(5th November 2025) - JAGUAR COHORT
Made by Chris Cownden
I'm creating recap guides as revision notes for future use. Please use this document as you see fit.

5th Lab Recap Guide: Security+ (Bug Bounty)

Recording is here , Transcript is here

As this was a different style of Lab, I've included a section called "Jon's Thought Process For Web Application Hacking" that outlines WHAT he does, what the CONCEPT is and WHY he is doing it.

Timestamps

00:00 HackerOne Bug Bounty Platform

32:04 Access Control Exploitation Insight

36:02 Image Upload and Application Processing

40:30 "Exploiting Image Upload Flaws"

51:18 "HTML, JavaScript, and Web Pages"

01:06:14 Streamlining Payload Reviews

01:11:17 Burp Suite Repeater Usage Explained

01:23:44 SVG XSS Payload Testing

01:35:14 JavaScript Error and Alert Explained

01:41:42 "Experimenting With SVG Tweaks"

Recommended Resources

- HackerOne - Sign up for free to get paid to find Bug Bounties
- Google Authentication App
- Burp Suite - Download Community version for free
- OWASP Juice Shop (Fake E-commerce Site to practice finding bugs
- ZAP - Web App Scanner
- Claude - Alternative to ChatGPT
- Jon's Slides - These slides show you how to use Burp Suite and other tools
- Jon's email for 1-to-1 support jon@optima-it.co.uk

(5th November 2025) - JAGUAR COHORT
Made by Chris Cownden
I'm creating recap guides as revision notes for future use. Please use this document as you see fit.

Other Recordings To Date

Date	Order	Recordings	Lab Recap Guides
5th November 2025	Lab 5	RECORDING	THIS ONE
2nd November 2025	3rd Sunday	RECORDING	
29th October 2025	Lab 4	RECORDING	RECAP GUIDE
22nd October 2025	Lab 3	RECORDING	RECAP GUIDE
19th October 2025	2nd Sunday	RECORDING	
15th October 2025	Lab 2	RECORDING	RECAP GUIDE
8th October 2025	Lab 1	RECORDING	RECAP GUIDE
5th October 2025	1st Sunday	RECORDING	

Core Concepts

What is a Bug Bounty?

A bug bounty is a program where organisations offer monetary rewards (or prizes) to individuals who discover and report security vulnerabilities in their software or web applications. Platforms like HackerOne facilitate this process.

Why You Need to Know:
Bug bounty programs are now mainstream in cybersecurity, and knowing how they operate, their purpose, and the terminology used around them is crucial for both real-world pen testing and the Security+ exam.

Web Application Mapping

- "Mapping the app" means exploring all functionality—login forms, file uploads, user flows—to identify attack surfaces.

(5th November 2025) - JAGUAR COHORT
Made by Chris Cownden
I'm creating recap guides as revision notes for future use. Please use this document as you see fit.

File Upload Testing

- Uploading unexpected file types (e.g., .html, .svg, .jpeg) to probe server-side validation/sanitization.
- Double file extensions (.jpg.html) might bypass weak filters.
- SVG files can sometimes hold embedded JavaScript payloads (common vector for XSS).

Cross Site Scripting (XSS)

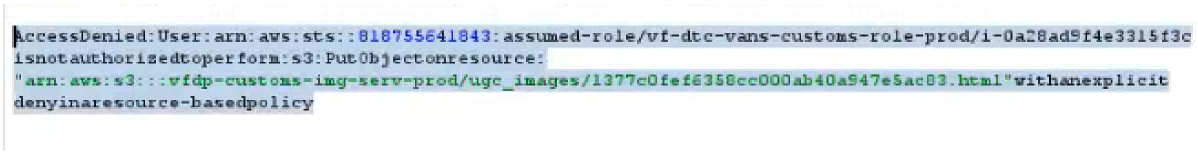
- **What is it?**
  Injecting arbitrary JavaScript/HTML into a site, causing it to execute in another user's browser.
- **Testing:** Use payloads like `<script>alert(1)</script>` or image tag onerror handlers: `<img src=x onerror=alert(1)>`.
- **Stored vs. Reflected:**
  - *Stored:* Payload is saved and shown to others.
  - *Reflected:* Payload is only echoed back to the submitter.

Cookies & Trackers

- Modern web apps generate huge amounts of background requests, cookies and third-party trackers.
- Some can be injection points for attacks or sources of sensitive info.

Information Disclosure

- Sometimes you can access images or resources from direct URLs.
- Leaking internal info (such as AWS account numbers or internal paths) is a reportable issue like the example below. "arn: aws: sts: number"
- Ask ChatGPT *"what are the negative security consequences of leaking internal info (such as AWS account numbers or internal paths) "arn: aws: sts: number"?* - There are LOTS!!!


```
AccessDenied:User:arn:aws:sts::818755641843:assumed-role/vf-dtc-vans-customs-role-prod/i-0a28ad9f4e3315f3c
isnotauthorizedtoperform:s3:PutObjectonresource:
"arn:aws:s3:::vfdp-customs-img-serv-prod/ugc_images/1377c0fef6358cc000ab40a947e5ac83.html"withanexplicit
denyinaresource-basedpolicy
```

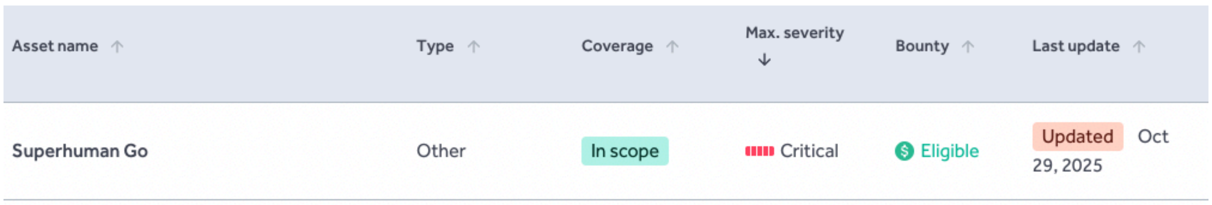
Jon's Thought Process For Web Application Hacking

I've included Jon's thought process for approaching a particular part of the Bug Bounty Exercise with Van's Custom Shoe Feature. It will include what he did, what the concept is

(5th November 2025) - JAGUAR COHORT
Made by Chris Cownden
I'm creating recap guides as revision notes for future use. Please use this document as you see fit.

and why he did each step. This should help you better understand the why and how behind the steps.

Choosing a Target inside HackerOne and Explaining Scope



Asset name	Type	Coverage	Max. severity	Bounty	Last update
Superhuman Go	Other	In scope	Critical	Eligible	Updated Oct 29, 2025

- **WHAT**
    - Jon picks a program on HackerOne (e.g., VF Corporation) and reads the program's scope, policy, and disclosure rules.
- **CONCEPT**
    - Bug-bounty programs publish a **scope**: which domains, IPs, APIs, mobile apps and features you *may* test, and what's explicitly *out of scope*.
    - Jon also defines rules (safe testing, disclosure timelines, allowed data handling).
- **WHY**
    - **Benefit:** Keeps testing legal and focused where the company wants help.
    - **Problem:** Testing out-of-scope targets can be criminal.
    - **Safe solution:** Confirm scope in writing; if in doubt, ask the program triage contact before testing.
- **Quick tip:** Save the program rules and a screenshot showing the scope before you start.

Mapping Functionality

- **WHAT**
    - Jon browses the site: signs up, uses features (Van's custom shoe builder), finds points where the app accepts user input or files.
- **CONCEPT**
    - This is **ENUMERATION** which means mapping routes, forms, upload endpoints, API calls and UI flows to find where inputs enter the system.

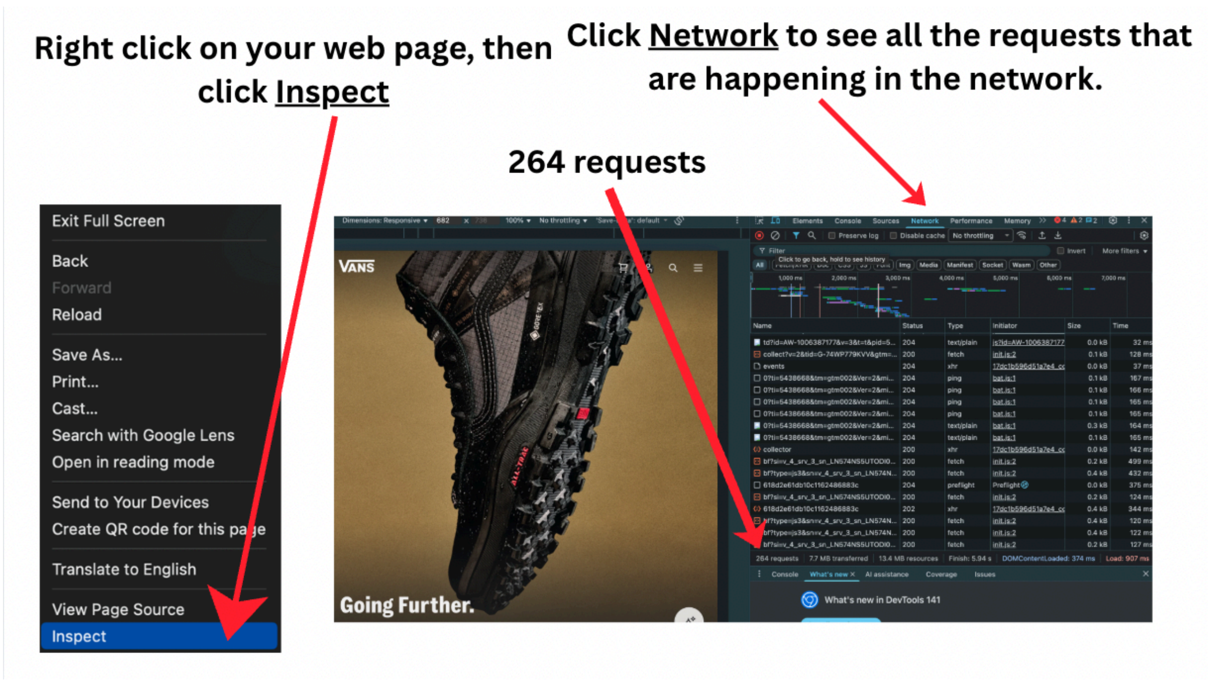
(5th November 2025) - JAGUAR COHORT
Made by Chris Cownden
I'm creating recap guides as revision notes for future use. Please use this document as you see fit.

   - **INPUTS** = common places for bugs (forms, uploads, submitting reviews, query parameters, profile fields).

- **WHY**
   - **Benefit:** Helps prioritise likely weak spots (file uploads, profile fields).
   - **Problem:** Missing an input vector wastes time.
   - **Safe solution:** Make a simple map (screenshot + URL + expected input type) and mark which parts accept files or user text.

- **Quick tip:** Keep your mapping non-destructive, don't try risky actions until you're sure they're allowed. Use a tool like Burp Suite to doing this.

Uploading a File (Image)



- **WHAT**
   - Jon right clicks on his screen and clicks Inspect, and then clicks Network
   - Jon uploads an image (cat pic) and watches how the site responds and where the file appears.

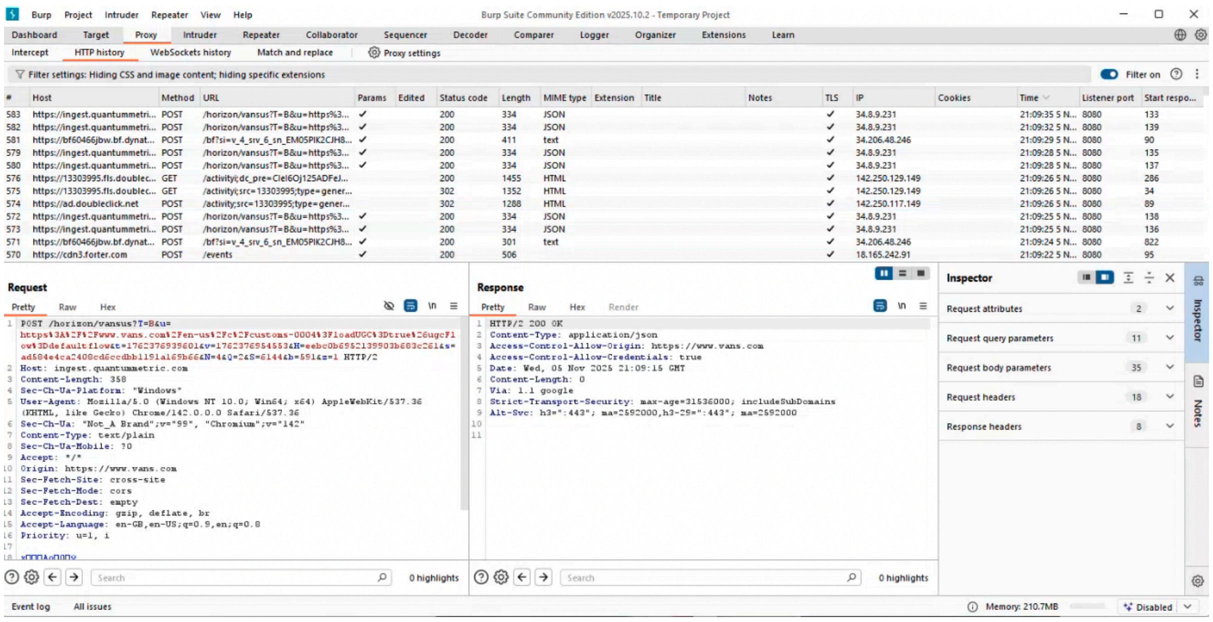
- **CONCEPT**
   - File uploads test how the server validates files, where it stores them, and how it serves them back to users.
   - Important checks: file type checks, storage location, metadata handling, and rendering behaviour.

- **WHY**

- **Benefit:** Reveals storage and rendering policies (public vs private, direct URL vs proxied).
- **Problem:** Poor handling can expose private files or let malicious content be served.
- **Safe solution:** Use benign test files and document responses; don't try to upload known-malicious content.

- **Quick tip:** Note the uploaded file's returned URL, response headers and whether the file is shown inline or as a download.

Inspecting Network Traffic and Application Behaviour


Burp Project Intruder Repeater View Help
Burp Suite Community Edition v2025.10.2 - Temporary Project
Dashboard Target Proxy Intruder Repeater Collaborator Sequencer Decoder Comparer Logger Organizer Extensions Learn
Intercept HTTP history WebSockets history Match and replace Proxy settings
Filter on
Event log All issues
Memory: 210.7MB
Disabled

Filter settings: Hiding CSS and image content; hiding specific extensions

#	Host	Method	URL	Params	Edited	Status code	Length	MIME type	Extension	Title	Notes	TLS	IP	Cookies	Time	Listener port	Start respo...
583	https://ingest.quantummetri...	POST	/horizon/vansus?T=B&u=https%3...	✓		200	334	JSON				✓	34.8.9.231		21:09:35 5 N...	8080	133
582	https://ingest.quantummetri...	POST	/horizon/vansus?T=B&u=https%3...	✓		200	334	JSON				✓	34.8.9.231		21:09:32 5 N...	8080	139
581	https://bf60466jbw.bf.dynat...	POST	/bf?si=v_4_srv_6_sn_EM05PIK2CJH8...	✓		200	411	text				✓	34.206.48.246		21:09:29 5 N...	8080	90
579	https://ingest.quantummetri...	POST	/horizon/vansus?T=B&u=https%3...	✓		200	334	JSON				✓	34.8.9.231		21:09:28 5 N...	8080	135
580	https://ingest.quantummetri...	POST	/horizon/vansus?T=B&u=https%3...	✓		200	334	JSON				✓	34.8.9.231		21:09:28 5 N...	8080	137
576	https://13303995.fls.doublec...	GET	/activityi;dc_pre=Clel6Oj125ADFeJ...			200	1455	HTML				✓	142.250.129.149		21:09:26 5 N...	8080	286
575	https://13303995.fls.doublec...	GET	/activityi;src=13303995;type=gener...			302	1352	HTML				✓	142.250.129.149		21:09:26 5 N...	8080	34
574	https://ad.doubleclick.net	POST	/activity;src=13303995;type=gener...			302	1288	HTML				✓	142.250.117.149		21:09:26 5 N...	8080	89
572	https://ingest.quantummetri...	POST	/horizon/vansus?T=B&u=https%3...	✓		200	334	JSON				✓	34.8.9.231		21:09:25 5 N...	8080	138
573	https://ingest.quantummetri...	POST	/horizon/vansus?T=B&u=https%3...	✓		200	334	JSON				✓	34.8.9.231		21:09:25 5 N...	8080	136
571	https://bf60466jbw.bf.dynat...	POST	/bf?si=v_4_srv_6_sn_EM05PIK2CJH8...	✓		200	301	text				✓	34.206.48.246		21:09:24 5 N...	8080	822
570	https://cdn3.forter.com	POST	/events	✓		200	506					✓	18.165.242.91		21:09:22 5 N...	8080	95

Request (Pretty Raw Hex)

```
POST /horizon/vansus?T=B&u=
https%3A%2F%2Fwww.vans.com%2Fen-us%2Fc%2Fcustoms-0004%3FloadUGC%3Dtrue%26ugcFl
ow%3Ddefaultflow&t=1762376939601&v=1762376954553&H=eebc0b6952139903b683c261&s=
ad584e4ca2408cd6ccdbb1191a169b66&N=4&Q=2&S=6144&b=591&z=1 HTTP/2
Host: ingest.quantummetric.com
Content-Length: 358
Sec-Ch-Ua-Platform: "Windows"
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36
(KHTML, like Gecko) Chrome/142.0.0.0 Safari/537.36
Sec-Ch-Ua: "Not_A Brand";v="99", "Chromium";v="142"
Content-Type: text/plain
Sec-Ch-Ua-Mobile: ?0
Accept: */*
Origin: https://www.vans.com
Sec-Fetch-Site: cross-site
Sec-Fetch-Mode: cors
Sec-Fetch-Dest: empty
Accept-Encoding: gzip, deflate, br
Accept-Language: en-GB,en-US;q=0.9,en;q=0.8
Priority: u=1, i

x[illegible]
```

Search 0 highlights

Response (Pretty Raw Hex Render)

```
HTTP/2 200 OK
Content-Type: application/json
Access-Control-Allow-Origin: https://www.vans.com
Access-Control-Allow-Credentials: true
Date: Wed, 05 Nov 2025 21:09:15 GMT
Content-Length: 0
Via: 1.1 google
Strict-Transport-Security: max-age=31536000; includeSubDomains
Alt-Svc: h3=":443"; ma=2592000,h3-29=":443"; ma=2592000
```

Search 0 highlights

Inspector

Request attributes	2
Request query parameters	11
Request body parameters	35
Request headers	18
Response headers	8

- **WHAT**
  - Jon opens browser dev tools (Network tab) and Burp Suite to observe requests when he uploads or interacts.

- **CONCEPT**
  - Browser dev tools show HTTP requests/responses, headers, payload sizes and timing revealing how the client and server communicate.
  - Look specifically for POST endpoints, response codes, cookies, auth headers, and returned URLs.

- **WHY**
  - **Benefit:** Reveals how data is sent and what the server expects.
  - **Problem:** Misconfigured headers or predictable tokens can be abused.

  - **Safe solution:** Record the requests you observe; don’t alter live traffic without authorization (use a test account or staging if available).

- **Quick tip:** Save HAR files or screenshots for your notes, they’re useful for reports.

Locating the Uploaded File on Server

- **WHAT**
  - Jon finds that the response contains a URL pointing to an S3 bucket

- **CONCEPT**
  - Many apps store uploaded files in cloud object storage; sometimes files are public, sometimes private and served via signed URLs.
  - Public object storage with predictable naming can leak content.

- **WHY**
  - **Benefit:** Finding storage type shows exposure level (public S3 vs private proxied storage).
  - **Problem:** Public buckets or predictable URLs can reveal sensitive content or allow enumeration.
    Safe solution: Note whether URLs are publicly accessible, whether listing is allowed, and whether URLs look signed/time-limited.

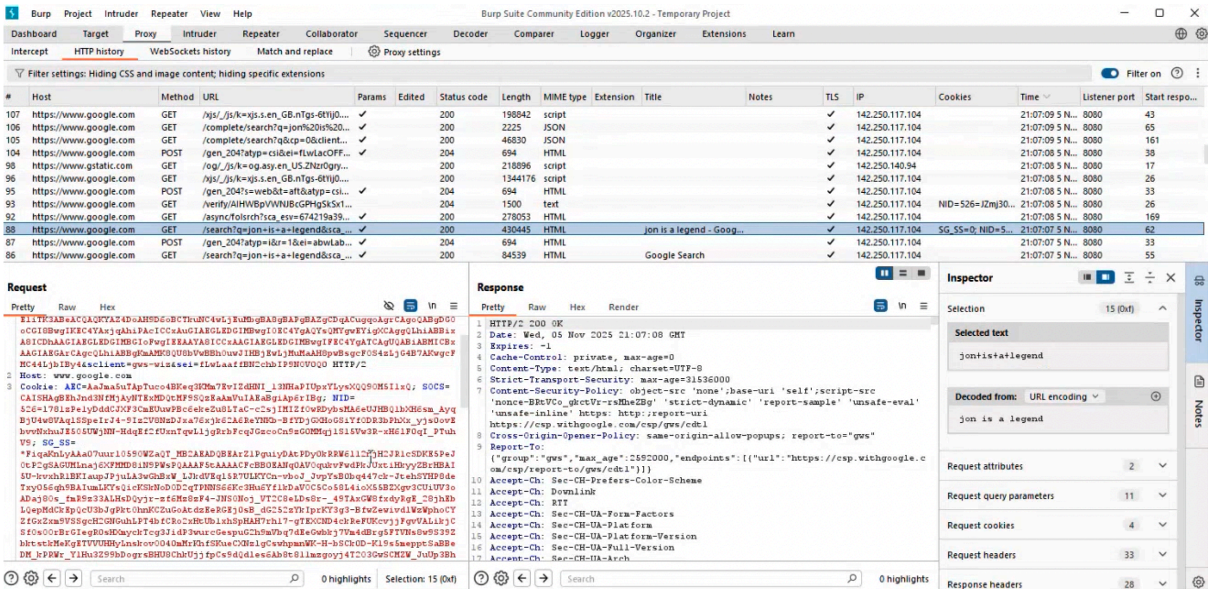
- **Quick tip:** Don’t try to brute-force object keys. Document what you can see and report it.

Manipulating Requests with a Web Proxy

(5th November 2025) - JAGUAR COHORT

Made by Chris Cownden

I'm creating recap guides as revision notes for future use. Please use this document as you see fit.



Filter settings: Hiding CSS and image content; hiding specific extensions

#	Host	Method	URL	Params	Edited	Status code	Length	MIME type	Extension	Title	Notes	TLS	IP	Cookies	Time	Listener port	Start respo...
107	https://www.google.com	GET	/xjs/_/js/k=xjs.s.en_GB.nTgs-6tYij0....	✓		200	198842	script				✓	142.250.117.104		21:07:09 5 N...	8080	43
106	https://www.google.com	GET	/complete/search?q=jon%20is%20...	✓		200	2225	JSON				✓	142.250.117.104		21:07:09 5 N...	8080	65
105	https://www.google.com	GET	/complete/search?q&cp=0&client...	✓		200	46830	JSON				✓	142.250.117.104		21:07:09 5 N...	8080	161
104	https://www.google.com	POST	/gen_204?atyp=csi&ei=fLwLacOFF...	✓		204	694	HTML				✓	142.250.117.104		21:07:08 5 N...	8080	38
98	https://www.gstatic.com	GET	/og/_/js/k=og.asy.en_US.ZNzr0gry...			200	218896	script				✓	142.250.140.94		21:07:08 5 N...	8080	17
96	https://www.google.com	GET	/xjs/_/js/k=xjs.s.en_GB.nTgs-6tYij0....			200	1344176	script				✓	142.250.117.104		21:07:08 5 N...	8080	26
95	https://www.google.com	POST	/gen_204?s=web&t=aft&atyp=csi...	✓		204	694	HTML				✓	142.250.117.104		21:07:08 5 N...	8080	33
93	https://www.google.com	GET	/verify/AIHWBpVWNJBcGPHgSkSx1...			204	1500	text				✓	142.250.117.104	NID=526=JZmj30...	21:07:08 5 N...	8080	26
92	https://www.google.com	GET	/async/folsrch?sca_esv=674219a39...	✓		200	278053	HTML				✓	142.250.117.104		21:07:08 5 N...	8080	169
88	https://www.google.com	GET	/search?q=jon+is+a+legend&sca_...	✓		200	430445	HTML		jon is a legend - Goog...		✓	142.250.117.104	SG_SS=0; NID=5...	21:07:07 5 N...	8080	62
87	https://www.google.com	POST	/gen_204?atyp=i&r=1&ei=abwLab...	✓		204	694	HTML				✓	142.250.117.104		21:07:07 5 N...	8080	33
86	https://www.google.com	GET	/search?q=jon+is+a+legend&sca_...	✓		200	84539	HTML		Google Search		✓	142.250.117.104		21:07:07 5 N...	8080	55

Request

```
EliTK3ABeACQAQKYAZ4DoAH9D6oBCTkuNC4wLjEuMbgBA8gBAPgBAZgCDqACuggoAgrCAgoQABgDG0
oCGI8BwgIKEC4YAxjqAhiPAcICCxAuGIAEGLEDGIMBwgI0EC4YgAQYsQMYgwEYigXCAggQLhiABBix
A8ICDhAAGIAEGLEDGIMBGIoFwgIEEAAYA8ICCxAAGIAEGLEDGIMBwgIFEC4YgATCAgUQABiABMICBx
AAGIAEGArCAgcQLhiABBgKmAMK8QU8bVwBBh0uwJIHBjEwLjMuMaAH8pwBsgcFOS4zLjG4B7AKwgcF
MC44LjbIBy4&sclient=gws-wiz&sei=fLwLaaffBN2chbIP9N0VOQ0 HTTP/2
Host: www.google.com
Cookie: AEC=AaJma5uTApTuco4BKeq3KMm7EvIZdHNI_13NHaPIUpxYLysXQQ9OM5IlxQ; SOCS=
CAISHAgBEhJnd3NfMjAyNTExMDQtMF9SQzEaAmVuIAEaBgiAp6rIBg; NID=
526=178lzPelyDddCJXF3CmEUuwPBc6ekeZu8LTaC-c2sjIMIZf0wRDybsMA6eUJHBQlbXH6sm_Ayq
BjU4w8VAqlSSpeIrJ4-9Iz2V8NzDJxa76xjk62A6ReYNKb-BfYDjGXHoGSiYf0DR3bPhXx_yjs0ovE
bvvNxhuJE505UWjNN-HdqEf2fUxnTqwLljgRrbFcqJGzcoCn9zGOMMqjlS15Vw3R-xH61FOqI_PTuh
V9; SG_SS=
*FiqaKnLyAAa07uurl0590WZaQT_MB2ABADQBEArZ1PguiyDAtPDyOkRRW6112rjH2JRlcSDKE5PeJ
0tP2gSAGUMLnaj6XFMMD8iN9PWsPQAAAF5tAAAACFcBBOEANq0AV0qukvFwdPkJUxtiHkyyZBrHBAI
5U-kvxhRlBKIaupJPjuLA3wGhBxW_LJkdVEql5R7ULKYCn-vboJ_JvpYsB0bq447ck-JtehSYHP8de
Txy056qh9BAIumLKYsQicKSkNoD0D2qTPNNS66Kc3Hu6YflkDaV0C5Co58L4ioX55BZXgv3CUiUV3o
ADaj80s_fmR9z33ALHsDQyjr-zf6Mz8zF4-JNS0Noj_VT2C8eLDs8r-_49TAxGW8fxdyRgE_28jhEb
LQepMdCkEpQcU3bJqPkt0hnKCZuGoAtdzEeRGEj0sB_dG252zYkIprKY3g3-BfwZewivdlWzWphoCY
ZfGxZxm9VSSgcH2GNGuhLPT4bfCRo2xHtUblxhSpHAH7rh17-gTEXCND4ckReFUKcvjjFgvVALikjC
Sf0s00rBrGIegR0sHXmyckTcg3JidP3wurcGespuG2h9mVbq7dEeGwbkj7Vm4dBrg5FTVNs8w9S39Z
bkrtstkMeKgETVVUHHyLnskov0O40mMrKhfSKueCXNzlgCswhpmnWK-H-bSCkOD-Kl9s5mepptSaBBe
DM_kPRWr_YlHu3Z99bDogrsBHU8ChkUjjfpCs9dQdles6Ab8t8llmzgoyj4T203GwSCMZW_JuUp3Bh
```

Response

```
HTTP/2 200 OK
Date: Wed, 05 Nov 2025 21:07:08 GMT
Expires: -1
Cache-Control: private, max-age=0
Content-Type: text/html; charset=UTF-8
Strict-Transport-Security: max-age=31536000
Content-Security-Policy: object-src 'none';base-uri 'self';script-src
'nonce-BRtVCo_gkctVr-rsMheZBg' 'strict-dynamic' 'report-sample' 'unsafe-eval'
'unsafe-inline' https: http:;report-uri
https://csp.withgoogle.com/csp/gws/cdtl
Cross-Origin-Opener-Policy: same-origin-allow-popups; report-to="gws"
Report-To:
{"group":"gws","max_age":2592000,"endpoints":[{"url":"https://csp.withgoogle.c
om/csp/report-to/gws/cdtl"}]}
Accept-Ch: Sec-CH-Prefers-Color-Scheme
Accept-Ch: Downlink
Accept-Ch: RTT
Accept-Ch: Sec-CH-UA-Form-Factors
Accept-Ch: Sec-CH-UA-Platform
Accept-Ch: Sec-CH-UA-Platform-Version
Accept-Ch: Sec-CH-UA-Full-Version
Accept-Ch: Sec-CH-UA-Arch
```

Inspector

Selection: 15 (0xf)

Selected text: jon+is+a+legend

Decoded from: URL encoding

jon is a legend

Request attributes	2
Request query parameters	11
Request cookies	4
Request headers	33
Response headers	28

- **WHAT**
  - Jon uses an intercepting proxy (e.g., Burp Suite) to examine requests and experiment with changes in a controlled way.

- **CONCEPT**
  - Proxies let you pause, view and modify outgoing requests and incoming responses to understand server-side checks.
  - This is diagnostic: not required for basic testing, but useful for investigating behaviour.

- **WHY**
  - **Benefit:** Helps test server validation and understand what fields are enforced server-side vs client-side.
  - **Problem:** Modifying traffic can have real effects; doing it carelessly may break services or access data you shouldn't.
  - **Safe solution:** Use a test account, limit load, and follow program rules. Avoid destructive or privacy-invading manipulations.

- **Quick tip:** Log every change you make and the resulting server behaviour for reporting.

Experimenting with Payloads

- **WHAT**
  - Jon uploads different file types (images, SVGs, malformed but harmless files) to see what the application accepts and how it renders them.

- **CONCEPT**
    - "Payloads" are test inputs designed to exercise validation logic. Some formats (SVG) can contain active content; others might be malformed to trigger errors.
    - The goal is to discover weak validation or unsafe rendering.

- **WHY**
    - **Benefit:** Finds gaps where the app accepts input it shouldn't or renders it unsafely.
    - **Problem:** Certain payloads can cause harm or access other users' data if used badly.
    - **Safe solution:** Use non-malicious test payloads and focus on observable behavior (does the server accept it? how is it displayed?). Do not use real malware or attempt privilege escalation.

Interpreting Errors and Application Responses

- **WHAT**
    - Jon watches server responses: acceptance messages, error codes, stack traces, or encoding errors from malformed uploads.

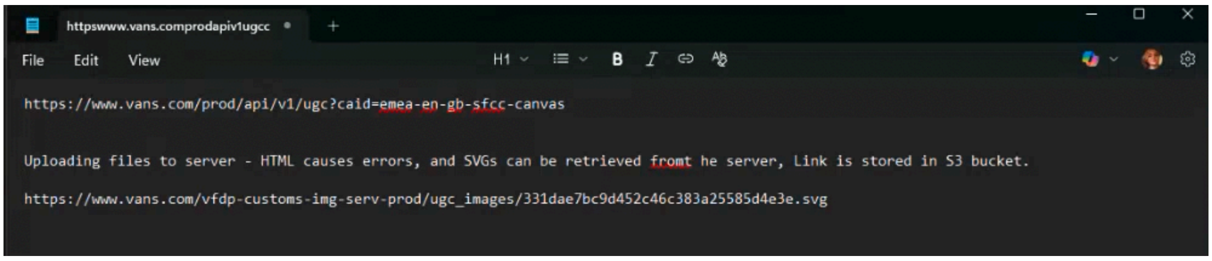
- **CONCEPT**
    - Error messages and HTTP codes are signals. Detailed errors can leak technology, file paths, or logic flow — which help an attacker craft exploits.

- **WHY**
    - **Benefit:** Helpful errors speed debugging and testing; in security, too-detailed errors can be a problem.
      Problem: Verbose errors can expose internals (frameworks, file paths, library versions).
    - **Safe solution:** Record exact responses; recommend to owners that production systems use generic error messages and proper logging instead of leaking internals to users.

- **Quick tip:** Capture full responses but redact sensitive data in your notes and reports.

Documenting Your Process

(5th November 2025) - JAGUAR COHORT
Made by Chris Cownden
I'm creating recap guides as revision notes for future use. Please use this document as you see fit.



```
https://www.vans.com/prod/api/v1/ugc?caid=emea-en-gb-sfcc-canvas

Uploading files to server - HTML causes errors, and SVGs can be retrieved fromt he server, Link is stored in S3 bucket.
https://www.vans.com/vfdp-customs-img-serv-prod/ugc_images/331dae7bc9d452c46c383a25585d4e3e.svg
```

- **WHAT**
  - Jon saves URLs, request/response logs, screenshots and writes step-by-step notes about what he tried and saw to come back to at a later date.
- **CONCEPT**
  - Good documentation makes findings reproducible and credible — and is essential for responsible disclosure and bug-bounty triage.
- **WHY**
  - **Benefit:** Clean reports increase likelihood of validation and reward.
  - **Problem:** Poor documentation wastes reviewers' time and may get your report closed.
  - **Safe solution:** Include scope proof, steps to reproduce, impact summary, and suggested remediation. Redact any customer or private data you accidentally captured.

Terminology

Bug Bounty: A monetary reward for finding security flaws in a company's software.
Scope: The list of systems, features, or endpoints that are permitted for testing during bug bounties.

Nmap: A network scanning tool used for enumeration and vulnerability assessment.
S3 Bucket: Amazon's cloud storage container, often used for storing uploaded files and assets.

Cross-Site Scripting (XSS): A web vulnerability where malicious JavaScript is injected into web pages.

SVG (Scalable Vector Graphics): An XML-based image file type that can be vulnerable to script injection.

Access Control List (ACL): A permission system determining user access to resources on an application.

(5th November 2025) - JAGUAR COHORT

Made by Chris Cownden

I’m creating recap guides as revision notes for future use. Please use this document as you see fit.

POST Request: An HTTP method for sending data from client to server (commonly used for file uploads).

Signal: A reputation metric reflecting quality and impact of bug reports on HackerOne.

Information Disclosure: Exposing sensitive internal data to unauthorized or public access.

Sanitization: The process of cleaning and filtering input to prevent attacks such as XSS.

Enumeration: The process of discovering and mapping out features, endpoints, or assets during hacking.

Dynamic Application Security Testing (DAST): Tools and methods that analyze running applications for vulnerabilities.

IDOR (Insecure Direct Object Reference): A vulnerability allowing unauthorized access to resources by manipulating input.

BLOB (Binary Large Object): A storage format for large, binary data in web applications (commonly used with file uploads).

Payload: Data (often malicious or for testing) sent as part of an attack, such as code designed to exploit a vulnerability.

Brute Forcing: A trial-and-error attack method to gain unauthorized access, often by guessing passwords or resource names.

Intercept: To capture and possibly alter data being sent between a browser and a server, commonly done with tools like Burp Suite.

Repeater: A feature in Burp Suite that allows repeated, manual modification and resending of web requests for testing.

Encoding: Converting data into a different format (like base64) for transmission; not the same as encryption.

Onerror: An HTML attribute used to specify code (often JavaScript) that runs if an image or script fails to load—often leveraged in XSS.

Base64: A method of encoding binary data as ASCII text, commonly used for images and file transfers in web applications.

Cookie: Small data stored in a user's browser to track sessions and preferences, sometimes exploited in attacks.

Vulnerability Scanner: An automated tool that looks for security weaknesses in software or websites.

Sanitization Bypass: Techniques used by attackers to get around security protections that filter or clean user input.

Resource-Based Policy: Rules applied (often in cloud services like AWS) to control access to files or objects, such as in S3 buckets.

Acronyms

MFA = Multi Factor Authentication
S3 = Simple Storage Service (Amazon S3 Bucket)
HTML = HyperText Markup Language
XML = Extensible Markup Language
SVG = Scalable Vector Graphics
PNG = Portable Network Graphics (image format)
DAST = Dynamic Application Security Testing
IDOR = Insecure Direct Object Reference
CPSA = Certified Practitioner Security Analyst
OWASP = Open Web Application Security Project
BLOB = Binary Large Object

(5th November 2025) - JAGUAR COHORT

Made by Chris Cownden

I'm creating recap guides as revision notes for future use. Please use this document as you see fit.

Analogy from today's session - Ultimate Treasure Hunt

Imagine you're a treasure hunter at a massive amusement park. The owners want to make the park safer and more fun, so they invite people like you to find hidden problems maybe a wobbly step or a creaky door. If you find and report these issues, you might get a cool badge, a shoutout, or even a cash prize.

But here's the catch, you're not allowed to snoop around every ride. Some areas are out of bounds, and if you go poking your nose where it isn't allowed, you'll get in trouble (maybe even chased out by security!). So you grab your map (the "scope"), follow the rules, and use a fancy magnifying glass and toolkit (think: Burp Suite, dev tools) to inspect things closely without breaking anything.

As you explore, you learn how each ride works. Maybe you spot a loose bolt on a roller coaster (a vulnerability!), or notice a locked gate that shouldn't be locked (an access issue!). You document what you found, how you found it, and how it could be fixed. If you do this well, your reputation as a top-notch treasure hunter grows, and you get invited to hunt for even bigger prizes.

Sometimes, the clues are tricky. You might try unlocking a door with a clever trick, but it doesn't work—this is like testing different ways to exploit a bug. But each attempt teaches you something new, even if you don't find gold right away.

Bug bounty hunting is basically being a detective-hero at the digital amusement park: exploring, respecting the boundaries, using your tools wisely, and helping make the park safer for everyone. And hey, as John reminded us, every attempt—success or not—gets you better at the game!

General Q&A from the session

1. What are bug bounties?

Bug bounties come up as part of the CPSA. It's when a company or somebody who has software puts out a monetary value or a prize for finding bugs within their system.

2. What does it mean when it's out of scope?

It means you can't test it. You're not allowed to test it.

3. What does it mean if it's now not in scope?

Sometimes if they're out of scope all of a sudden, they might have had a breach somewhere and don't want anybody playing with it. Potentially an active investigation or breach.

4. Is encoded the same as encrypted?

No, encoding and encryption are not the same. Encrypted is when you run it through an algorithm and add a key or encryption cipher; encoded is just for data transformation without security.

5. How do I interject the cross-site scripting payload?

You need specialist tooling, like Burp Suite Community Edition (free) or OWASP ZAP, to intercept and modify requests. Burp Suite is the more common professional tool.

6. Is Burp Suite a free application?

Yes, Burp Suite Community Edition is free; Professional is paid.

7. Why do you think this Burp Suite is a useful tool?

Because you can catalog and tamper with web requests easily to test and exploit bugs, making the process much simpler than manually searching through requests.

8. Can cookies be hacked?

Yes, you can absolutely try and inject or exploit through cookies. Cookies are a potential injection point for exploits.

9. Do you need to run Burp Suite in a virtual machine?
No, John runs it natively and isn't too worried for web applications, rarely encountering malware in that context.

10. What happens if the server doesn't accept a file type during upload (like HTML or SVG)?

The server responds with "access denied;" application usually allows only certain types (PNG, JPG, SVG, etc.) and will reject others for security.

11. How easy is it to bypass file type restrictions in uploads?

Not always straightforward, but many such restrictions can be bypassed with creativity. Double file extensions or tampering with file content/headers are common tricks.

12. How do you create an XSS payload in SVG?

You inject JavaScript or event handlers into SVG tags, such as `<svg><image onerror="alert(1)" ...></svg>`, using the scalable vector graphics format.

13. Does uploading an SVG with XSS actually work?

It depends on the server’s sanitization. Sometimes, sanitization can be bypassed. In the tested scenario, certain SVG payloads caused encoding errors, but with continued testing, such attacks may work.

14. How does AI like Claude help in practical hacking?

Claude (and similar AIs) can help generate payloads, analyze responses, and streamline parts of the vulnerability testing process. It’s particularly helpful for repetitive or creative tasks.

Extra test ideas for image-related vulnerabilities

I asked ChatGPT what else Jon could have done to look for Image Vulnerabilities. This is what it suggested. I haven’t got a clue what this means but I’m sure we’ll cover it soon.

- EXIF / metadata abuse
- SVG with active content
- Malformed / corrupted images
- Content-type / MIME sniffing attacks
- Insecure Direct Object References (IDOR) / predictable URLs
- Open S3 / bucket misconfigurations
- Race conditions / temp-file attacks
- Server-Side Request Forgery (SSRF) via image processing
- Thumbnail generation & format conversions
- Cached content abuse / CDN cache poisoning
- Filename / path traversal
- Dependency & CVE analysis

Keep up the great work team! You’ve got this!

All the best, Chris Cownden (fellow JAGUAR cohort team player)

Feel free to connect with me here: https://www.linkedin.com/in/chriscownden/