RSAConference2020

San Francisco | February 24 – 28 | Moscone Center

HUMAN ELEMENT

SESSION ID: DSO-R02

Case Files from 20 Years of Business Logic Flaws

Chetan Conikee, CTO

ShiftLeft

#RSAC

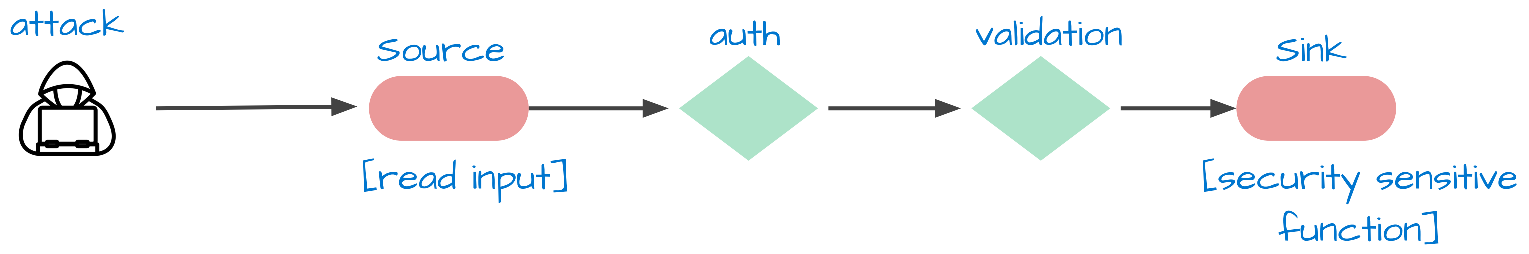
Security Issues

Two broad categories

- **Vulnerabilities** that have **common** characteristics across different applications
- **Design Flaws** that are **specific** to the application or business domain

Vulnerabilities

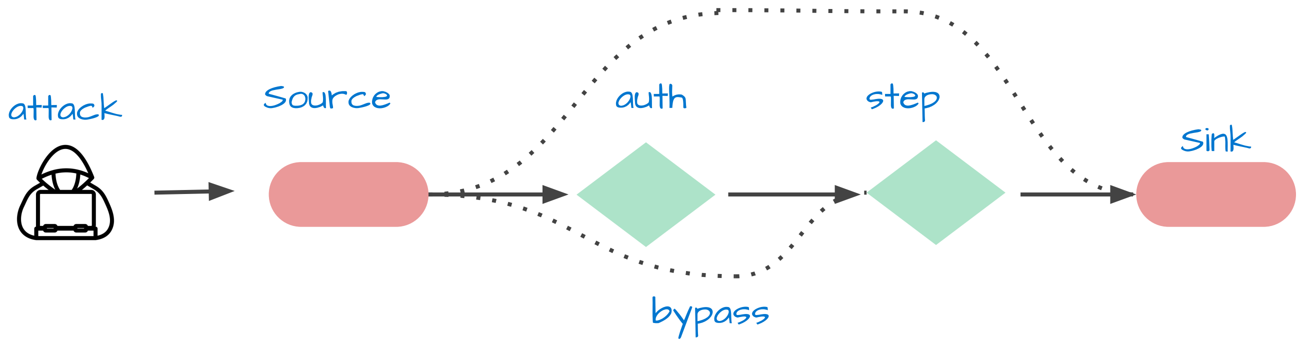
Implementation problems in code



- Improper API **usage patterns**, e.g. **`malloc`** without **`free`**,
- Lack of **input validation**, leading to injections, buffer overflows etc.
- Lack of **access controls**, which may lead to confidential information being leaked, altered or denied access to.

Flaws

Misuse of application by circumventing business rules



- Abuse functionality
- Manipulate parameter(s)
- Bypass or side step workflow

CHECK ENGINE

FLAWS

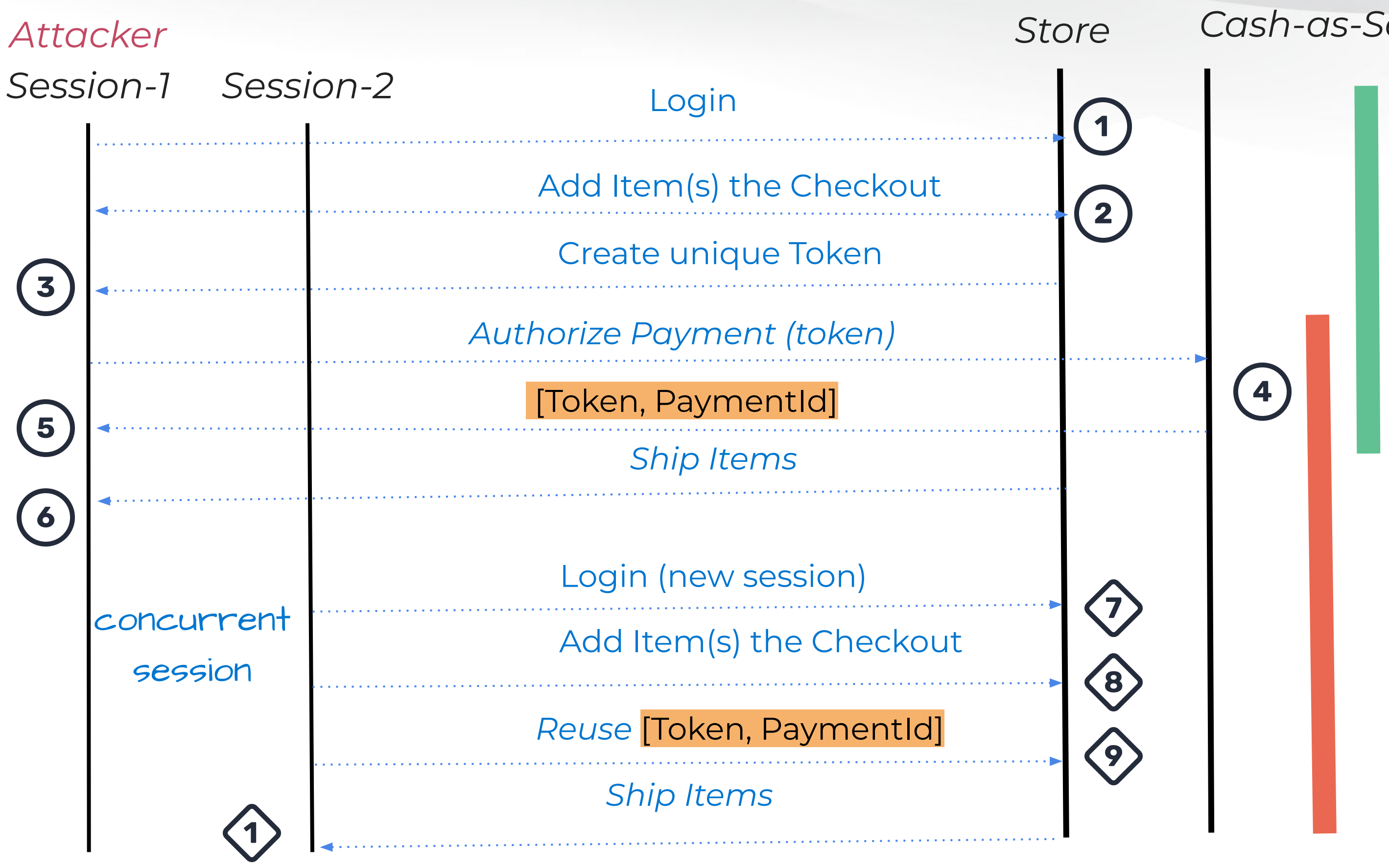
Source of Flaws

- High velocity development
- Poor documentation and testing
- Security not a part of early design process
- Lack of automated checks in CI pipeline
- Lack of architectural risk analysis to identify attack resistance, ambiguity and weakness in software design

OWASP Categorization (WIP)

- Business Consistency Security
- Verification Code Breakthrough
- Business Authorization Security
- Business Process Out of Order
- Business Interface Invocation
- Aging Bypass Testing
- Identity Authentication security
- Retrieving Password vulnerabilities
- Business Data Tampering
- Users Enter Legality

FLAWS (Example: ***Paying less for more items***)



Initiate low cost transaction using normal workflow. Intercept request and capture [TOKEN, PAYMENTID]

Login using another browser (new session) and initiate new transaction, side-step auth phase and, reuse [TOKEN, PAYMENTID]

ShiftLeft

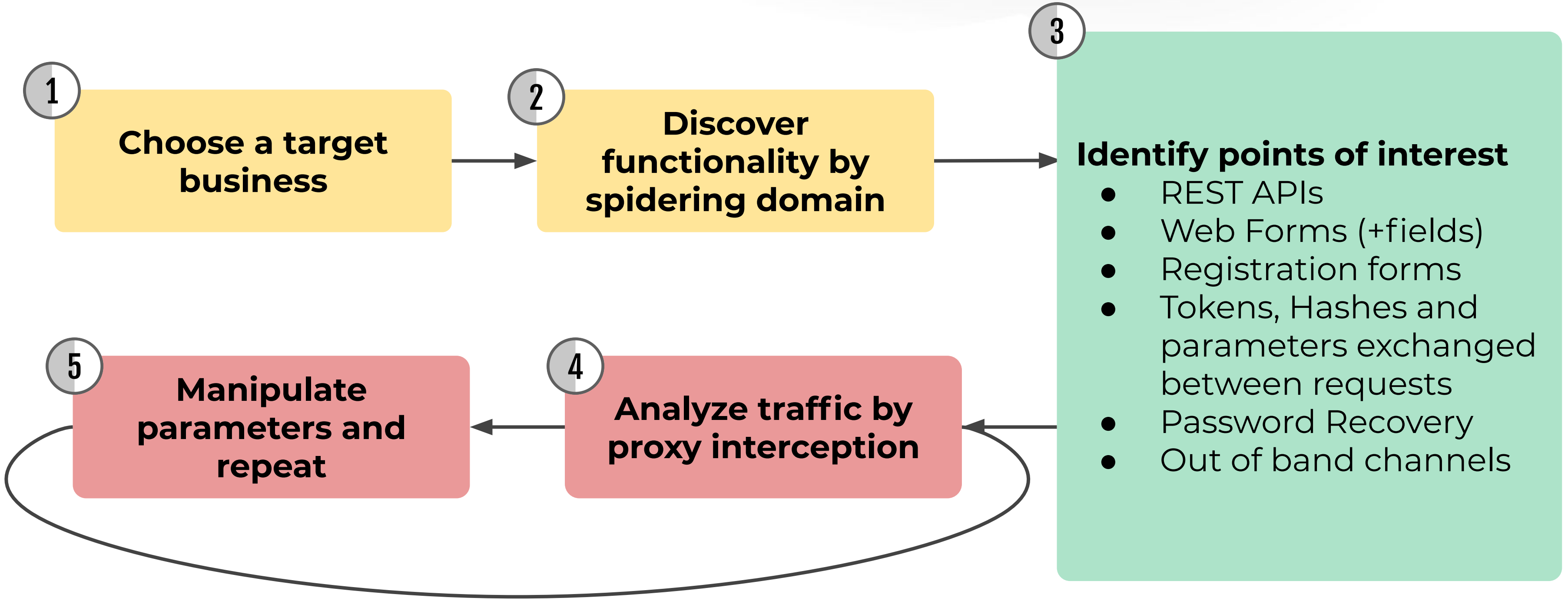
RSAConference2020

FLAWS (Example: *Paying less for more items)*

- Is **meta-data** (cost,quantity,user,session) mapped to a payment transaction **token**.
- Is an authorization/capture **unique** to a **userId**?
- Is an authorization/capture **unique** to a **session** scope belonging to same **userId**?

EXPLOITING FLAWS

Adversarial Mindset

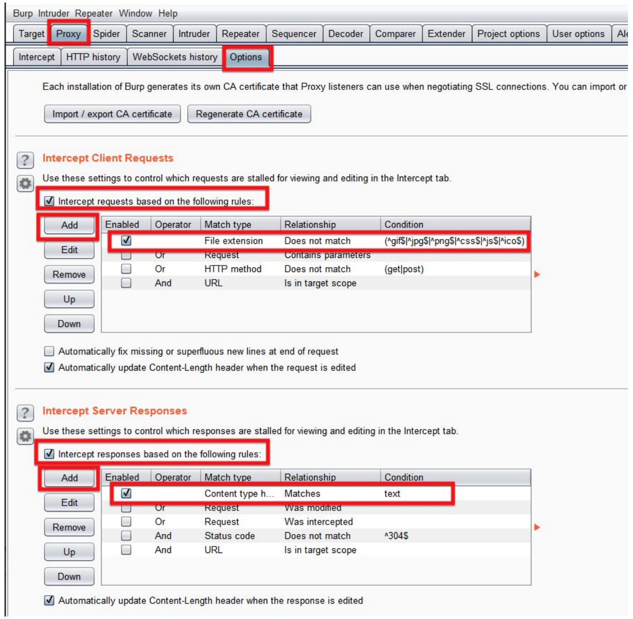


1. **Choose a target business**
2. **Discover functionality by spidering domain**
3. **Identify points of interest**
   - REST APIs
   - Web Forms (+fields)
   - Registration forms
   - Tokens, Hashes and parameters exchanged between requests
   - Password Recovery
   - Out of band channels
4. **Analyze traffic by proxy interception**
5. **Manipulate parameters and repeat**

An Attacker’s Toolchain (Burp)

1 **Interceptor** -> Choose Target and Intercept via proxy

2 **Repeater** -> Automate parameter manipulation and observe response(s)



Each installation of Burp generates its own CA certificate that Proxy listeners can use when negotiating SSL connections. You can import or

Intercept Client Requests

Use these settings to control which requests are stalled for viewing and editing in the Intercept tab.

☑ Intercept requests based on the following rules:

Enabled	Operator	Match type	Relationship	Condition
☑		File extension	Does not match	(^gif$\|^jpg$|^png$\|^css$|^js$\|^ico$)
☐	Or	Request	Contains parameters	
☐	Or	HTTP method	Does not match	(get|post)
☐	And	URL	Is in target scope	

☐ Automatically fix missing or superfluous new lines at end of request

☑ Automatically update Content-Length header when the request is edited

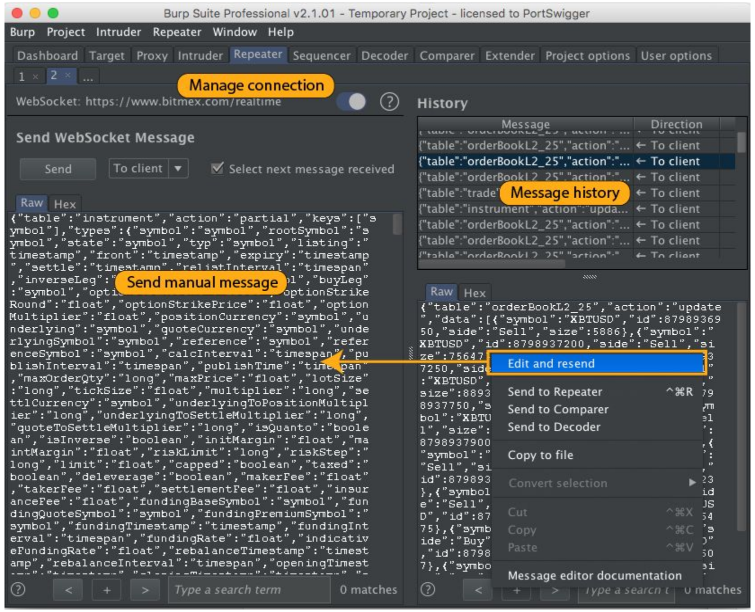
Intercept Server Responses

Use these settings to control which responses are stalled for viewing and editing in the Intercept tab.

☑ Intercept responses based on the following rules:

Enabled	Operator	Match type	Relationship	Condition
☑		Content type h...	Matches	text
☐	Or	Request	Was modified	
☐	Or	Request	Was intercepted	
☐	And	Status code	Does not match	^304$
☐	And	URL	Is in target scope	

☑ Automatically update Content-Length header when the response is edited



WebSocket: https://www.bitmex.com/realtime

☑ Select next message received

```
{"table":"instrument","action":"partial","keys":["s
ymbol"],"types":{"symbol":"symbol","rootSymbol":"s
ymbol","state":"symbol","typ":"symbol","listing":"
timestamp","front":"timestamp","expiry":"timestamp
","settle":"timestamp","relistInterval":"timespan"
,"inverseLeg":"[illegible]l","buyLeg"
:"symbol","opti[illegible]optionStrike
Round":"float","optionStrikePrice":"float","option
Multiplier":"float","positionCurrency":"symbol","u
nderlying":"symbol","quoteCurrency":"symbol","unde
rlyingSymbol":"symbol","reference":"symbol","refer
enceSymbol":"symbol","calcInterval":"timespan","pu
blishInterval":"timespan","publishTime":"timespan"
,"maxOrderQty":"long","maxPrice":"float","lotSize"
:"long","tickSize":"float","multiplier":"long","se
ttlCurrency":"symbol","underlyingToPositionMultipl
ier":"long","underlyingToSettleMultiplier":"long",
"quoteToSettleMultiplier":"long","isQuanto":"boole
an","isInverse":"boolean","initMargin":"float","ma
intMargin":"float","riskLimit":"long","riskStep":"
long","limit":"float","capped":"boolean","taxed":"
boolean","deleverage":"boolean","makerFee":"float"
,"takerFee":"float","settlementFee":"float","insur
anceFee":"float","fundingBaseSymbol":"symbol","fun
dingQuoteSymbol":"symbol","fundingPremiumSymbol":"
symbol","fundingTimestamp":"timestamp","fundingInt
erval":"timespan","fundingRate":"float","indicativ
eFundingRate":"float","rebalanceTimestamp":"timest
amp","rebalanceInterval":"timespan","openingTimest
```

Message	Direction
{"table":"orderBookL2_25","action":"...	← To client
{"table":"orderBookL2_25","action":"...	← To client
{"table":"orderBookL2_25","action":"...	← To client
{"table":"orderBookL2_25","action":"...	← To client
{"table":"trade"[illegible]	← To client
{"table":"instrument","action":"upda...	← To client
{"table":"orderBookL2_25","action":"...	← To client
{"table":"orderBookL2_25","action":"...	← To client
{"table":"orderBookL2_25","action":"	← To client

```
{"table":"orderBookL2_25","action":"update
","data":[{"symbol":"XBTUSD","id":87989369
50,"side":"Sell","size":5886},{"symbol":"
XBTUSD","id":8798937200,"side":"Sell","si
ze":75647[illegible]
7250,"side[illegible]
:"XBTUSD",[illegible]
size":8893[illegible]
8937750,"s[illegible]
bol":"XBTU[illegible]
l","size":[illegible]
8798937900[illegible]
"symbol":"[illegible]
"Sell","si[illegible]
id":879893[illegible]
},{"symbol[illegible]
e":"Sell",[illegible]
D","id":87[illegible]
75},{"symb[illegible]
ide":"Buy"[illegible]
,"id":8798[illegible]
7},{"symbo[illegible]
```

Weaponizing Flaws

Observe

Abuse Functionality

Enumerate

Weak password recovery

Exploit

Workflow bypass

Observe

Information Leakage

Enumerate

Insufficient Authorization

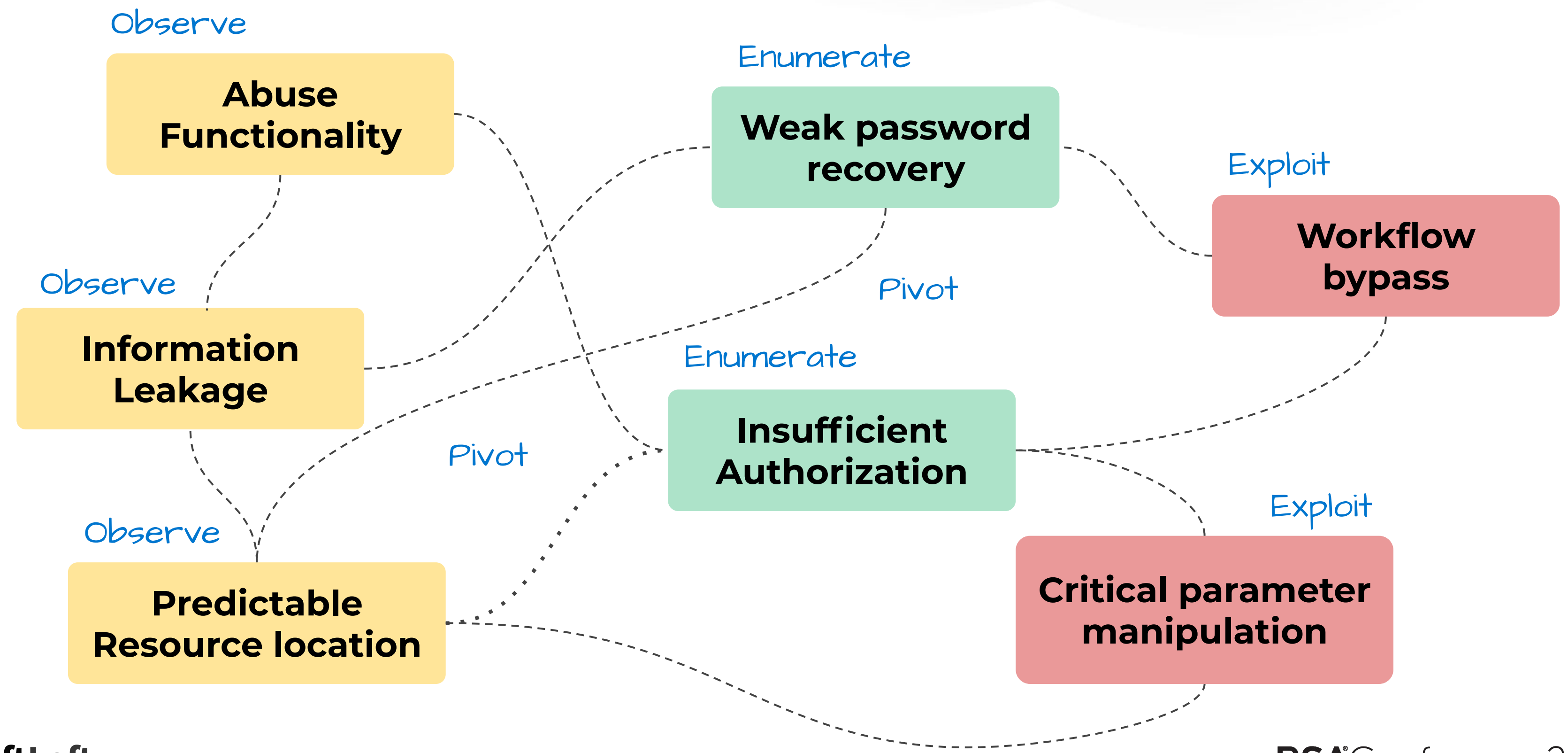
Exploit

Critical parameter manipulation

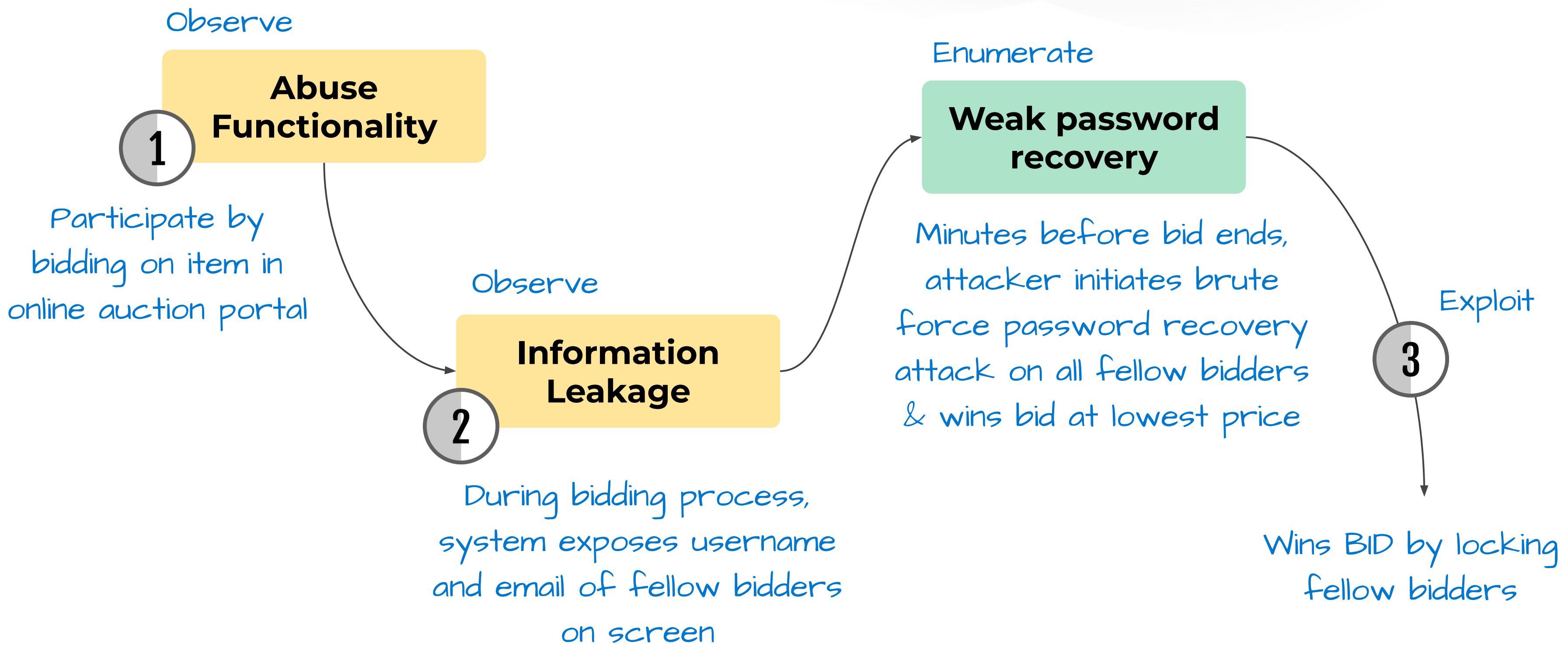
Observe

Predictable Resource location

Weaponizing Flaws



Weaponizing Flaws - Case 1 (Bidding)



Observe

1 Abuse Functionality

Participate by bidding on item in online auction portal

Observe

2 Information Leakage

During bidding process, system exposes username and email of fellow bidders on screen

Enumerate

Weak password recovery

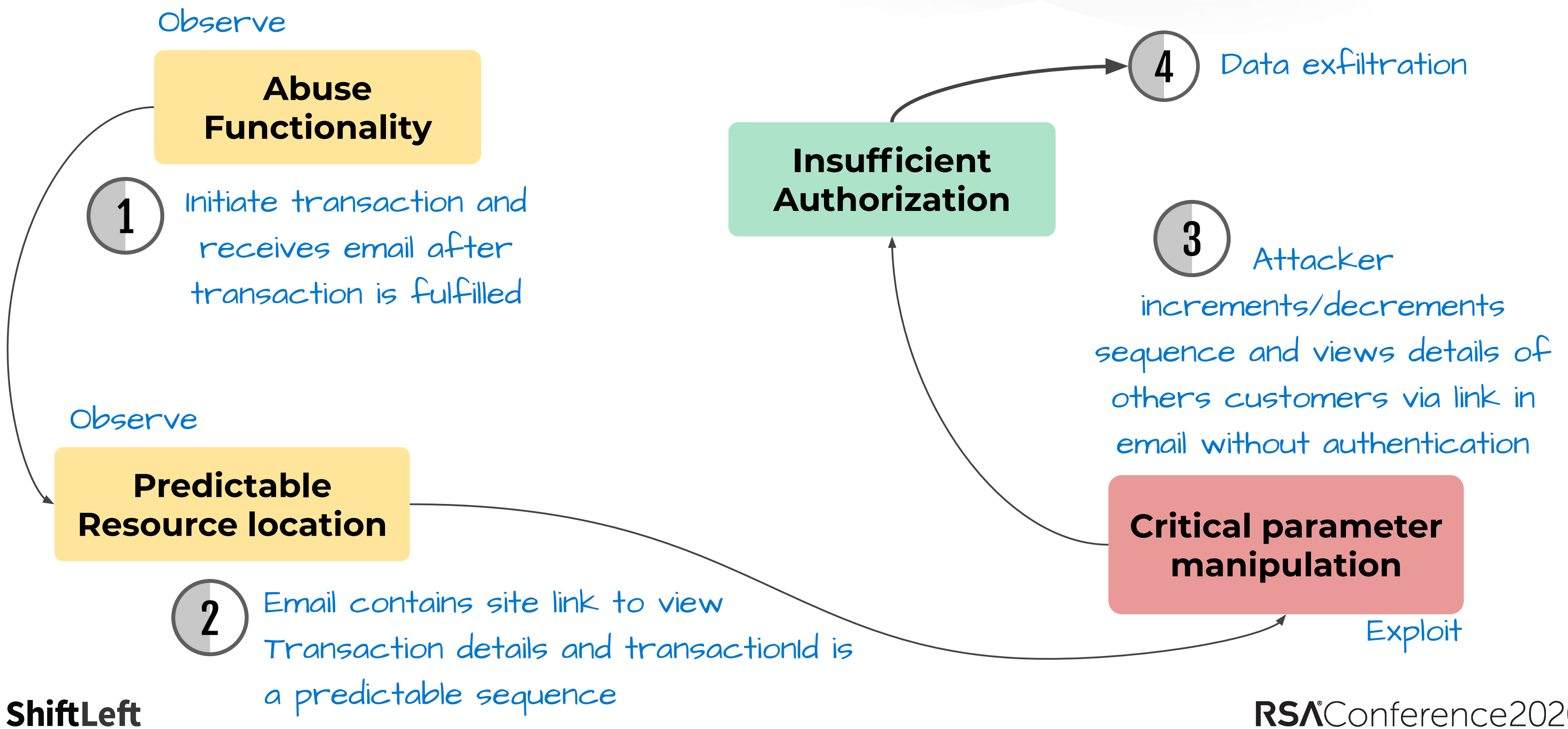
Minutes before bid ends, attacker initiates brute force password recovery attack on all fellow bidders & wins bid at lowest price

Exploit

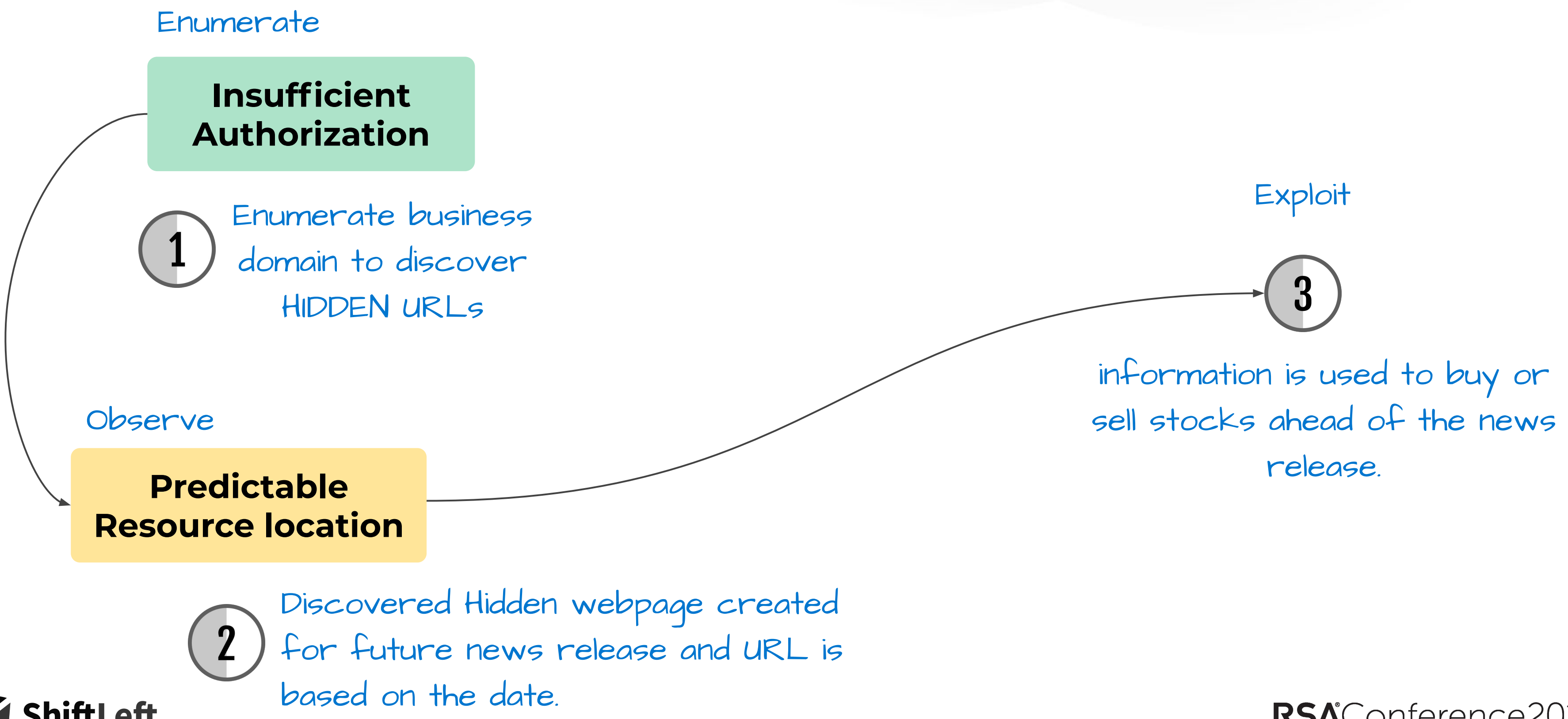
3

Wins BID by locking fellow bidders

Weaponizing Flaws - Case 2 (Finance)


Observe
Abuse Functionality
1
Initiate transaction and receives email after transaction is fulfilled
Observe
Predictable Resource location
2
Email contains site link to view Transaction details and transactionId is a predictable sequence
Critical parameter manipulation
Exploit
3
Attacker increments/decrements sequence and views details of others customers via link in email without authentication
Insufficient Authorization
4
Data exfiltration

Weaponizing Flaws - Case 3 (SaaS)



Enumerate business domain to discover HIDDEN URLs

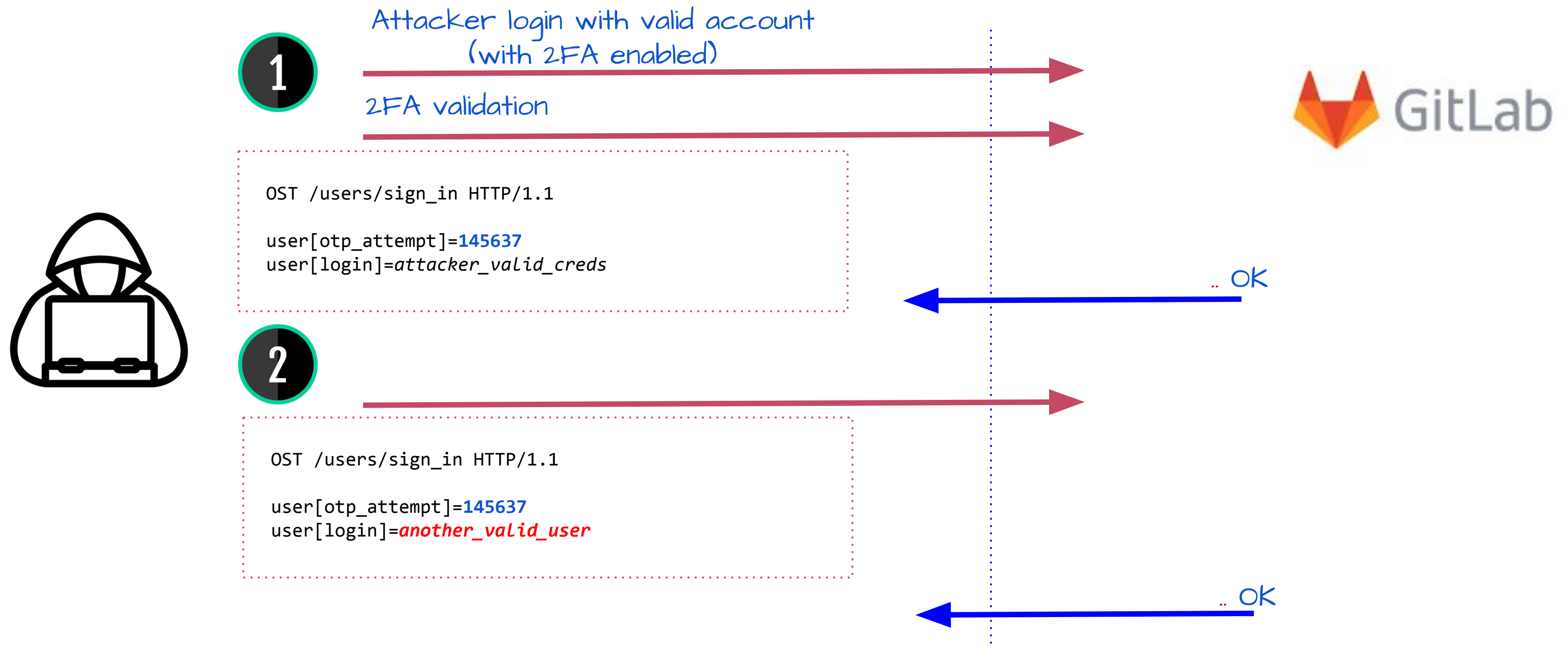
Discovered Hidden webpage created for future news release and URL is based on the date.

information is used to buy or sell stocks ahead of the news release.

CASE FILES

Bypass 2FA

Source : https://hackerone.com/reports/128085



```
OST /users/sign_in HTTP/1.1

user[otp_attempt]=145637
user[login]=attacker_valid_creds
```

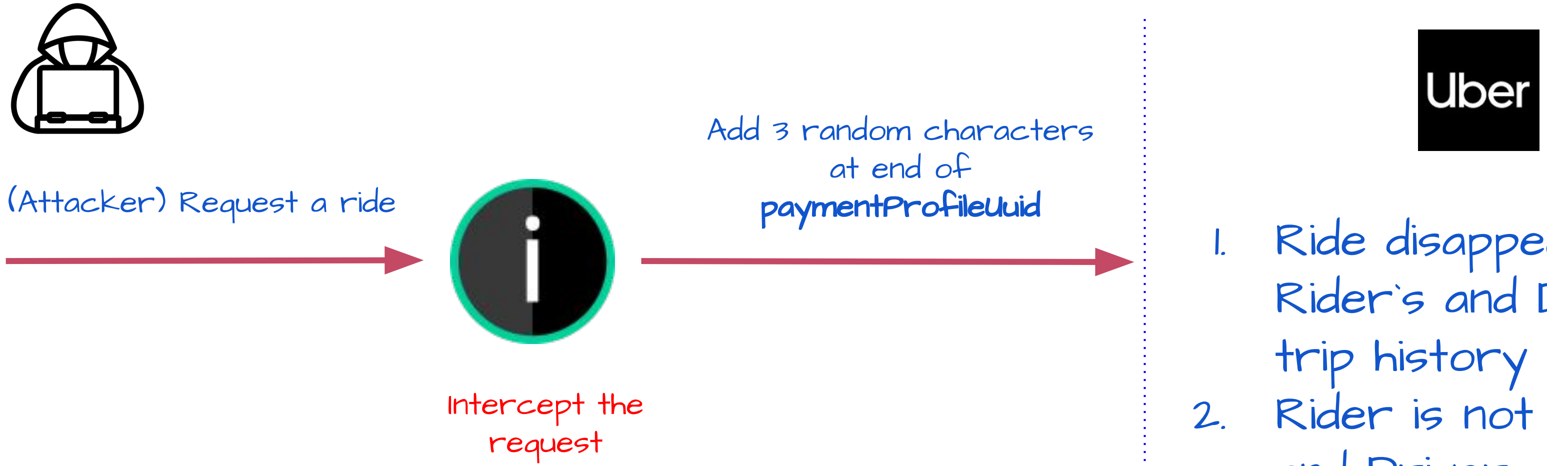
```
OST /users/sign_in HTTP/1.1

user[otp_attempt]=145637
user[login]=another_valid_user
```

Attacker signed in as valid user without authentication

Free Rides

Source : https://hackerone.com/reports/574638

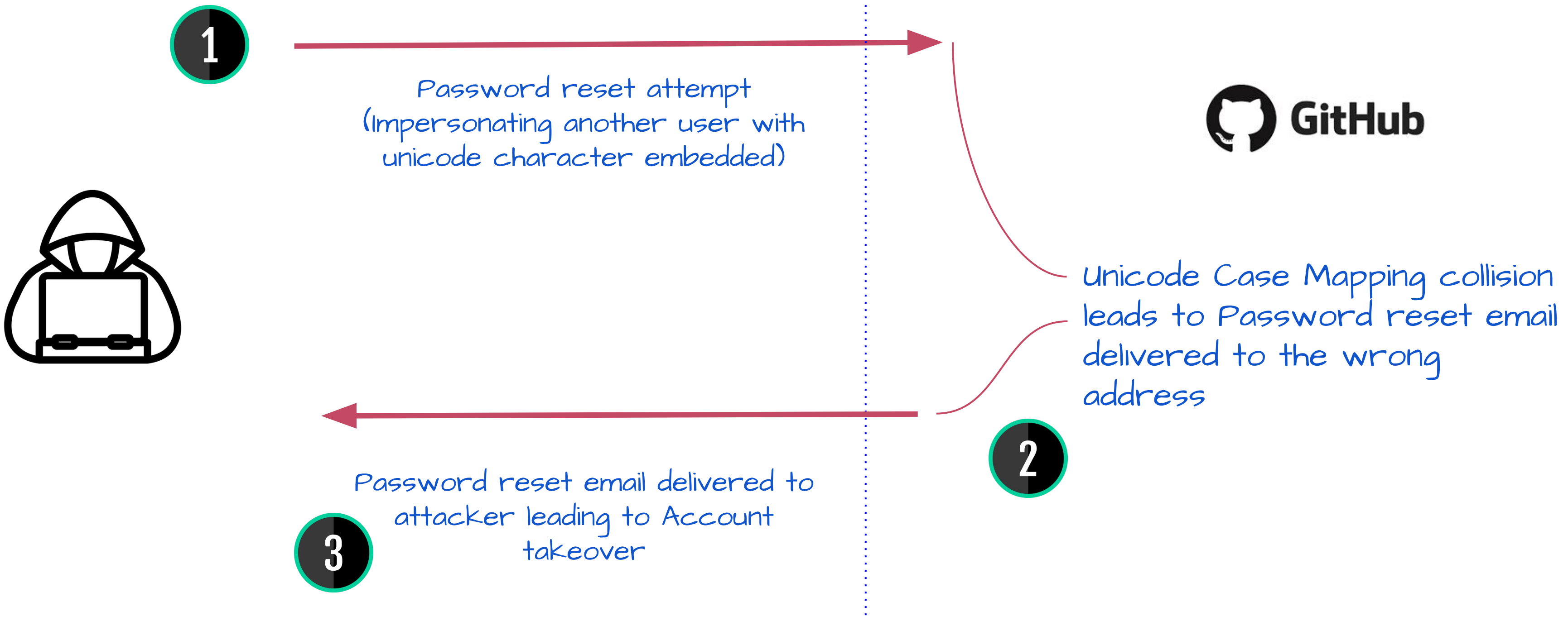
Lack of proper **paymentProfileUUID** validation allows any number of free rides without any outstanding balance


(Attacker) Request a ride
Intercept the request
Add 3 random characters at end of **paymentProfileUuid**
Uber

1. Ride disappears from Rider`s and Driver`s trip history
2. Rider is not charged and Driver will not be paid

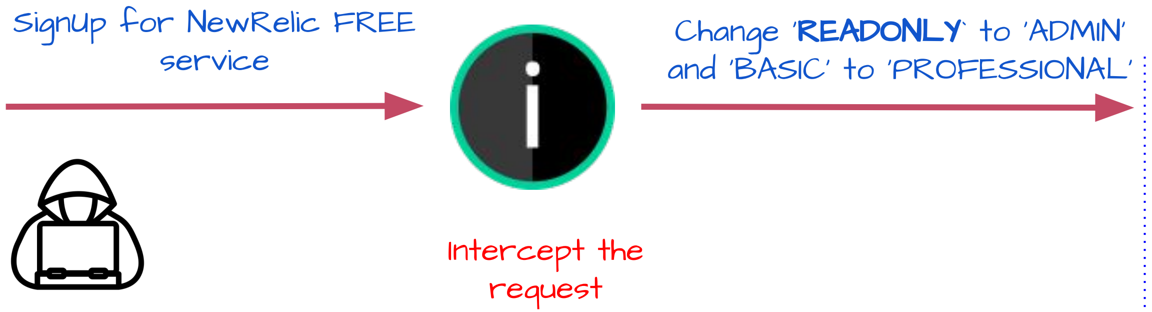
Hacking forgot password workflow

Source : https://eng.getwisdom.io/hacking-github-with-unicode-dotless-i/


1
Password reset attempt (Impersonating another user with unicode character embedded)
GitHub
Unicode Case Mapping collision leads to Password reset email delivered to the wrong address
2
Password reset email delivered to attacker leading to Account takeover
3

Availing Premium tier for free

Source : https://hackerone.com/reports/219356

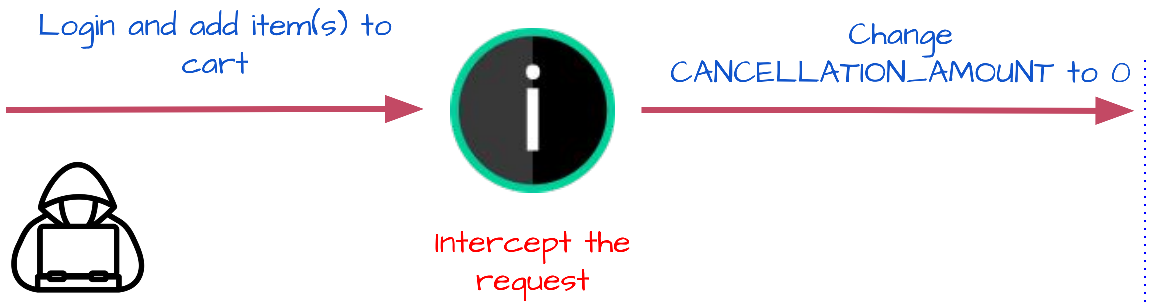

SignUp for NewRelic FREE service
Change 'READONLY' to 'ADMIN' and 'BASIC' to 'PROFESSIONAL'
Intercept the request

```
HTTP/1.1 200 OK
{"data":{"currentUser":{"userData":[
{"userLevel":ADMIN,
 "subscriptionLevel":" PROFESSIONAL"}
 ]
}}}
```

By switching 'userLevel' to 'ADMIN' and 'subscriptionLevel' to 'PROFESSIONAL' , he was able to avail professional features at cost of freemium

Buy items for less than intended price

Source : https://hackerone.com/reports/614523


Login and add item(s) to cart
Change CANCELLATION_AMOUNT to 0
Intercept the request

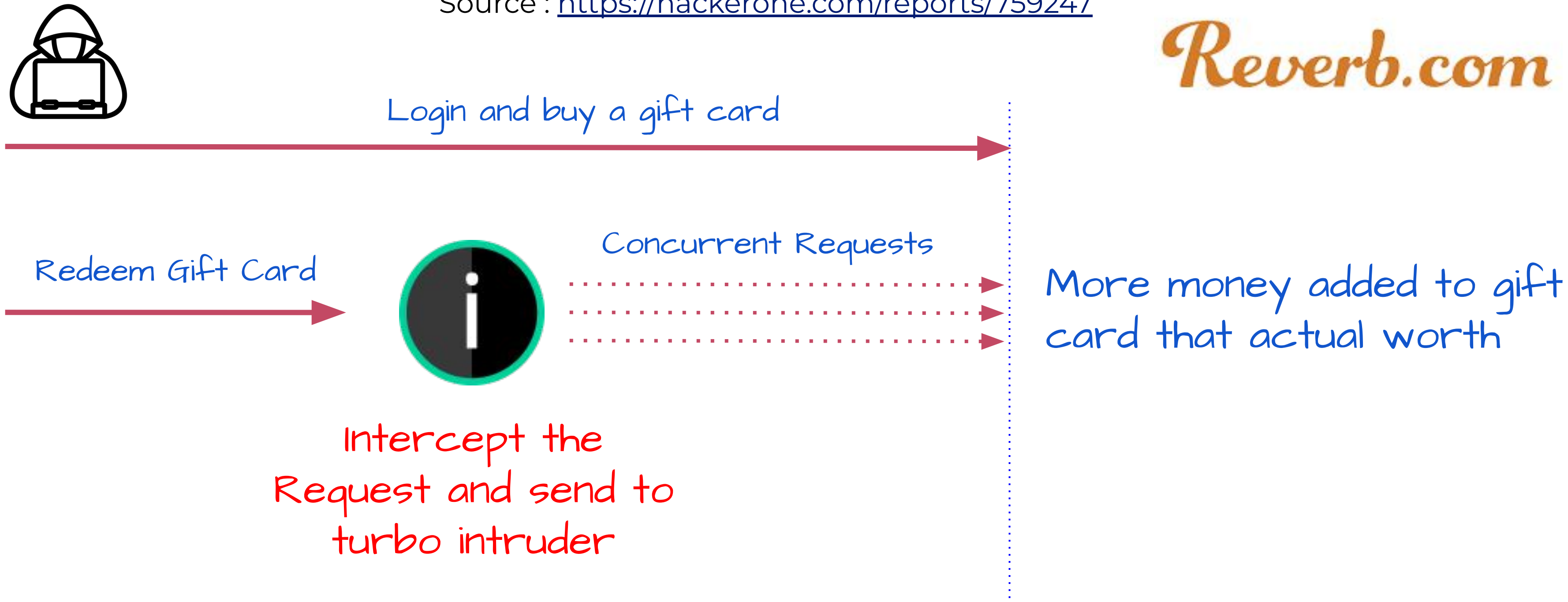
zomato

```
HTTP/1.1 200 OK
{...
{...
 "cancellation_amount":0}
 ]
}}}
```

Changing CANCELLATION_AMOUNT to 0 reduces ORDER_AMOUNT and cancels all previous cancellation amounts

Race Condition to get free stuff and steal money

Source : https://hackerone.com/reports/759247

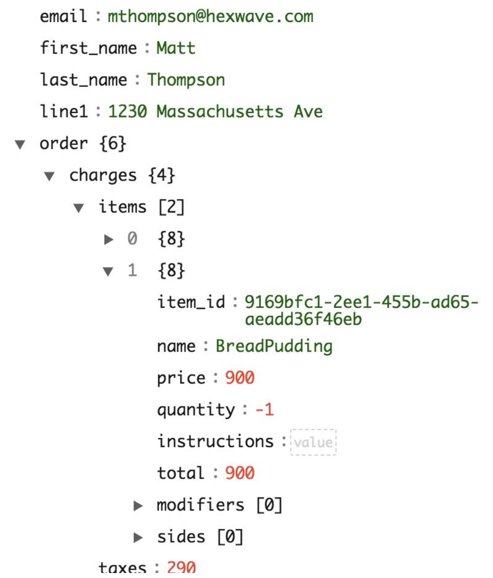

Reverb.com
Login and buy a gift card
Redeem Gift Card
Concurrent Requests
More money added to gift card that actual worth
Intercept the Request and send to turbo intruder

Total price manipulation using -ve quantity

Source : https://hackerone.com/reports/364843

1

Order Request JSON Object containing an additional item with negative quantity manipulates total item of order

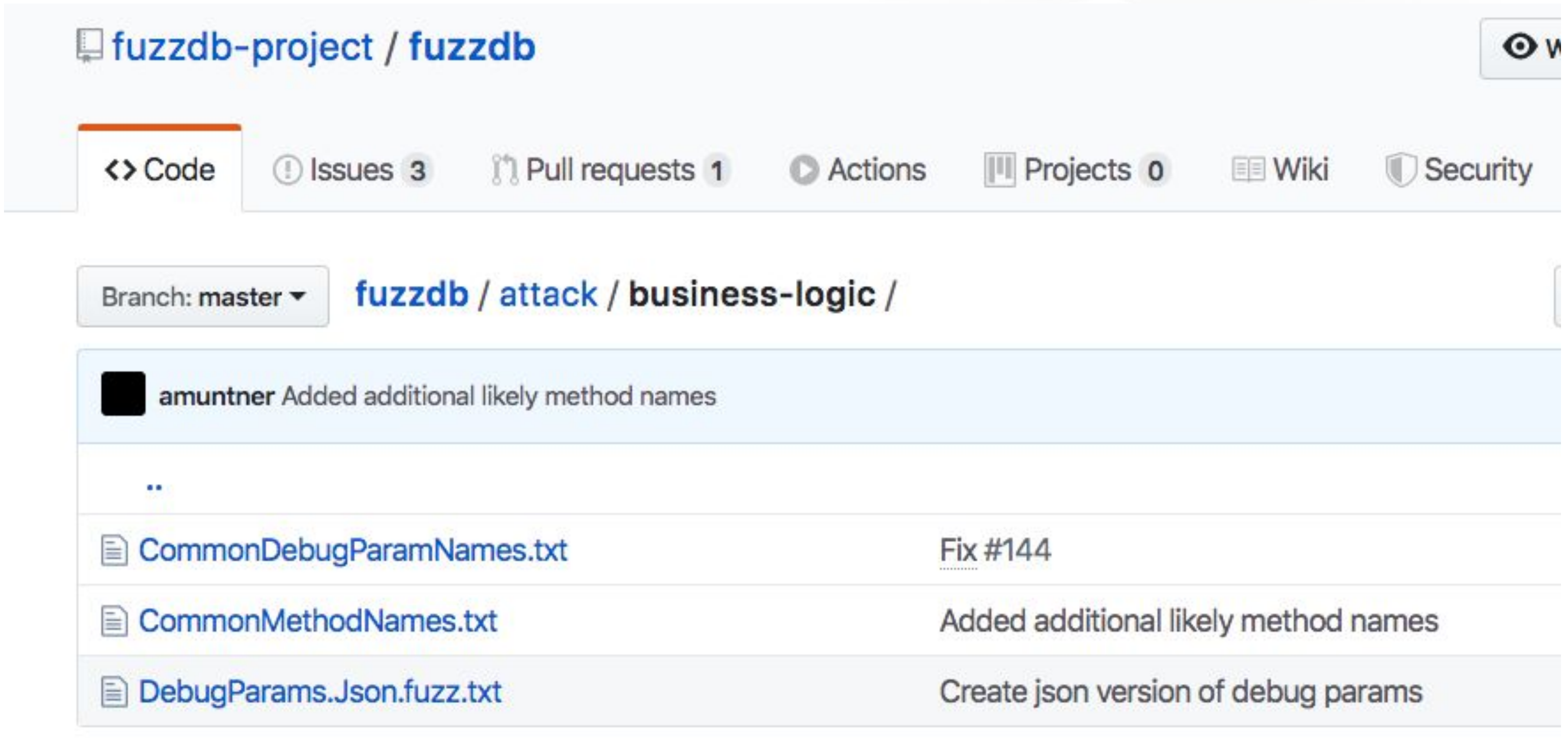

```
email : mthompson@hexwave.com
first_name : Matt
last_name : Thompson
line1 : 1230 Massachusetts Ave
▼ order {6}
  ▼ charges {4}
    ▼ items [2]
      ▶ 0  {8}
      ▼ 1  {8}
          item_id : 9169bfc1-2ee1-455b-ad65-
                    aeadd36f46eb
          name : BreadPudding
          price : 900
          quantity : -1
          instructions : value
          total : 900
        ▶ modifiers [0]
        ▶ sides [0]
    taxes : 290
```

2

Significantly reduce price of order

Attack AUTOMATION

Tools that hackers can leverage



fuzzdb-project / fuzzdb

Code | Issues 3 | Pull requests 1 | Actions | Projects 0 | Wiki | Security

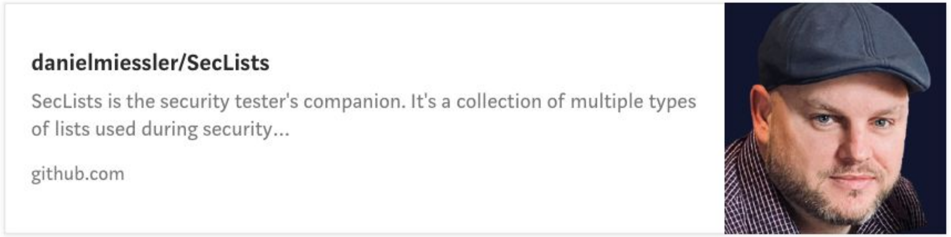
Branch: master ▾ fuzzdb / attack / business-logic /

amuntner Added additional likely method names

..

CommonDebugParamNames.txt	Fix #144
CommonMethodNames.txt	Added additional likely method names
DebugParams.Json.fuzz.txt	Create json version of debug params

Tools that hackers can leverage


danielmiessler/SecLists

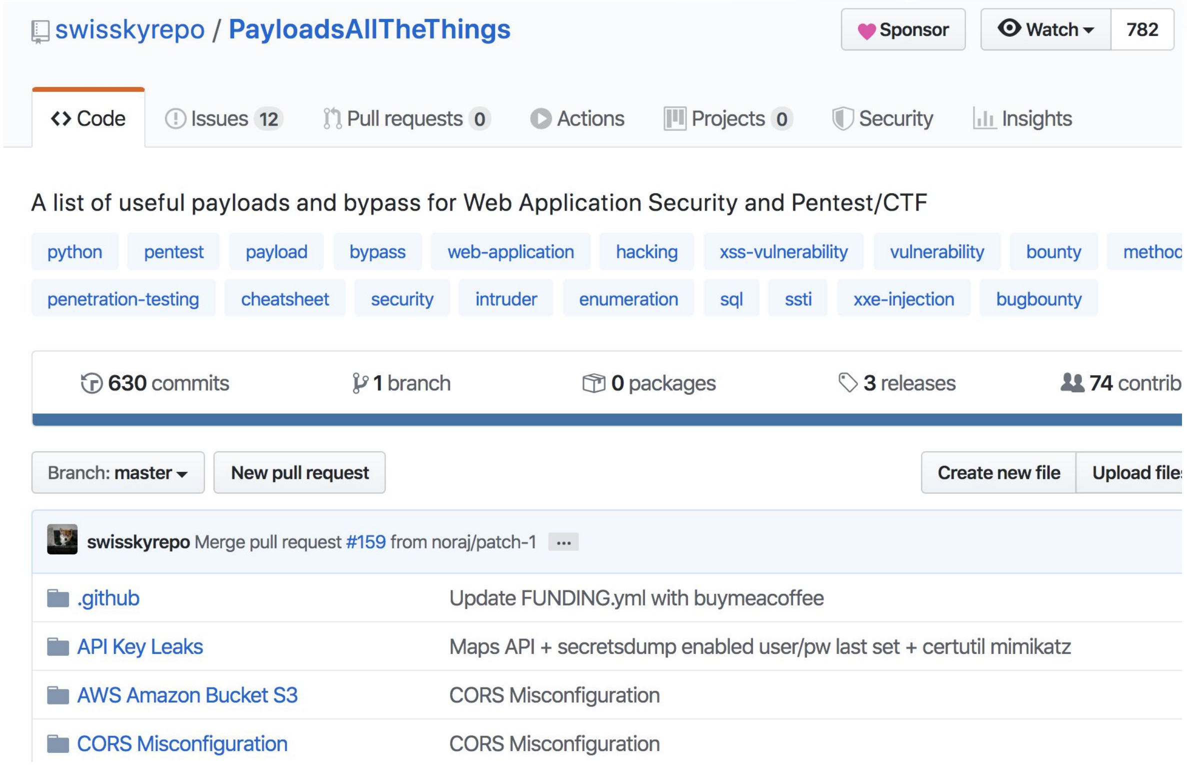
SecLists is the security tester's companion. It's a collection of multiple types of lists used during security...

github.com

SecList is pretty good for detecting already known vulnerabilities. Another way of fuzzing is to generate payloads randomly, including payloads like the below to try to induce errors in the web app:

- Payloads that are really long,
- payloads that contain odd characters of various encodings,
- and payloads that contain certain special characters, like the new line character, the line feed character, and more.

Tools that hackers can leverage



swisskyrepo / **PayloadsAllTheThings**

Sponsor | Watch 782

Code | Issues 12 | Pull requests 0 | Actions | Projects 0 | Security | Insights

A list of useful payloads and bypass for Web Application Security and Pentest/CTF

python pentest payload bypass web-application hacking xss-vulnerability vulnerability bounty methoc penetration-testing cheatsheet security intruder enumeration sql ssti xxe-injection bugbounty

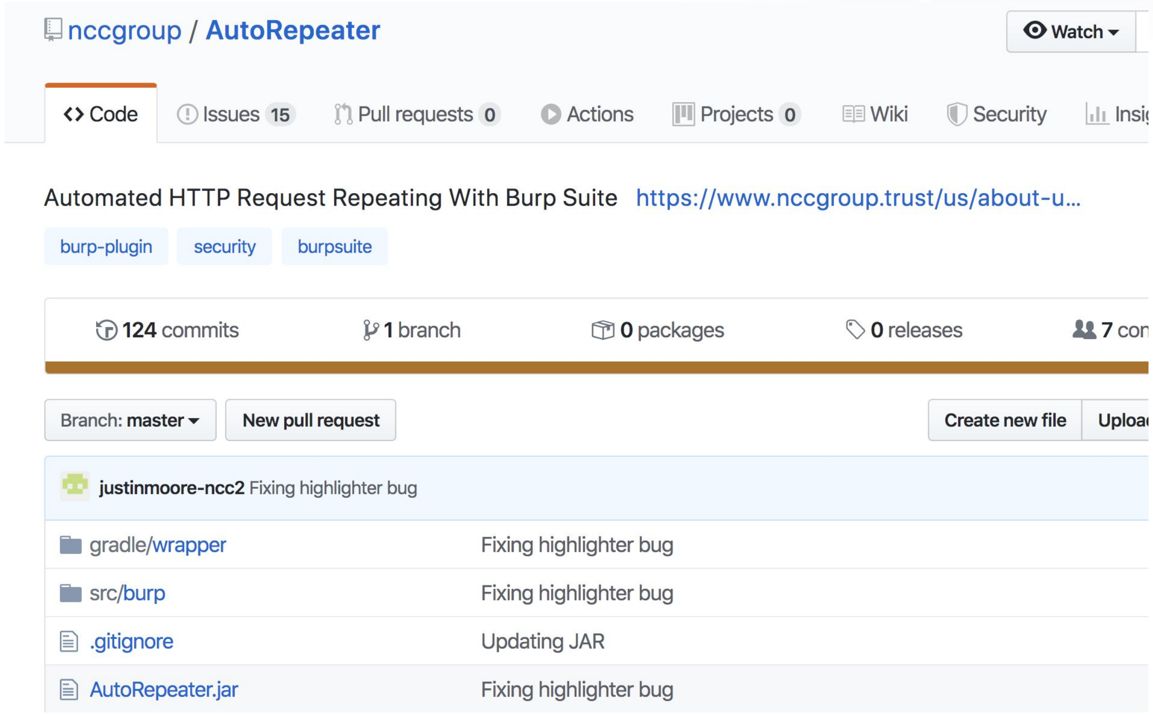
630 commits | **1** branch | **0** packages | **3** releases | **74** contrib

Branch: master | New pull request | Create new file | Upload file

swisskyrepo Merge pull request #159 from noraj/patch-1 …

.github	Update FUNDING.yml with buymeacoffee
API Key Leaks	Maps API + secretsdump enabled user/pw last set + certutil mimikatz
AWS Amazon Bucket S3	CORS Misconfiguration
CORS Misconfiguration	CORS Misconfiguration

Tools that hackers can leverage


nccgroup / AutoRepeater

Automated HTTP Request Repeating With Burp Suite https://www.nccgroup.trust/us/about-u...

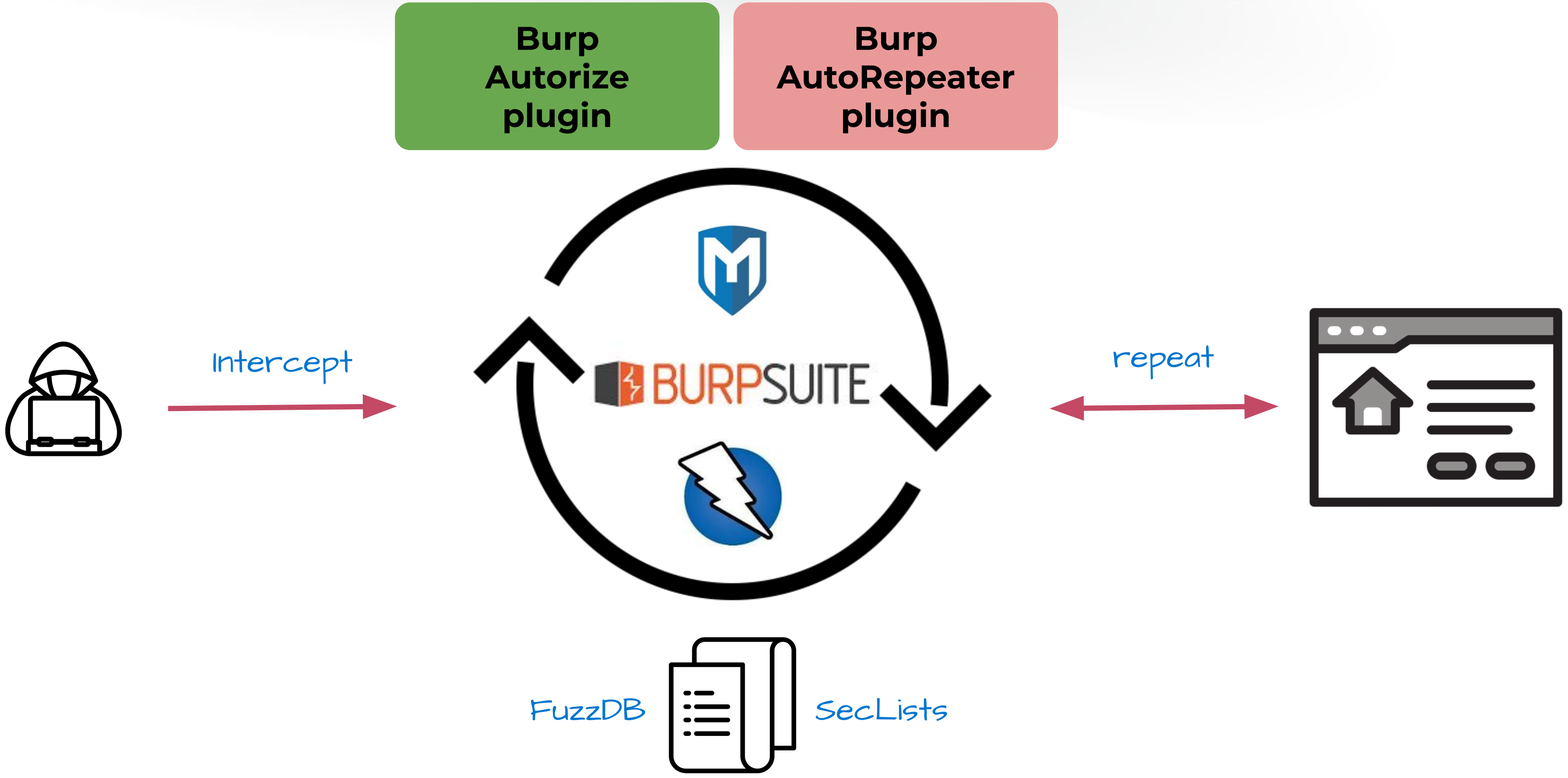
burp-plugin security burpsuite

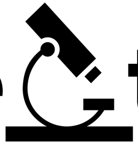
124 commits 1 branch 0 packages 0 releases 7 con

justinmoore-ncc2 Fixing highlighter bug

gradle/wrapper	Fixing highlighter bug
src/burp	Fixing highlighter bug
.gitignore	Updating JAR
AutoRepeater.jar	Fixing highlighter bug

LEVERAGING OPEN SOURCE



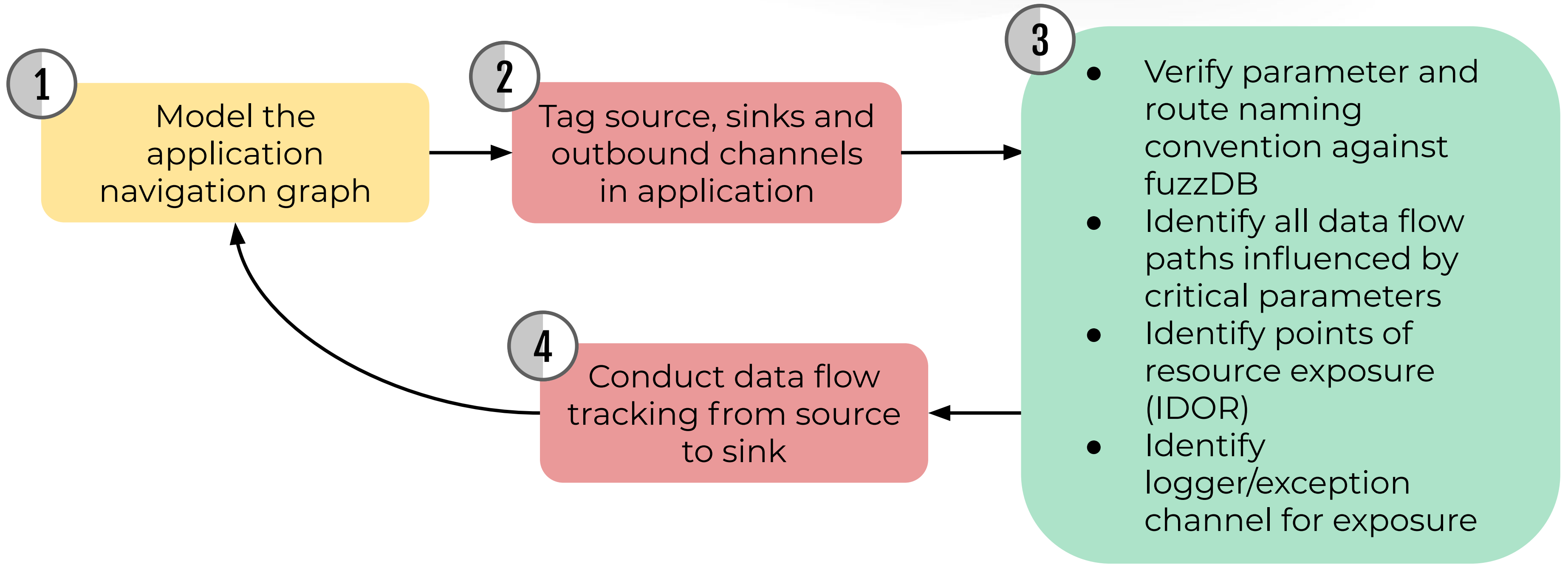
Deteting FLAWS

Can automated scanners detect logic flaws?

- Logic flaws are NOT pattern based

  ```
  find . -regex "[logic pattern]"
  ```
- Need to
  - Understand the business domain
  - Understand multi stage workflow/events and test if it can be manipulated

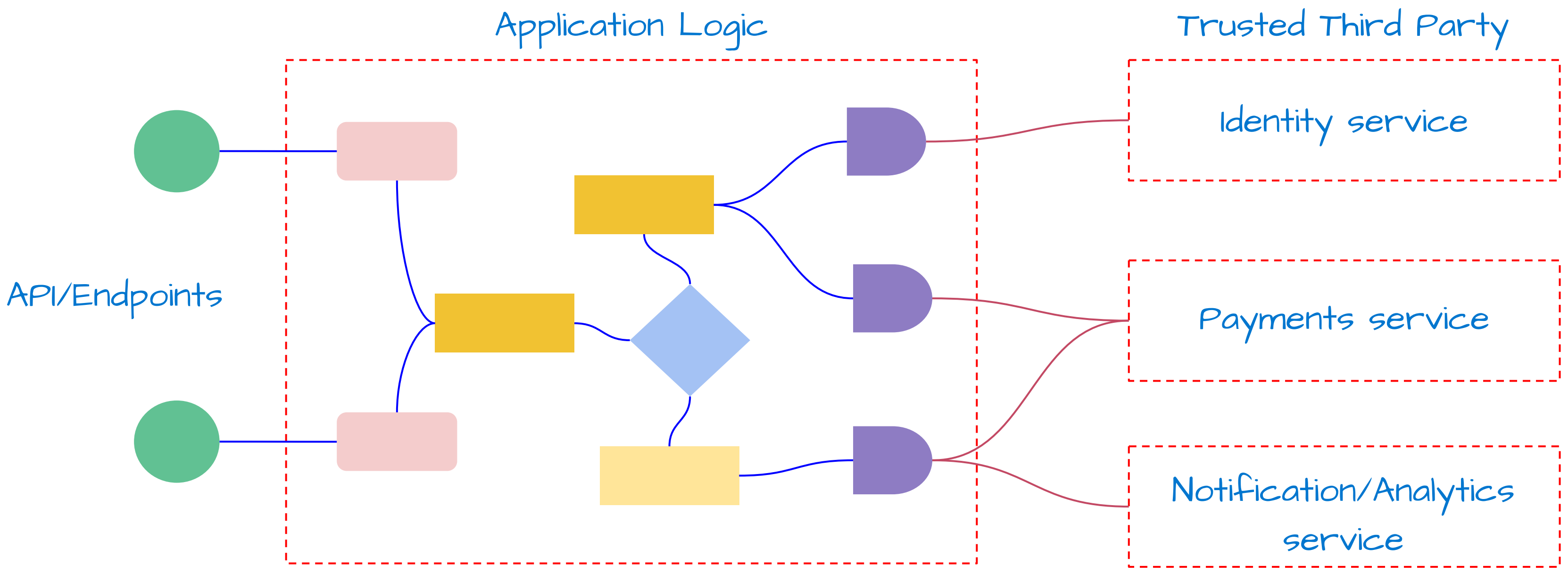
Defender mindset


1
Model the application navigation graph
2
Tag source, sinks and outbound channels in application
3
• Verify parameter and route naming convention against fuzzDB
• Identify all data flow paths influenced by critical parameters
• Identify points of resource exposure (IDOR)
• Identify logger/exception channel for exposure
4
Conduct data flow tracking from source to sink

DETECTING LOGIC FLAWS

- Document every aspect of your application design thoroughly
- Introduce Security Review and Analysis in early stages of architecture and development
- During Security Review reflect on every assumption made in design

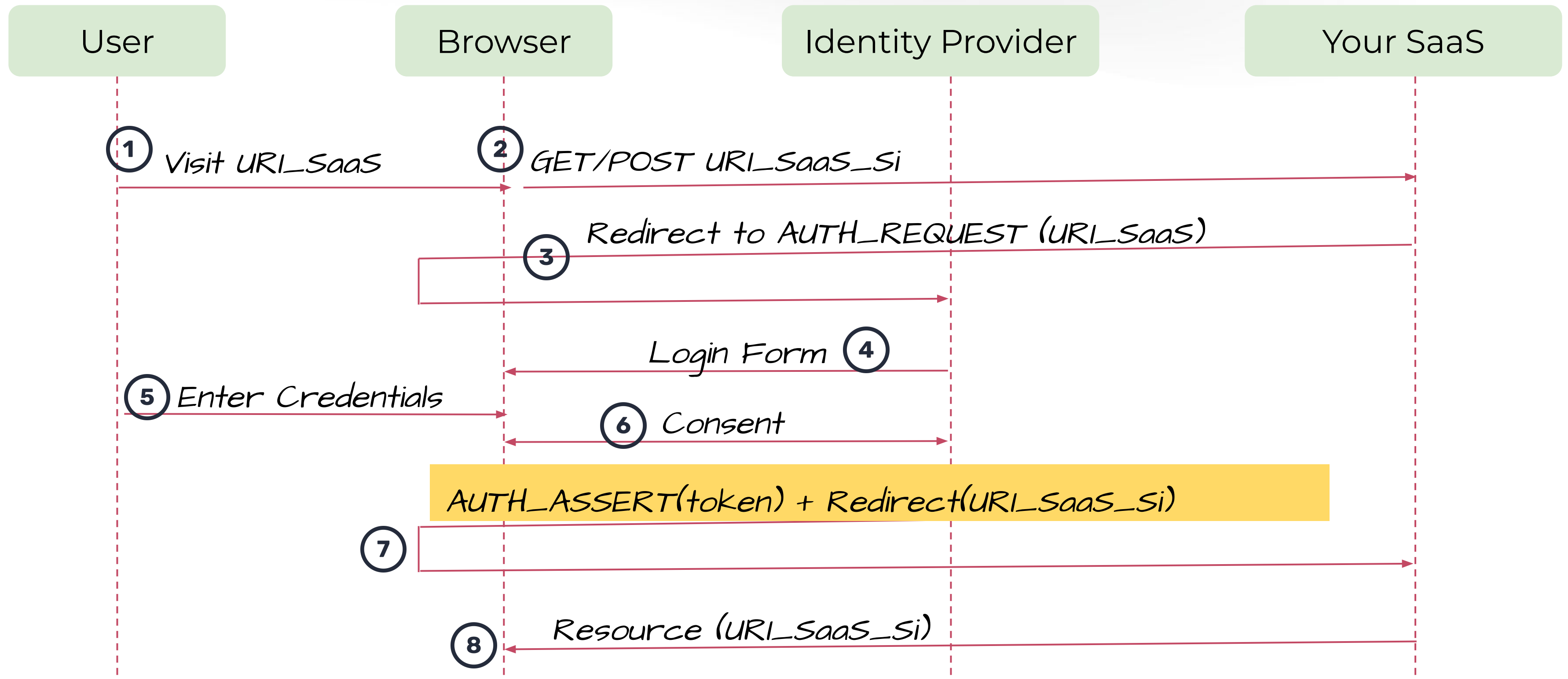
Understand your application design



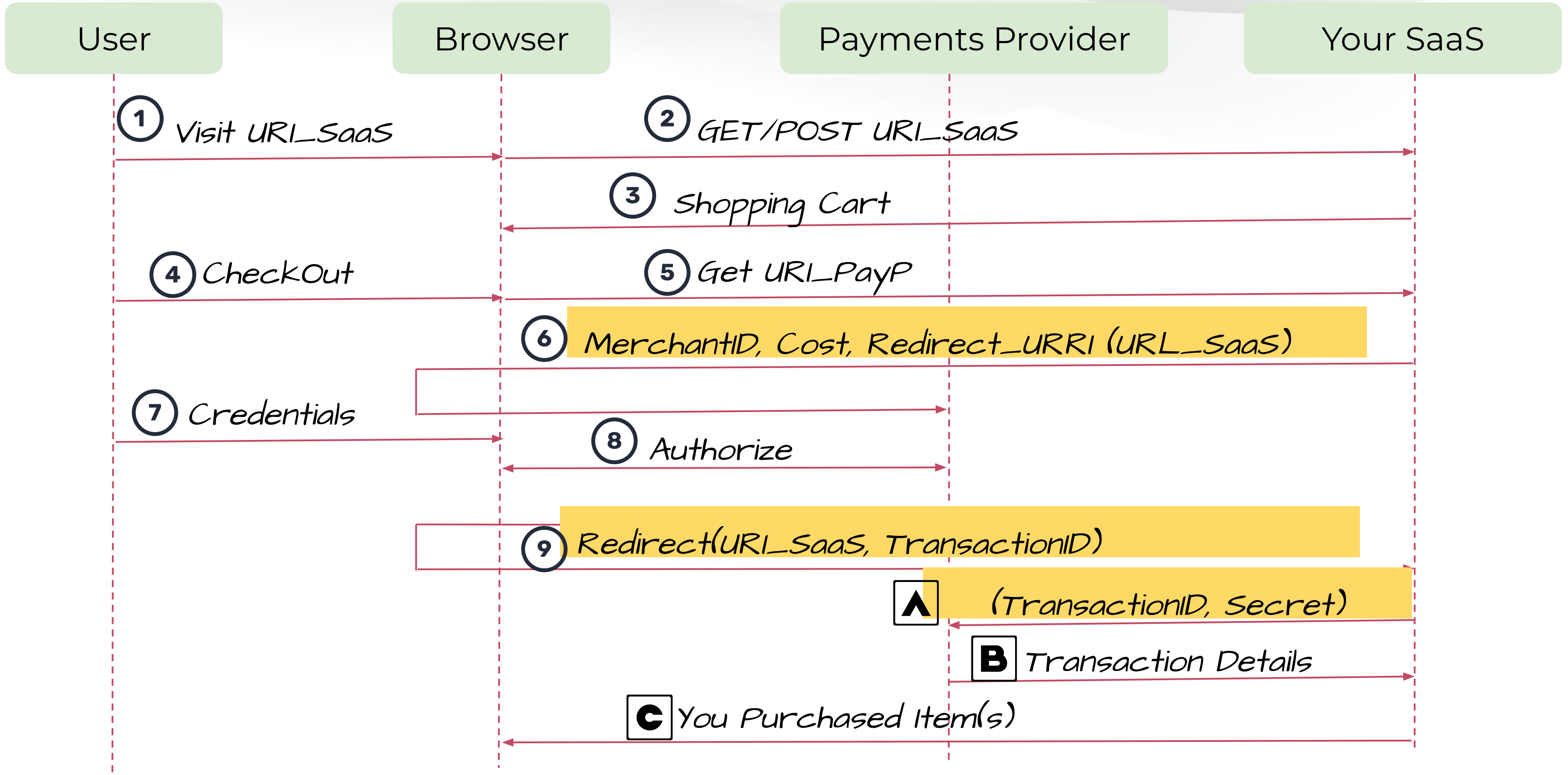
AVOIDING LOGIC FLAWS

- Detect endpoints/parameters that user/attacker can control
- Detect combinations of parameter(s) that impact flow in code (in and across services)
- Infer relationships between parameters
- Identify data that can be manipulated

THREAT MODELING (IDENTITY PROVIDER)



THREAT MODELING (PAYMENTS PROVIDER)



AVOIDING LOGIC FLAWS

- Detect if session token be repurposed across users (no affinity to user)
- Detect if individual steps in a multi step workflow be accessed directly
    - Running single step multiple times
    - Skipping steps
    - Running steps out of order

FINISH