



Rat's mobile methodology

[What is it?](#)

[The methodology](#)

[Picking a platform](#)

[Intigriti](#)

[Hackerone](#)

[Bugcrowd](#)

[Picking a program](#)

[VDP or paid?](#)

[Uncle Rat's attack strategy](#)

[Starting out](#)

[Manually exploring the application](#)

[PoC || GTFO](#)

[Exploring the requests in burp suite](#)

[What parameters do i test for what vulnerabilities?](#)

[Automated testing](#)

[General tips](#)

[Video's](#)

What is it?

Over time i ran into some issues when i was following other people's methodologies. I was testing like my mentors and my hero's but it never felt like their way of testing quite fitted my way of life and i never found any bugs doing that.

I am a stubborn rat. When i can't find a way i make my own. I've developed a methodology that i think minimizes dupes beause i focus on out-thinking the competition instead of being the first to find a new subdomain or asset and testing it. I love leftovers 🐭.

The methodology

Picking a platform

This is very important in my opinion. There are 3 major players that i focussed on with each having their own unique advantages and disadvantages. I personally will always recommend Intigrity but that's a personal preference and you will see why.


All these points are based on my opinion. Opinions can change and alter over time, make up your own mind. You need to pick the program that is right for you and i encourage you to fact check these bullet points and even investigate a little for your own.

You have several options here. You can either go with one of the major platforms or try your hand at some google dorking to find a good bug bounty program to fit your needs.

- Intigrity
- HackerOne
- Bugcrowd
- Synack (You need to apply)
- yeswehack
- Google dorking:

sushiwushi/bug-bounty-dorks

List of Google Dorks to search for companies that have a responsible disclosure program or bug bounty program which are not affiliated with known bug bounty platforms such as

 <https://github.com/sushiwushi/bug-bounty-dorks>



Intigrity

This is of course my preferred platform as i have been hunting on it for a while so i'll have a bit more to inform about when it comest to this platform.

+ Positive items

- Less crowded
- Amazing triagers that will go out of their way to help you

- After reporting a few good valid bugs, you can get invited to the Initigrity slack workspace where the community manager and triagers are always there and ready to help with your questions
- Wide range of programs to pick from (mobile, web, vps hosters,...)
- They do not punish you for reporting an invalid bug unlike other platforms where you get negative karma

— Negative Items

- They have a smaller amount of programs when you compare them to other big platforms such as hackerone or bugcrowd
- Most programs are in Dutch or some other EU language. This can be an advantage however if you just install a translate plugin. It's a barrier and every barrier drops off some other hackers so we can use these to our advantage if we are willing to go a bit further.
- There could be more public programs. You would do well picking a VPD (less competition) over a paid program and reporting a few bugs. This will get you invites to private programs which Intigrity has plenty off.

Hackerone

I don't have much experience with this platform so i won't have as much to say but i did form an opinion in my time using the platform.

+ Positive items

- The host CTFs that give you private invites when you find flags
- They have a huge selection of programs
- They have great triagers who will help you out when you write an invalid report by adding useful comments
- They organise regular event to improve the community knowledge

— Negative Items

- If you create an invalid report, you will get negative karma, get too much negative karma and you will have to prove yourself in a trial period where you

can't submit to some programs and have to wait 30 days before you can make any submission at all

- Since this is the biggest bounty platform out there, there is a lot of competition but there is also a lot more programs to distribute the competing hackers upon

Bugcrowd

I have not hacked much on bugcrowd either so i can't tell you much about them, here's what i can tell.

+ Positive items

- They have a huge selection of programs
- They have great triagers who will help you out when you write an invalid report by adding useful comments
- They have the bugcrowd academy where you can learn about all the vulnerabilities

— Negative Items

- Since this is the biggest bounty platform out there, there is a lot of competition but there is also a lot more programs to distribute the competing hackers upon

Picking a program

Picking a program is like picking a shoe, make sure you pick a program that fits you and that you enjoy hacking on. For the purposes of this chapter, i always try to go for applications where i can create users with different levels of privilege. I love testing on those programs because they contain a fair bit of business logic and we all know where this can lead you to. Examples of these programs are:

- <https://app.intigriti.com/programs/suivo/suivoweb/detail>
- <https://app.intigriti.com/programs/sentiance/sentiance/detail>
- <https://app.intigriti.com/programs/hoplr/hoplr/detail>

I also have some general tips for you as well. A lot of people will have the general tendency to go for a target with a very broad scope. This is a very bad idea if you ask me, let me explain why.

Recon only serves to help you find a target where you can apply your main methodology. After the recon you still need to hack and this is what a lot of people forget. After you spend hours doing your recon, all that work will just be to get you started. This is why i would recommend you try and fine tune your attack strategy on main apps first (No *.target.com domains)

VDP or paid?

VDP - vulnerability disclosure program (points or swag, no pay)

I would highly recommend people to start out on VDP programs, i have several reasons for this recommendation:

- VDP programs will be less crowded, no pay = less hackers
- VDP programs might be less hardened due to their nature. Since they do not require payouts, they can spend a bit less on bug bounties and risk a bit more
- Points will get you invites to private programs, these will contains less hackers and they might be paid

Fine tune your strategy and i assure you that the money will follow but you need to give it time! Rome was not built in one day either.

Uncle Rat's attack strategy

Starting out

My mobile testing strategy is very similar to my web application testing strategy. I don't really participate in static code analysis and rather keep myself entertained with the dynamic testing. (running the application and analyzing the communication with the server.

As far as methodologies, we can opt to either focus on the API that the mobile application is interacting with or the security model of the application itself and i always prefer the API interactions since i am convinced that mobile applications are reasonably secure due to the nature of mobile operating systems like android.

That being said, vulnerabilities may still exist in the security models of the apps but i chose not to focus on those.

When you first start attacking your target i want you to keep the following in mind:

- Start out with registering an account if possible or editing your account, make sure you include the following:
 - Where-ever possible, insert both XSS attack vectors. This will be carried throughout your manual testing and might expose integrations issues that are hard to spot otherwise
 - Attack vector example: ``
 - Insert that attack vector into every possible field, even mobile applications can be built on webviews which can contain XSS attack vectors
 - Create your own attack vector!
- After registering, explore the website like any normal user would
- Make a mindmap of the functionality
- Keep Burp suite proxy open in the background

Manually exploring the application

So now we have picked a platform, we have a program, now we finally get to the juicy part.

When i approach a new target the very first thing i do is get to know my target. I need to know all the functionality that's available and i do that by registering an account and using the application like a normal user would. while i do this i pay special attention to the privilege levels that are available and the functionality that was added on later like import or export functions. Any function that's added on top of another function brings an integration point because the two functions need to communicate. Often the original feature might be well secured but the additional functionality might not contain the same security features as the original.

While i explore manually i will have burp open and my scope set properly, this will populate the site map while i use the application. I need that for later. I can also read the manual (If there is none, it try google dorking). Reading the manual is very important, we are bug bounty hunters and we are at war with out targets. The manual is like a blue print of the application and in war information is power. As a last thing we can also read the api documentation if there is any or read the swagger docs.

I'll make sure to insert a 🌿 XSS attack vector as soon as i create an account wherever possible to give myself the biggest chance of popping an XSS attack later down the line when i use the application.

This whole process of exploring my target and information gathering takes a couple of days easily. I want to know the enemy i am at war with. If it helps you, create a mindmap of your target using xmind.


XMind - Full-featured mind mapping and brainstorming tool.

XMind is the most professional and popular mind mapping tool. Millions of people use XMind to clarify thinking, manage complex information, brainstorming, get work organized, remote and work from

🔗 <https://www.xmind.net/>

Mind

Ideas Grow on Trees.



PoC || GTFO

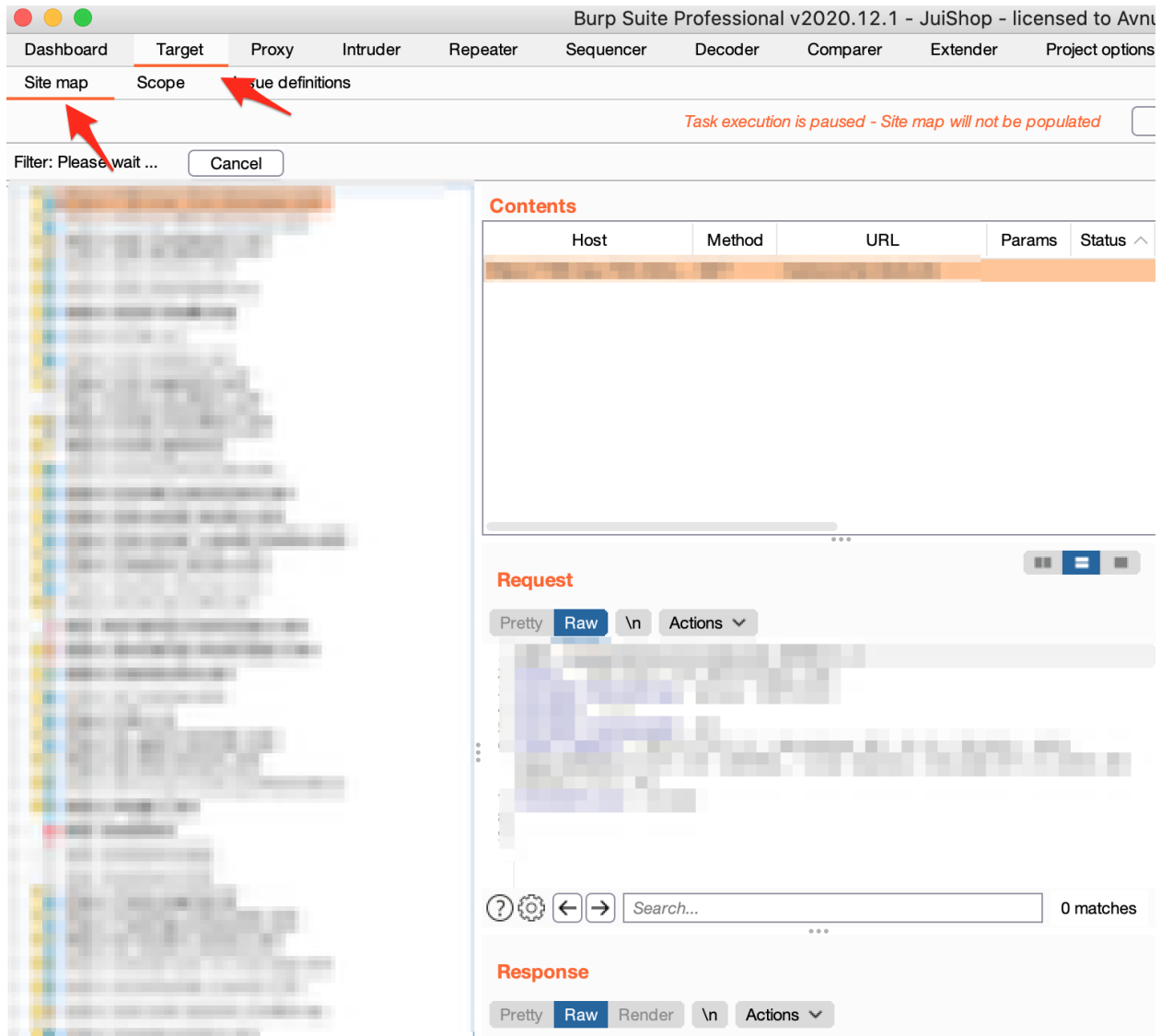
When you are doing your initial recon, and especially if you use burp suite pro, you might find a lot of alarms going off. Say for example you use Burp Suite Pro, it might return some issues that it's found. Do not get too excited though, these vulnerabilities are often false positives. It is very important that you confirm ANY finding from ANY tool manually.

In bug bounties we have a saying "PoC || GTFO" which means you need to prove impact or Get The F** out. We say this because it matters, some reports might be totally unexploitable or not even impactful for the companies.

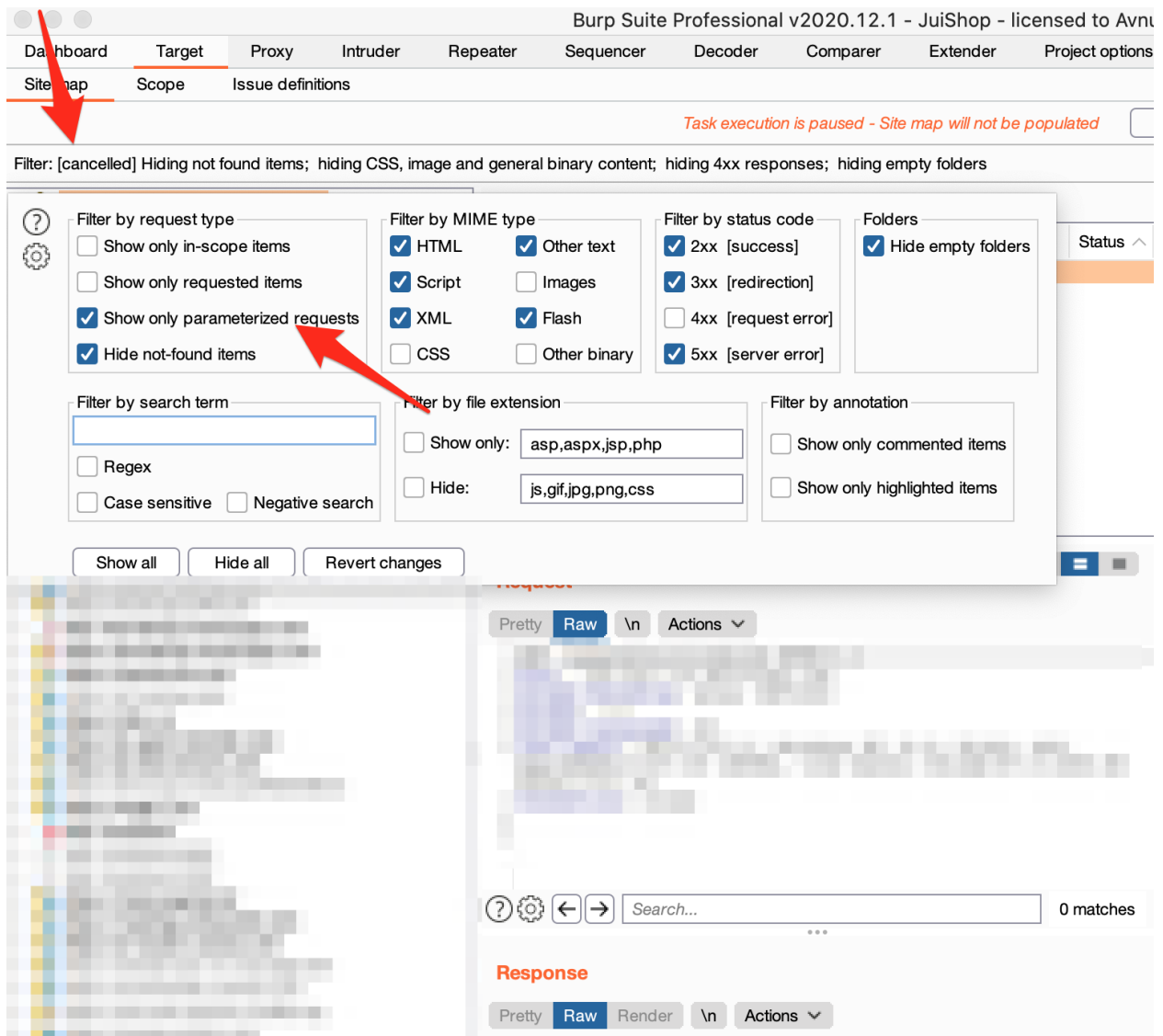
Exploring the requests in burp suite

After i know how my applications works, i am ready and armed with knowledge that will help me recognise vulnerabilities faster. I need to know how my

application is supposed to work before i can start attacking it and now that we have a surface level map of our application, we need to go deeper. It's time to go to burp where our site map has been filling.



Burp suite has some amazing tricks in store for us. We can apply a filter so burp will only show us the parameterised requests in the site map. Navigate to the site map tab and click the big square box at the top to apply a filter and now click the checkbox next to "show only parametrised requests".



As a bug bounty my next step would be to go through these requests and see if i can find any interesting parameters i can tamper with. It comes down to common sense which parameters you investigate manually but it goes without saying parameters like "view=week" in a calender are not that interesting to us. We are looking for requests with parameters that trigger some business logic.

An example i am going to give is facebook, we all know facebook and facebook has several features we can use. We will focus on making a post, imagine we have the following requests in our burp window:

- POST /CreatePost
- {Content=text}

- GET /comments
- GET /post
- POST /comment
 - {Content=text}
- POST /Like
 - {PostID=id}
- POST /Unlike
 - {PostID=id}

When someone makes a POST /CreatePost request we can try several things like XSS, SSTI,... on the Content parameter those are certainly interesting attack surfaces we can explore.

```

1 POST /api/CreatePost/ HTTP/1.1
2 Host: ferretshop.herokuapp.com
3 Accept-Encoding: gzip, deflate
4 Accept: */*
5 Accept-Language: en
6 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/
7 Connection: close
8 Cookie: io=35s6eZ502ts9uGOPAAAC; language=en; welcomebanner_status=
  Yy9pbWFnZXNMvdXBsb2Fkcy9kZWZhdWx0LnN2ZyIsInRvdHBTZWNYZXQiOiIiLCJpc
9 Content-Length: 36
10
11 {
12   Content: "The XSS Rat is amazing"
13 }

```

But those might have a bigger chance of duplicates because they are obvious. If you like a post however, a POST /Like request is being sent. If you can manipulate how many likes a post gets, that would be a valid vulnerability so let's investigate this call further.

The call was being sent when we clicked the like button and then the button turned blue and when we clicked it again a POST /Unlike request gets sent to the

server so the amount of likes is added by 1 if you click the like button and subtracted by one if you click it again.

```
1 POST /api/Like/ HTTP/1.1
2 Host: ferretshop.herokuapp.com
3 Accept-Encoding: gzip, deflate
4 Accept: */*
5 Accept-Language: en
6 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit,
7 Connection: close
8 Cookie: io=35s6eZ502ts9uGOPAAAC; language=en; welcomebanner_status:
9 Content-Length: 36
10
11 {
12   PostId:1
13 }
```

```
1 HTTP/1.1 200 OK
2 Server: Cowboy
3 Connection: close
4 X-Powered-By: Express
5 Access-Control-Allow-Origin: *
6 X-Content-Type-Options: nosniff
7 X-Frame-Options: SAMEORIGIN
8 Content-Type: text/html; charset=utf-8
9 Vary: Accept-Encoding
10 Date: Tue, 05 Jan 2021 00:42:31 GMT
11 Via: 1.1 vegur
12 Content-Length: 3396
13
14 {
15   likes:1
16 }
```

```

1 POST /api/Unlike/ HTTP/1.1
2 Host: ferretshop.herokuapp.com
3 Accept-Encoding: gzip, deflate
4 Accept: */*
5 Accept-Language: en
6 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit,
7 Connection: close
8 Cookie: io=35s6eZ502ts9uGOPAAAC; language=en; welcomebanner_status:
9 Content-Length: 36
10
11 {
12   PostId:1
13 }

```

```



1 HTTP/1.1 200 OK
2 Server: Cowboy
3 Connection: close
4 X-Powered-By: Express
5 Access-Control-Allow-Origin: *
6 X-Content-Type-Options: nosniff
7 X-Frame-Options: SAMEORIGIN
8 Content-Type: text/html; charset=utf-8
9 Vary: Accept-Encoding
10 Date: Tue, 05 Jan 2021 00:42:31 GMT
11 Via: 1.1 vegur
12 Content-Length: 3396
13
14 {
15   likes:0
16 }

```


A smart hunter would send the POST /Like request to the repeater in burp and try to send it a couple of times and see if the amount of likes keeps going up. If it does, we have a vulnerability.




What parameters do i test for what vulnerabilities?

I will try to go through all the requests and see if I can find any interesting parameters. A parameter looks interesting to me in the following cases:

- If it will seem to interact with the logic of the webapplication (This is why it is so important to know the application properly)
- Any parameters that seems to contain a URL. If the target resolves the URL that i enter, it might be vulnerable to  SSRF
- Any parameter that appears to grab files from the local file system such as images being stored on the HDD of the webserver and being refered to as image=file.png. This might be vulnerable to LFI/RFI
- Any  CSRF parameter
- I will try to save my account settings, personal settings and any other settings i can find and look for hidden parameters


Automated testing

This will consist of me testing for  IDOR but my methods differ. I try to pick a target where i can invite users within the account and I can also create user of different priviledge levels. This way i can:

- Create two accounts and test for  IDOR
- Create two users in one account of the same priviledge levels and test for  IDOR
- Create two users in one account of different priviledge levels and test for  BAC

Make sure you test ALL the priviledge levels and if you can create your own, make sure to do so.

Some targets will call this "roles" others might call it "groups" but as long as you can set different levels of priviledges; you will be fine.

After all is said and done i'll try to fuzz some endpoints for  Command injection and i'll try to identify some other endpoints to attack.

General tips

- Get to know your application, spend some time on using it normally but keep burp open

- Start with XSS as early as possible if in scope, insert them into every field you see
- Create your own fuzzing lists
- Expand this list with your own findings
- VDP over paid if you want less competition
- PoC or GTFO
- Prove your impact

Video's

<https://youtu.be/Lxc-xb6xUk4>

<https://youtu.be/4q22s743hrl>