manually.
    - %00 to %FF is extremely useful

Pentesting Process Pyramid

Fully test the scope on each script and input



Producing High Value Penetration Tests

Reliable and consistent testing is important, and not relying on a single individuals' skills and efforts to complete a penetration test helps ensure the highest levels of standards.

Team Based Effort

Penetration Testing is a team effort, not an individual effort. Utilize a team to maximize the penetration test efforts.

Thoroughly Mapping Attack Surface

Leave no stone untouched. Many vulnerabilities are found in the "paths least travelled". Fully explore!

Reporting

Document findings, process, discrepancies and hypothesis. It will be useful now and later.

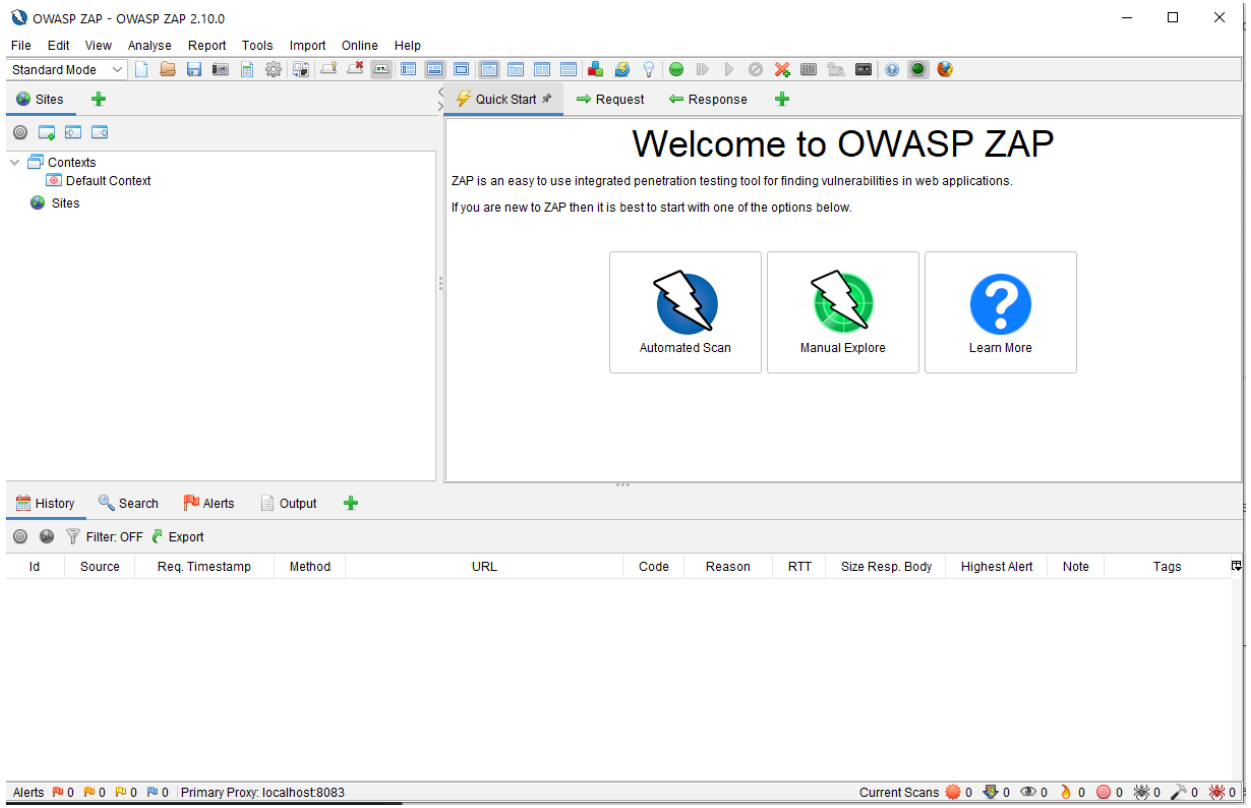
Hypothesis and Knowledge Sharing

A team is stronger. Produce hypothesis to uncover potential attacks across all layers. Strengthen the team knowledge by working as one.

What Can Developers Do?

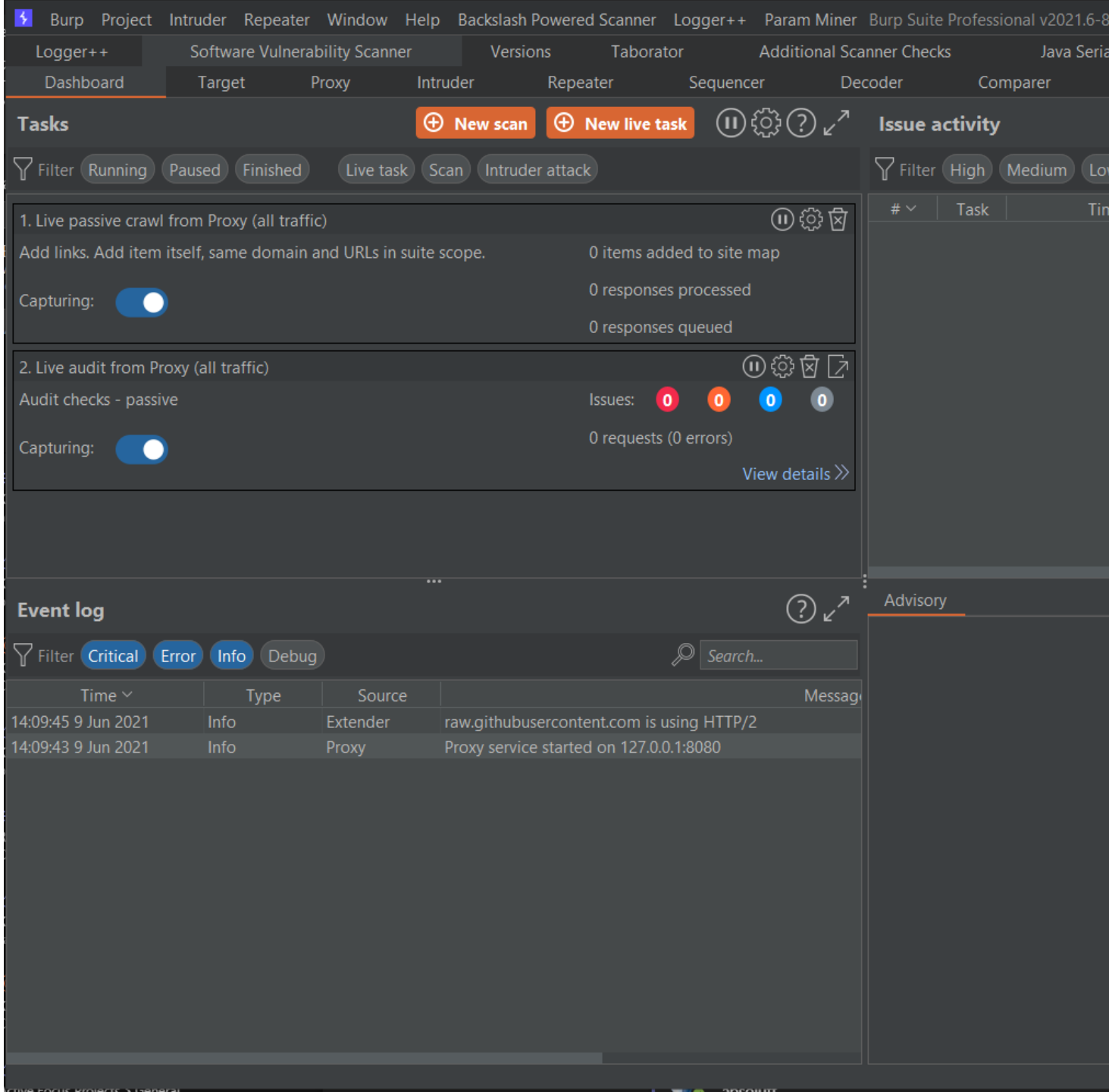
OWASP ZAP

- Nice free Attack Proxy for testing web applications
- Has a nice site-map feature
- Can scan for vulnerabilities
- Allows fuzzing for vulnerabilities
- Chaining of proxies
- WebSocket support
- Good developer support



Burp Suite

- Defacto tool by pentester
- Strong fuzzing capabilities
- Extension support
- Very flexible and robust
- Well developed scanner
- Spidering engine with good SPA support



BurpSuite Free vs Pro

- Paid version has more features, primarily aimed towards penetration testers.
- Vulnerability Scanner which benefits from extensions
- Non-throttled Intruder for fuzzing scripts
- Project and session support
- Handy features such as:
  - Finding script references, scripts and comments
  - Content discovery
  - Reports on attack surface

Navigational Hotkeys

```
Ctrl-Shift-T - Target Tab
Ctrl-Shift-P - Proxy Tab
Ctrl-Shift-R - Repeater Tab
Ctrl-Shift-I - Intruder Tab
Ctrl-Shift-O - Project Options Tab
Ctrl-Shift-D - Dashboard Tab
Ctrl-Equal - next tab
Ctrl-Minus - previous tab
```

Editor Encoding / Decoding Hotkeys

```
Ctrl-B - Base64 selection
Ctrl-Shift-B - Base64 decode selection
Ctrl-H - Replace with HTML Entities
(key characters only)

Ctrl-Shift-H - Replace HTML entities
with characters

Ctrl-U - URL encode selection (key
characters only)
Ctrl-Shift-U - URL decode selection
```

Burp Collaborator

The collaborator enables the penetration tester to listen for call-backs from vulnerable scripts and services via auto-generation of unique DNS names and works on the following protocols:

- DNS
- HTTP & HTTPS
- SMTP & SMTPS

Use the Burp extension Taborator to make Burp Collaborator easier to use on-the-fly.

Global Hotkeys

```
Ctrl-I - Send to Intruder
Ctrl-R - Send to Repeater

Ctrl-S - Search (places cursor in
search field)
Ctrl-. - Go to next selection
Ctrl-m - Go to previous selection

Ctrl-A - Select all
Ctrl-Z - Undo
Ctrl-Y - Redo
```

Editors Hotkeys

```
Ctrl-Delete - Delete Word
Ctrl-D - Delete Line
Ctrl-Backspace - Delete Word Backwards

Ctrl-Home - Go to beginning of document
Ctrl-Shift-Home - Go to beginning of
document and select data on its way
Ctrl-End - Go to end of document
Ctrl-Shift-End - Go to end of document
and select data on its way
Ctrl-Left - Go to Previous Word
Ctrl-Shift-Left - Go to Previous Word
and select data on its way
Ctrl-Right - Go to Next Word
Ctrl-Shift-Right - Go to Next Word and
select data on its way
```

Tool Specific Hotkeys

```
Ctrl-F - Forward Request (Proxy)
Ctrl-T - Toggle Proxy Intercept On and
Off
Ctrl-Space - Send Request (Repeater)

Double-click <TAB> - Rename a tab
```

OFFENSIVE OPERATIONS

Burp Suite
Cheat Sheet v1.0

By Chris Dale @chrisadale

SANS sans.org/offensive-operations

Purpose

This cheat sheet enables users of Burp Suite with quicker operations and more ease of use.
Burp Suite is the de-facto penetration testing tool for assessing web applications. It enables penetration testers to rapidly test applications via signature features like repeater, intruder, sequencer, and extender.

It is split into two pages, one page containing common shortcuts to use within the application, the second page containing useful extensions and tips-and-tricks. It is recommended to manually check and test the different extensions available in the product; many which may be very useful to your testing, but outside of what this cheat sheet can cover.

Burp Suite comes in a free community edition and a commercial professional edition. It has a built in Chromium browser for easy set-up of HTTP and SSL/TLS interception.

POCKET REFERENCE GUIDE

A Dedicated Browser For Testing

- You should have a “Hack Naked” browser
  - I’m using Firefox Developer Edition for testing and Chrome for browsing
- Configure it with useful proxies, and keep your web-surfing and “googling” to other browsers
  - Eventually use a proxy like SwitchySharp to control which domains are included or excluded from proxying
- My browser has been configured so that I control it to maximum extent

Using ZAP to fuzz

1

OWASP ZAP - OWASP ZAP 2.10.0

File Edit View Analyse Report Tools Import Online Help

Standard Mode

Sites | Quick Start | Request | Response

Header: Text | Body: Text

```
GET http://attack:60101/vulnerabilities/xss_r/?name=test HTTP/1.1
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64;
Accept: text/html,application/xhtml+xml,application/xml
Accept-Language: en-GB,en;q=0.5
Accept-Encoding: gzip, deflate
Connection: keep-alive
Referer: http://attack:60101/vulnerabilities/xss_r/
Cookie: PHPSESSID=b1cd1oloknd ehotjqirkp0m2u4; security=
Upgrade-Insecure-Requests: 1
Host: attack:60101
```

- Contexts
  - Default Context
- Sites
  - http://attack
  - http://attack:60101
  - http://cdn.matomo.cloud
  - http://portainer
  - http://portainer:9000
  - http://localhost:9000
  - http://127.0.0.1:60101
  - http://127.0.0.1:60111
  - http://127.0.0.1:9000
  - http://127.0.0.1:62101
  - http://127.0.0.1:62100
  - http://neverssl.com

Find... Ctrl-F
Open/Resend with Request Editor...
Fuzz...
Run application
Open URL in System Browser
Invoke with Script...
Encode/Decode/Hash...
Open URL in Browser
Syntax
View
Can't Undo Ctrl-Z
Can't Redo Ctrl-Y
Cut Ctrl-X
Copy Ctrl-C
Paste Ctrl-V
Delete Delete
Select All Ctrl-A
Save Raw
Save XML

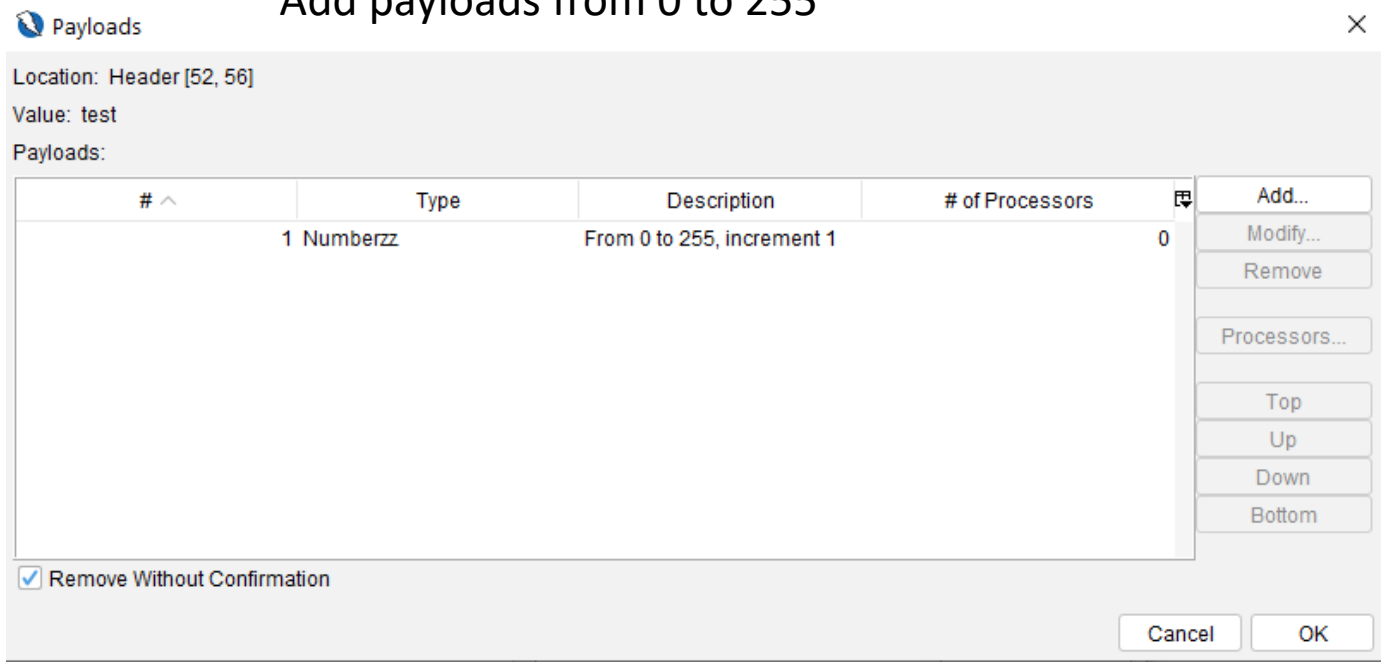
History | Search | Alerts | Output

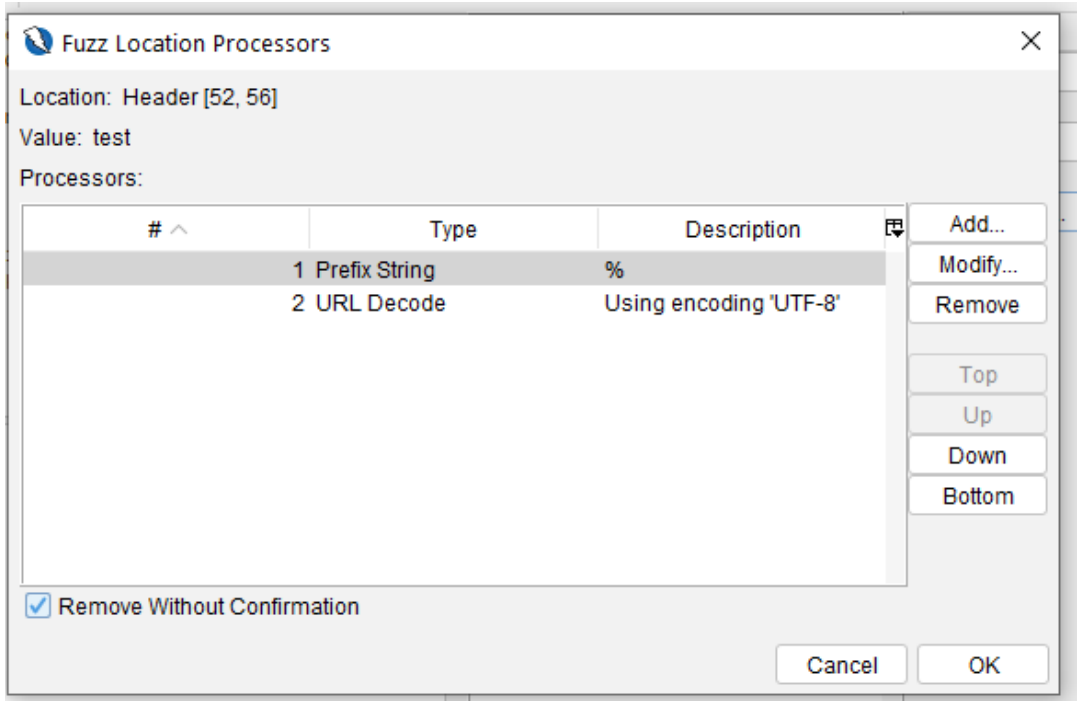
Filter: OFF | Export

Id	Source	Req. Timestamp	Method	URL	Code	Reason	RTT			Tags
82	Proxy	6/9/21, 4:09:34 PM	GET	http://attack:60101/security.php	200	OK	68 ms	5,		Form, Hidden, Script
84	Proxy	6/9/21, 4:09:36 PM	GET	http://attack:60101/vulnerabilities/csrf/	200	OK	38 ms	5,		Form, Password, Hi...
86	Proxy	6/9/21, 4:09:37 PM	GET	http://attack:60101/vulnerabilities/fi/?page=inclu...	200	OK	39 ms	4,		Script
88	Proxy	6/9/21, 4:09:38 PM	GET	http://attack:60101/vulnerabilities/csrf/	200	OK	38 ms	5,		Form, Password, Hi...
89	Proxy	6/9/21, 4:09:42 PM	GET	http://attack:60101/vulnerabilities/fi/?page=inclu...	200	OK	66 ms	4,414 bytes	Medium	Script

Fuzzing it – All Available Bytes

Add payloads from 0 to 255

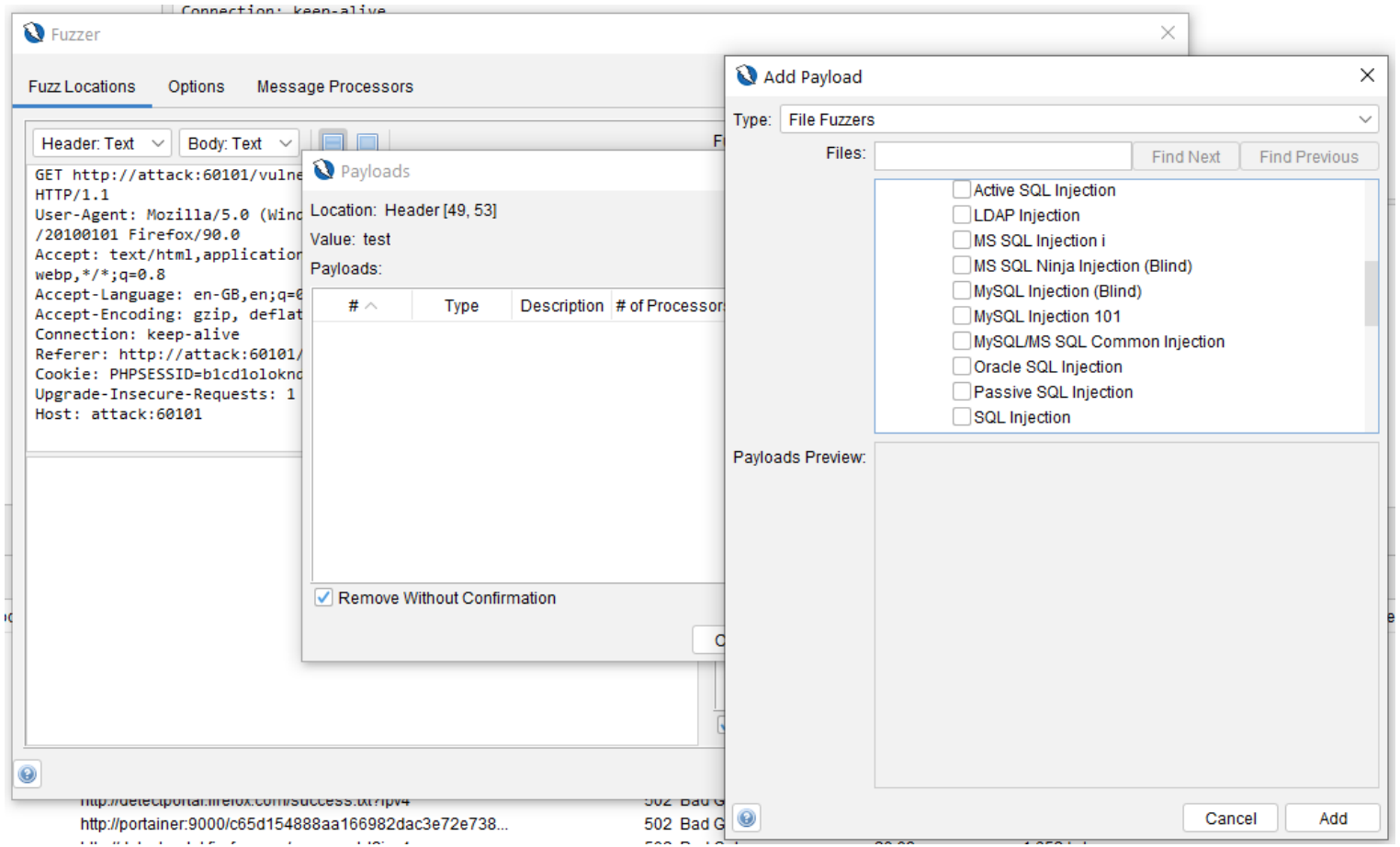




2

3

Fuzzing it – with wordlists



Fuzzing with Burp Suite

1

Target | Positions | Payloads | Resource Pool | Options

Payload Positions

Start attack

Configure the positions where payloads will be inserted into the base request. The attack type determines the way in which payloads are assigned to payload positions - see help for full details.

Attack type: Sniper

```
1 POST /example?p1=$p1val$&p2=p2val HTTP/1.0
2 Cookie: c=cval
3 Content-Length: 17
4
5 p3=p3val&p4=p4val
```

Add §
Clear §
Auto §
Refresh

2

Payload Sets

You can define one or more payload sets. The number of payload sets depends on the atta
set, and each payload type can be customized in different ways.

Payload set: 1 — Payload count: 0
Payload type: Numbers — Request count: 0

Payload Options [Numbers]

This payload type generates numeric payloads within a given range and in a specified forn

Number range

Type: Sequential / Random
From: 00
To: FF
Step:
How many:

Number format

Base: Decimal / Hex
Min integer digits: 2
Max integer digits: 2
Min fraction digits:
Max fraction digits:

3

Payload Processing

You can define rules to perform various processing tasks on each payload before it is used.

Add | Edit | Remove | Up | Down

Enabled	Rule
✓	Add Prefix: %
☐	URL-decode

URL-Decode should be tested on and off

Follow-Up on Discrepancies and Anomalies

Attack Save Columns — 13. Intruder attack of 127.0.0.1 - Temporary attack - Not saved to project file

Results Target Positions Payloads Resource Pool Options

Filter: Showing all items

Request	Payload	Status	Error	Timeout	Length	Comment
40	%27	200			463	
93	%5c	200			463	
0		200			4770	
1	%00	200			4770	
2	%01	200			4770	
3	%02	200			4770	
4	%03	200			4770	
5	%04	200			4770	
6	%05	200			4770	
7	%06	200			4770	
8	%07	200			4770	
9	%08	200			4770	
10	%09	200			4770	

Request Response

Pretty Raw Hex Render

```
1  HTTP/1.1 200 OK
2  Date: Tue, 07 Dec 2021 08:12:54 GMT
3  Server: Apache/2.4.25 (Debian)
4  Expires: Thu, 19 Nov 1981 08:52:00 GMT
5  Cache-Control: no-store, no-cache, must-revalidate
6  Pragma: no-cache
7  Vary: Accept-Encoding
8  Content-Length: 162
9  Connection: close
10 Content-Type: text/html; charset=UTF-8
11
12 <pre>
     You have an error in your SQL syntax; check the manual that corresponds to your MariaDB server version for the right syntax to use near ''\'' at line 1
   </pre>
```

Follow-Up on Discrepancies and Anomalies

Request	Payload	Status	Error	Timeout	Length	Comment
78	%4d					
49	%30	200			4872	
50	%31	200			4587	
51	%32	200			4587	
52	%33	200			4587	
53	%34	200			4587	
54	%35	200			4587	
55	%36	200			4587	
56	%37	200			4587	
57	%38	200			4587	
0		200			4469	
1	%00	200			4469	
2	%01	200			4469	

Request Response

Pretty Raw Hex Render

DVWA

Home
Instructions
Setup / Reset DB

Brute Force
Command Injection
CSRF
File Inclusion
File Upload
Insecure CAPTCHA

Vulnerability: Command Injection

Ping a device

Enter an IP address: Submit

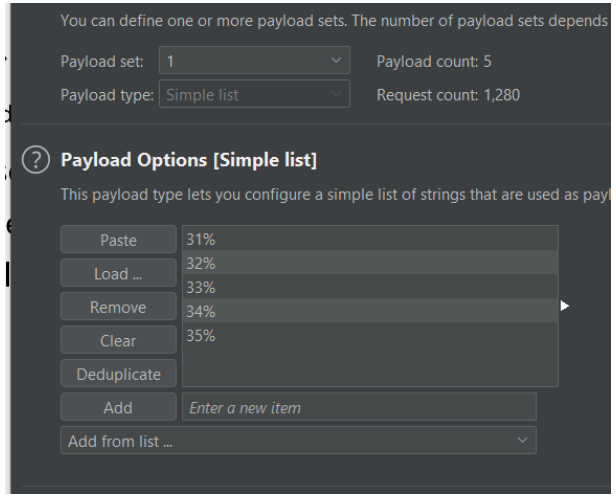
```
PING 0 (0.0.0.0): 56 data bytes
64 bytes from 127.0.0.1: icmp_seq=0 ttl=64 time=0.128 ms
64 bytes from 127.0.0.1: icmp_seq=1 ttl=64 time=0.064 ms
64 bytes from 127.0.0.1: icmp_seq=2 ttl=64 time=0.040 ms
64 bytes from 127.0.0.1: icmp_seq=3 ttl=64 time=0.066 ms
--- 0 ping statistics ---
4 packets transmitted, 4 packets received, 0% packet loss
round-trip min/avg/max/stddev = 0.040/0.075/0.128/0.033 ms
```

Optionally: For Hunting Vulnerabilities

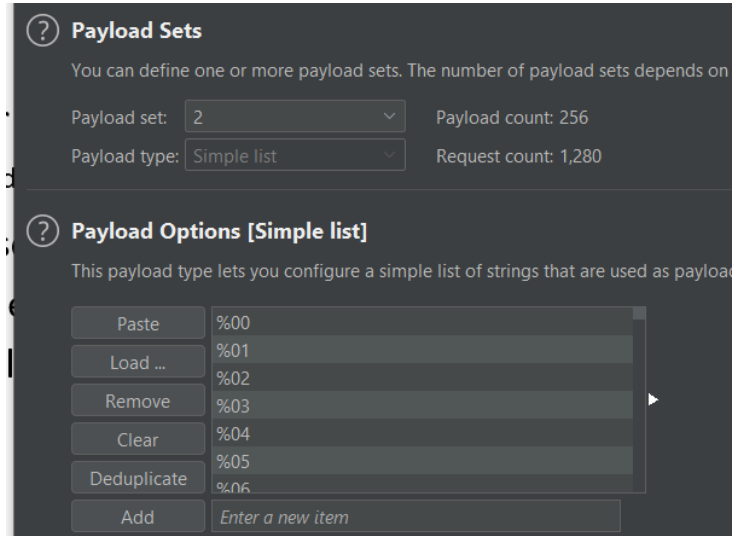
Instead of just patching it right away

- For the anomalies discovered, could we not sanitize them right away?
- If URL-Decode is turned on, typically the middle-ware is targeted
- Can method detect Boolean or Blind attacks?

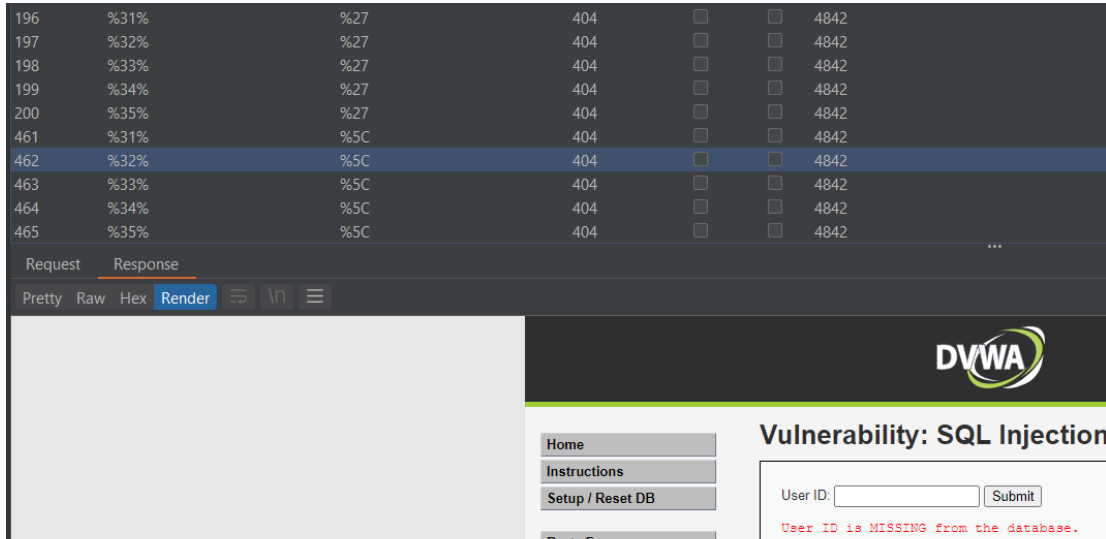
1



2



3



Thank You For Your Attention!

Twitter – https://twitter.com/ChrisADale

LinkedIn – https://www.linkedin.com/in/chrisad/

YouTube – https://www.youtube.com/c/chrisdale

SANS Profile – https://www.sans.org/profiles/chris-dale/

River Security – https://riversecurity.eu