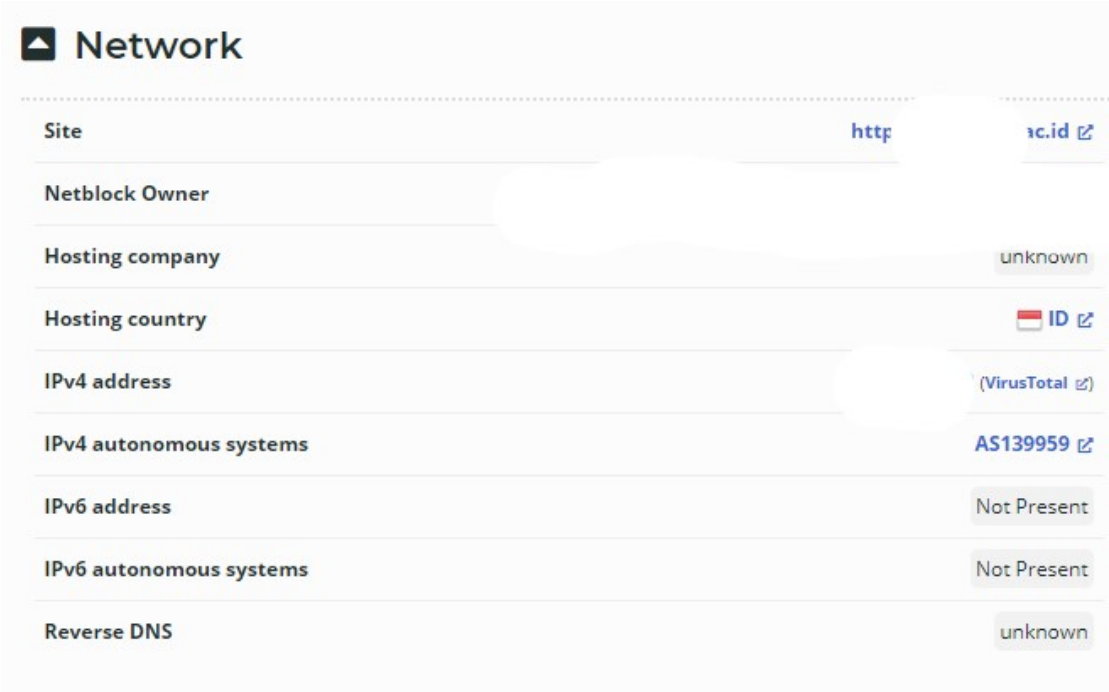
BAB IV

HASIL PENELITIAN

4.1 Information Gathering

Pengumpulan data pada *website* SIM xxx dimulai dengan mencari informasi yang tersedia pada lamat *website.* Pencarian Informasi dimulai dengan mencari IP *Adress* pada target yang dituju menggunakan *tools online netcraft*. Dan berikut hasil yang didapat.

4.1.1 Netcraft



Gambar 4. 1 Netcraft IP

Untuk mencari alamat IP dapat dilakukan dengan berbagai cara, salah satunya dengan menggunakan *tools online* seperti yang dilakukan pada gambar diatas. dengan memasukkan Alamat_*Website* akan mendapatkan berbagai macam informasi

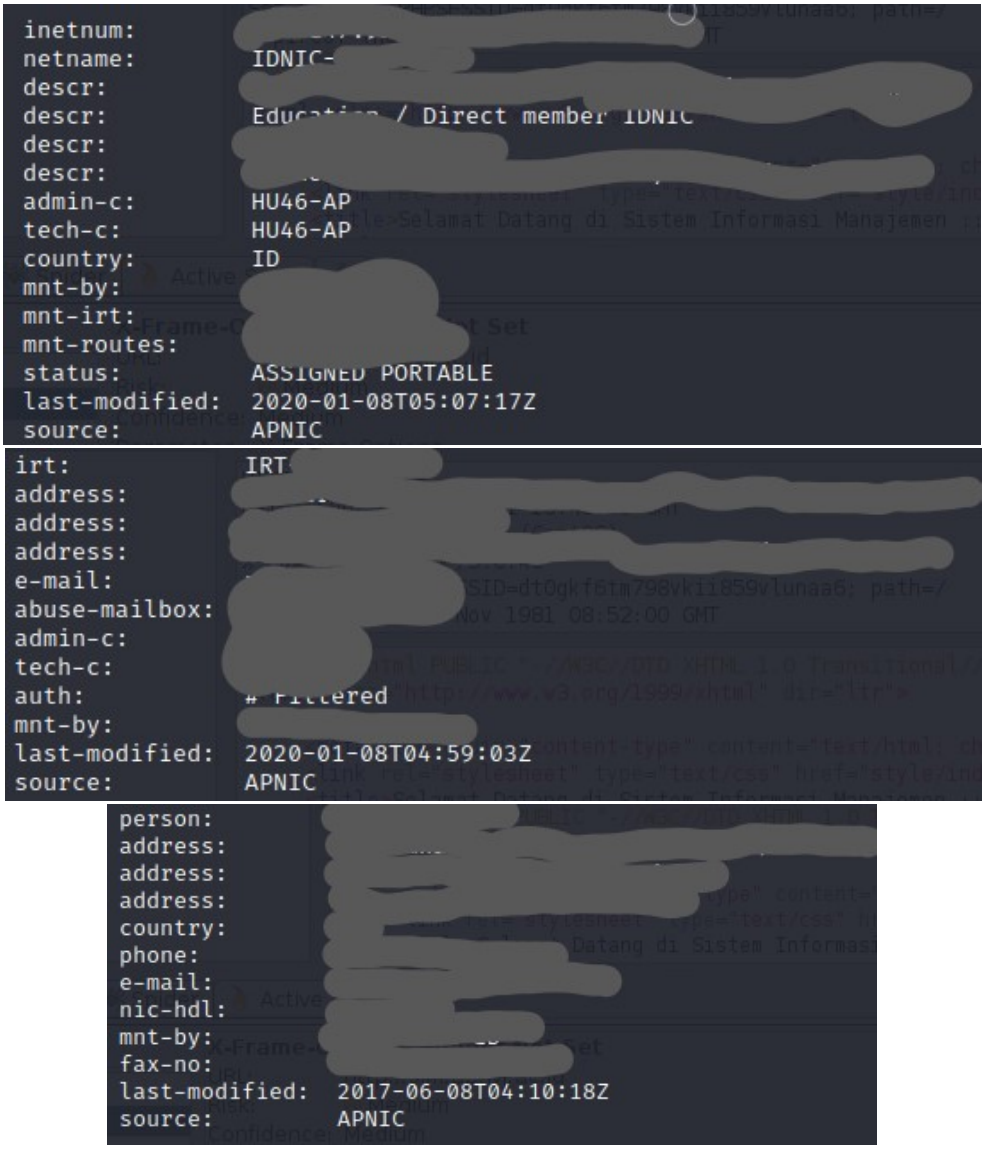
terkait *website* tersebut. Hasil IP yang didapat dari hasil menggunakan *tools netcraft* diatas pada SIM xxx yaitu 103.xxx.xx.x

4.1.2 *Whois*

Berikut *command Whois* yang digunakan dalam proses *Information Gathering* untuk melakukan pengecekkan data pemilik dari nama *domain* atau Alamat IP. Dengan alat ini dapat mengetahui siapa pemilik nama *domain* yang sesungguhnya. Seperti alamat *hosting*, alamat *server*, *email*, dan informasi lainnya terkait *website* SIM xxx

Whois Alamat_Website

Dan berikut hasil yang didapat dari *command* diatas



Gambar 4. 2 Hasil Whois

4.1.3 *httprint*

Proses *fingersprint webserver* menggunakan *Httprint* diatas bertujuan untuk menemukan versi dan jenis *server web* yang sedang berjalan untuk menentukan kerentanan yang diketahui dan eksploitasi yang sesuai. Berikut *command httprint* yang digunakan dalam proses *Fingerprint Webserver*

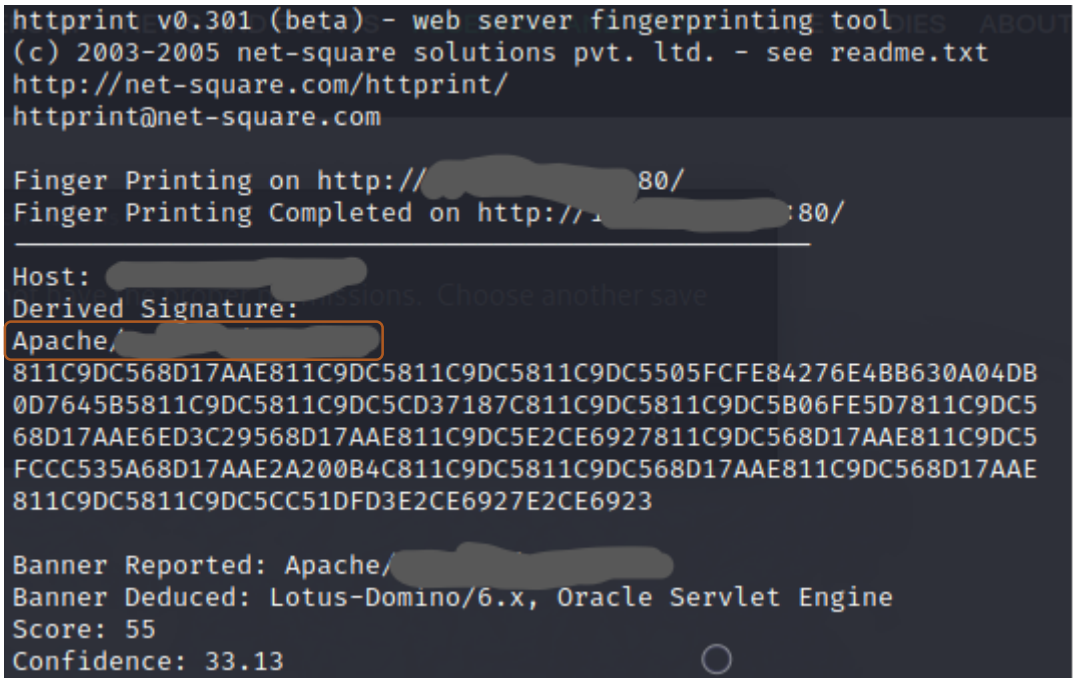
Sudo httprint -h http://103.xxx.xx.x -s signatures.txt

Ket :

-h : target berupa alamat ip

-s : file yang berisi tanda tangan sidik jari HTTP

Dan berikut hasil yang didapat dari *command* diatas



Gambar 4. 3 Hasil httprint

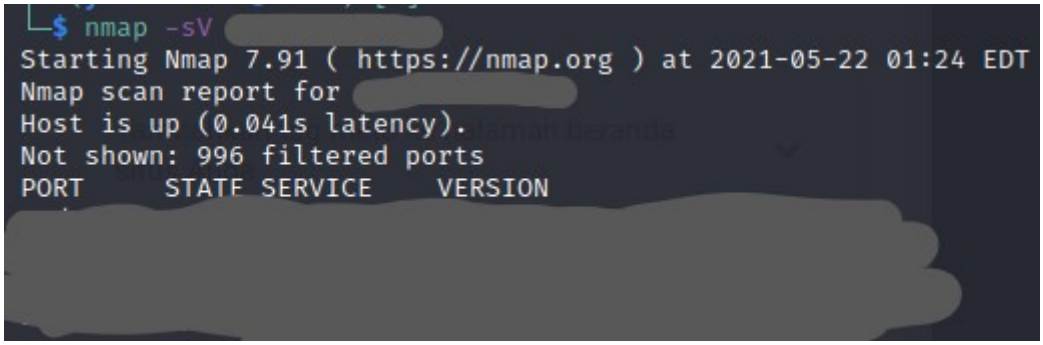
Dari hasil *fingerprint* yang dilakukan diatas informasi didapat adalah jenis *server* dan versi yang digunakan merupakan *Apache*/2.2.15 (CentOS)

4.1.4 *Nmap*

Network Mapping merupakan sebuah *tool open source* untuk eksplorasi dan audit keamanan jaringan yang akan dijadikan landasan untuk melakukan eksploitasi melalui *open port* yang tersedia dari hasil N*map* berikut.

Nmap -sV 103.xxx.xx.xx

Dan berikut hasil yang didapat dari *command* NMAP diatas



Gambar 4. 4 hasil Nmap

Hasil dari *scanning* N*map* diatas terdapat *open port* yang ditemukan pada *server* SIM xxx, yaitu *Port* 22 atau biasa disebut SSH, *Port* 80 atau biasa disebut HTTP, *Port* 443 atau biasa disebut SSL, dan *Port* 5432 atau biasa disebut *Postgresql*

4.1.5 *Whatweb*

Whatweb -v -a 3 Alamat_Website

Ket :

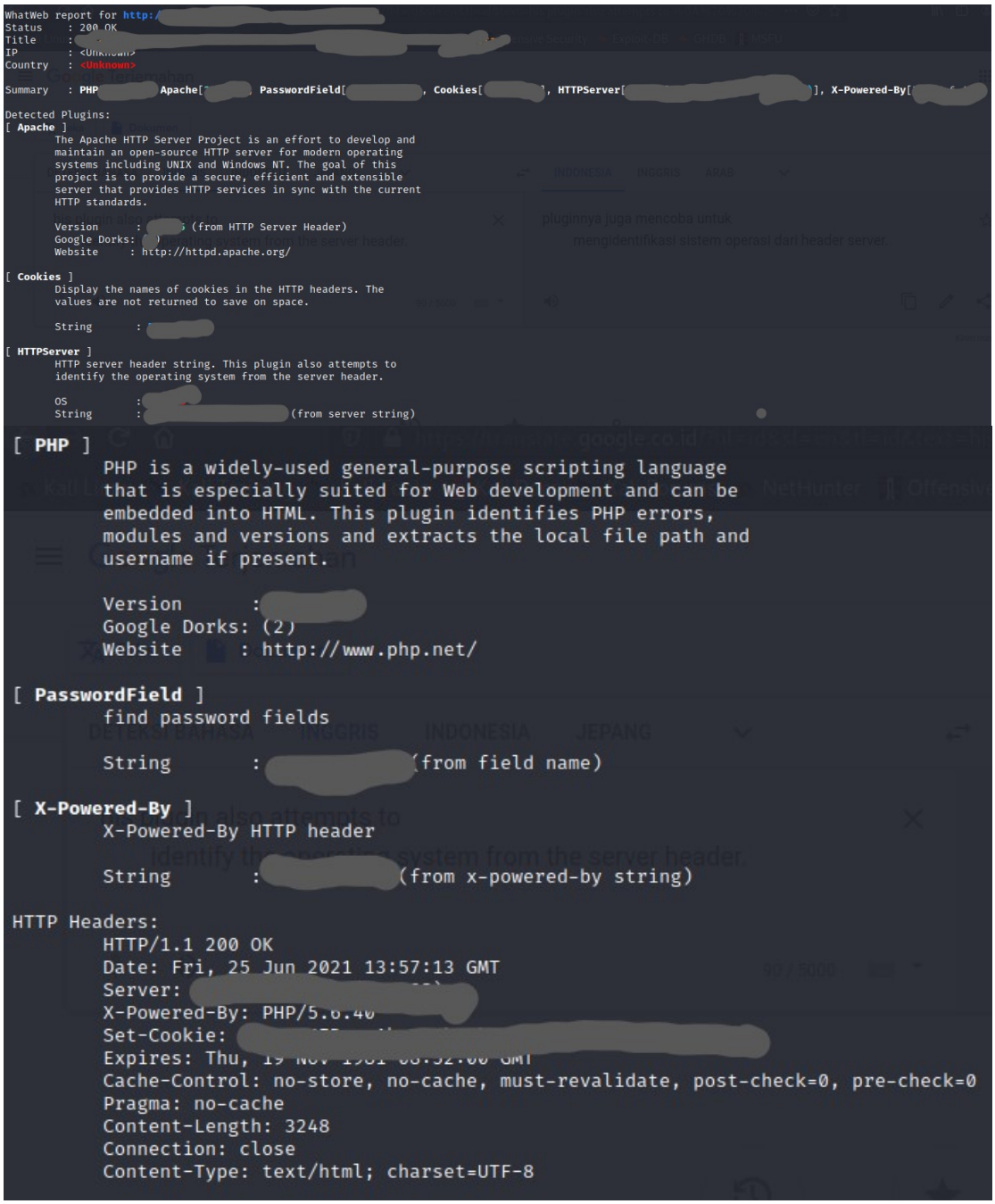
-v : *Output verbose* mencakup deskripsi *plugin*

-a : tingkat agresi *set level*

1. *Stealthy*. membuat satu permintaan HTTP per target dan juga mengikuti pengalihan
3. *Agresive*. jika *plugin* cocok, permintaan tambahan akan dibuat
4. *Heavy*. membuat banyak permintaan HTTP per target. URL dari semua *plugin* dicoba

Command *whois* diatas digunakan untuk mencari jenis kerangka kerja aplikasi web/CMS dari header HTTP, *Cookie*, *Source Code*, File, dan Folder tertentu dari suatu *Website*

Dan berikut hasil yang diperoleh dari *command whatweb* diatas.



Gambar 4. 5 Hasil Whatweb

Hasil yang didapat dari gambar diatas sebagai berikut. PHP[5.6.40] yaitu bahasa pemograman yg digunakan beserta versinya, *Apache*[2.2.15] yaitu jenis *server* yang digunakan, *PasswordField*[*txtPassword*] yaitu *format text password* yang digunakan, *Cookies*[PHPSESSID] yaitu tampilan nama

cookie di *header* HTTP, HTTP*Server*[centOS][Apache/2.2.15 (CentOS)] yaitu percobaan mengidentifikasi jenis server yang digunakan, X-*powered-By*[PHP/5.6.40] dari X-*powered-by string.*

4.1.6 Kesimpulan Information Gathering

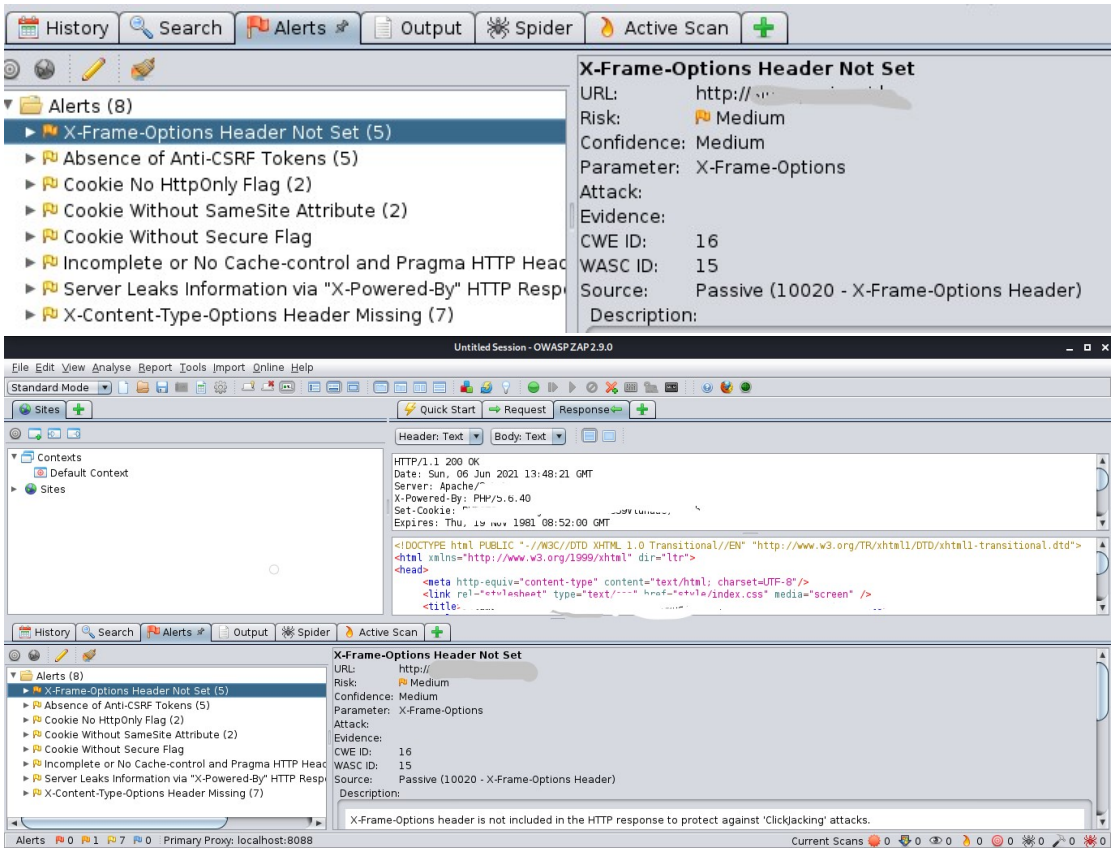
No	Proses	Hasil Akhir
1	*Netcraft*	IP *Address*, yaitu 103.xxx.xx.x
2	*Whois*	Alamat *Hosting, Server, Email*, dan Informasi lainnya terkait *Website* SIM xxx, yaitu Jl xxx, Jakarta xxx, xxx.net.id.
3	HTT*Print*	*Apache*/2.2.15(*Cent*OS)
4	N*map*	*Open Port* : *Port* 22/SSH, *Port* 80/HTTP, *Port* 5443/*Postgre*SQL, *Port* 443/SSL
5	*Whatweb*	HTTP *Header*

Tabel 4. 1 Kesimpulan Information Gathering

4.2 scanning

Scanning vulnerabillity dilakukan guna mencari celah kerentanan/keamanan pada website menggunakan sebuah tools yang dirancang khusus untuk mencari celah

keamanan berdasarkan level resiko dari setiap temuan celah. Dan berikut hasil *Scanning vulnerabillity* menggunakan *tools* OWASP ZAP versi 2.9.0



Gambar 4. 6 ***Scanning OWASP ZAP***

Vulner/celah keamanan yang diambil adalah yang *medium risk* dari hasil *scanning vulnerabillity* diatas adalah tidak adanya header X-*Frame-Options* dalam *respons* HTTP untuk melindungi dari serangan *Clickjacking* sehingga serangan *Clickjacking* dapat saja dilakukan dari luar terhadap *website* SIM XXX.

4.3 Testing

4.3.1 Injection

Injection attack yang dilakukan menggunakan *package* SQL*Map* yang tersedia pada *Kali Linux* bertujuan untuk mendapatkan informasi lengkap yang ada pada *database* suatu *website*. Dan berikut command yang digunakan untuk melakukan *injection attack*.

sqlmap -r request.txt

Keterangan :

-r : mencari *file text* yang akan digunakan sebagai bahan untuk eksploitasi

Request.txt : merupakan text yang berisi HTTP *post form* yang didapat dari proses *Scanning form* HTTPS *login* menggunakan *Burproxy.* Dan berikut isi *Request*.txt tersebut.

```
<?xml version="1.0"?>
<!DOCTYPE items [
<!ELEMENT items (item*)>
<!ATTLIST items burpVersion CDATA "">
<!ATTLIST items exportTime CDATA "">
<!ELEMENT item (time, url, host, port, protocol, method, path, extension, request, status, responselength, mimetype, response, comment)>
<!ELEMENT time (#PCDATA)>
<!ELEMENT url (#PCDATA)>
<!ELEMENT host (#PCDATA)>
<!ATTLIST host ip CDATA "">
<!ELEMENT port (#PCDATA)>
<!ELEMENT protocol (#PCDATA)>
<!ELEMENT method (#PCDATA)>
<!ELEMENT path (#PCDATA)>
<!ELEMENT extension (#PCDATA)>
```

```
<!ELEMENT request (#PCDATA)>
<!ATTLIST request base64 (true|false) "false">
<!ELEMENT status (#PCDATA)>
<!ELEMENT responselength (#PCDATA)>
<!ELEMENT mimetype (#PCDATA)>
<!ELEMENT response (#PCDATA)>
<!ATTLIST response base64 (true|false) "false">
<!ELEMENT comment (#PCDATA)>
]>
<items burpVersion="2020.9.1" exportTime="Fri May 28 12:59:45 EDT 2021">
  <item>
    <time>Fri May 28 12:58:50 EDT 2021</time>
    <url><![CDATA[https://Alamat_Website.AC.ID/front/gate/index.php]]></url>
    <host ip="103.xxx.xx.x">Alamat_Website.AC.ID</host>
    <port>443</port>
    <protocol>https</protocol>
    <method><![CDATA[POST]]></method>
    <path><![CDATA[/front/gate/index.php]]></path>
    <extension>php</extension>
    <request
base64="true"><![CDATA[UE9TVCAvZnJvbnQvZ2F0ZS9pbmRleC5waHAgSFRU
UC8xLjENCkhvc3Q6IHNpbS51cG52ai5hYy5pZA0KVXNlci1BZ2VudDogTW96a
WxsYS81LjAgKFgxMTsgTGludXggeDg2XzY0OyBydjo3OC4wKSBHZWNrby8y
MDEwMDEwMSBGaXJlZm94Lzc4LjANCkFjY2VwdDogdGV4dC9odG1sLGFwc
GxpY2F0aW9uL3hodG1sK3htbCxhcHBsaWNhdGlvbi94bWw7cT0wLjksaW1hZ2U
vd2VicCwqLyo7cT0wLjgNCkFjY2VwdC1MYW5ndWFnZTogZW4tVVMsZW47cT
0wLjUNCkFjY2VwdC1FbmNvZGluZzogZ3ppcCwgZGVmbGF0ZQ0KQ29udGVud
C1UeXBlOiBhcHBsaWNhdGlvbi94LXd3dy1mb3JtLXVybGVuY29kZWQNCkNvb
nRlbnQtTGVuZ3RoOiA0NQ0KT3JpZ2luOiBodHRwczovL3NpbS51cG52ai5hYy5p
ZA0KQ29ubmVjdGlvbjogY2xvc2UNClJlZmVyZXI6IGh0dHBzOi8vc2ltLnVwbnZq
LmFjLmlkL2Zyb250L2dhdGUvaW5kZXgucGhwDQpDb29raWU6IFBIUFNFU1NJ
RD1hdHZmOHBqN3Y3MWsyNWU3cHYxbXJqbmlwNw0KVXBncmFkZS1JbnNl
```

```
Y3VyZS1SZXF1ZXN0czogMQ0KDQp0eHRVc2VySUQ9YWRtaW4mdHh0UGFzc
3dvcmQ9MTIzNCZzdWJtaXQ9TG9naW4=]]></request>
      <status></status>
      <responselength></responselength>
      <mimetype></mimetype>
      <response base64="true"></response>
      <comment></comment>
   </item>
</items>
```

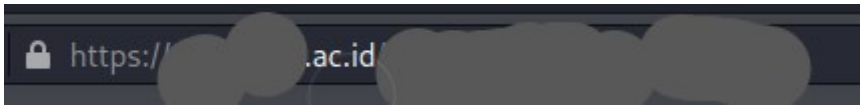
Dan berikut hasil yang didapat dari *injection testing* menggunakan *command* sql*map* diatas.

```
[13:25:07] [WARNING] heuristic (basic) test shows that POST parameter 'txtUs
erID' might not be injectable
[13:25:28] [WARNING] POST parameter 'txtUserID' does not seem to be injectab
le
[13:25:28] [WARNING] POST parameter 'txtUserID' does not seem to be injectab
le
[13:25:28] [WARNING] heuristic (basic) test shows that POST parameter 'txtPa
ssword' might not be injectable
[13:25:47] [WARNING] POST parameter 'txtPassword' does not seem to be inject
able
[13:25:48] [WARNING] heuristic (basic) test shows that POST parameter 'submi
t' might not be injectable
[13:26:06] [WARNING] POST parameter 'submit' does not seem to be injectable
[13:26:06] [CRITICAL] all tested parameters do not appear to be injectable.
Try to increase values for '--level'/'--risk' options if you wish to perform
 more tests. If you suspect that there is some kind of protection mechanism
involved (e.g. WAF) maybe you could try to use option '--tamper' (e.g. '--ta
mper=space2comment') and/or switch '--random-agent'
[13:26:06] [WARNING] your sqlmap version is outdated

[*] ending @ 13:26:06 /2021-05-28/
```

Gambar 4. 7 Injection attack with SQLMap

Hasil *testing injection* menggunakan sql*map* diatas menghasilkan *"all tested parameters do not appear to be injectable"* yang dapat disimpulkan bahwa celah untuk melakukan *injection attack* yang ditemukan pada *website* SIM xxx tidak ada.



Gambar 4. 8 URL SIM Without Parameter ID

Dikarenakan pada URL website SIM tidak ada parameter ID seperti contoh gambar diatas yang bisa digunakan sebagai landasan untuk melakukan *injection attack*, *Command Injection* yang digunakan merupakan command untuk menyerang *database* menggunakan *post parameter* yang didapatkan dari hasil *scanning menggunakan* OWASP ZAP yaitu sebagai berikut.

4.3.2 Broken Authentication

Untuk melakukan exploit *Broken Authentication* atau *password attack* pada halaman website menggunakan *tools hydra* pada kali linux, berikut *command* yang dijalankan.

Hydra Alamat_Website -V -L username.txt -P cobapassword.txt https-post-form "/front/gate/index.php:txtUserID=^USER^&txtPassword=^PASS^:Username atau Password anda salah"

keterangan :

-V : *verbose detail* deskripsi proses *exploit*

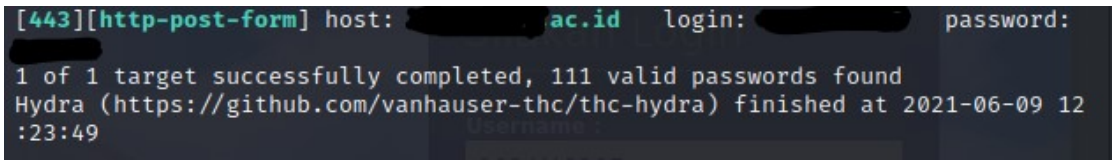
-L : mencari *username file* yang ada pada direktori *file*

-P : mencari *password file* yang ada pada direktori *file*

https-*post-form* : didapat dari metode *post* dari *scanning* menggunakan *Burpsuite*/OWASP ZAP

Command *hydra* yang digunakan untuk melakukan *burtforce* menyesuaikan dengan spesifikasi target. Target yang ingin diserang merupakan halaman *login* pada *website*, jadi menggunakan *post-form* yang didapat dari *website* tersebut.

Broken authentication attack menggunakan *tools hydra* dan berikut hasil yang didapat dari *bruteforce* yang dilakukan menggunakan *hydra* pada *kali linux*



Gambar 4. 9 Broken Authentication Attack with Hydra

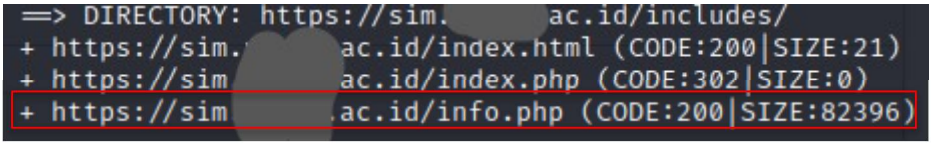
Dilihat dari hasil percobaan *bruteforce* diatas dapat disimpulkan percobaan *bruteforce* berhasil dilakukan dan terdapat celah untuk masuk pada *website* SIM xxx. Dan hasil yang diperoleh, terdapat banyak *username* dan *password* yang ditemukan dari hasil percobaan *bruteforce* menggunakan *hydra* dengan bantuan bebarapa informasi *post* http dari hasil *scanning burproxy* yang dilakukuan. Salah satu yang dapat digunakan untuk mencoba masuk ke sistem SIM xxx.

4.3.3 Sensitive Data Exposure

Pencarian *file* tertentu pada *website* SIM xxx dilakukan menggunakan *tools dirb* pada kali linux. Dan berikut hasil temuan *file* yang bisa diakses pada *website* SIM xxx.

Dirb https://Alamat_Website

Berikut hasil *command* diatas



Gambar 4. 10 ***Sensitive Data Scanning with Dirb***

Status *Code* 200 diatas menandakan berhasil/ditemukannya direktori pada URL yang dapat diakses pada Alamat_Website

Dan berikut hasil yang diperoleh dari percobaan URL yang didapat berdasarkan informasi yang ada pada gambar diatas

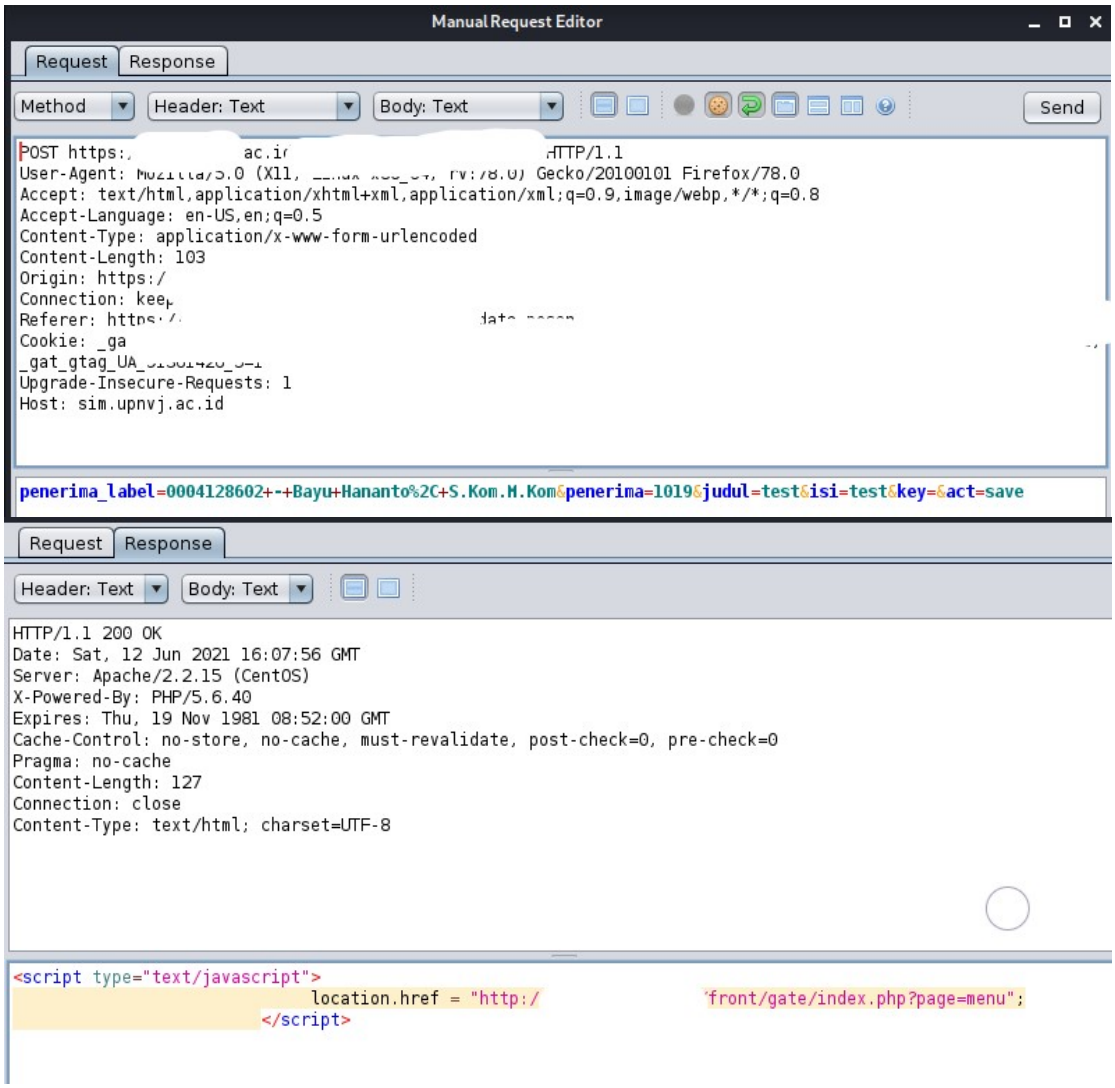
Gambar 4. 11 Sensitive Data Testing

Gambar diatas merupakan *file* data yang yang dapat diakses oleh *public* tanpa perlu masuk ke sistem *website* SIM xxx. Dapat dilihat dari gambar diatas bahwa Info.php berisi beberapa informasi yang sangat *detail* terhadap konfigurasi yang ada pada sistem *website* SIM xxx. Dari *Database* hingga informasi yang ada pada *server* seperti *open port configuration* dan data penting lainnya yang menguntungkan pihak *attacker*. Info.php perlu di*set* ulang agar *public* maupun attacker tidak dapat mengakses *file* yang termasuk dalam *sensitive* data tersebut.

4.3.4 XML External Entities (XXE)

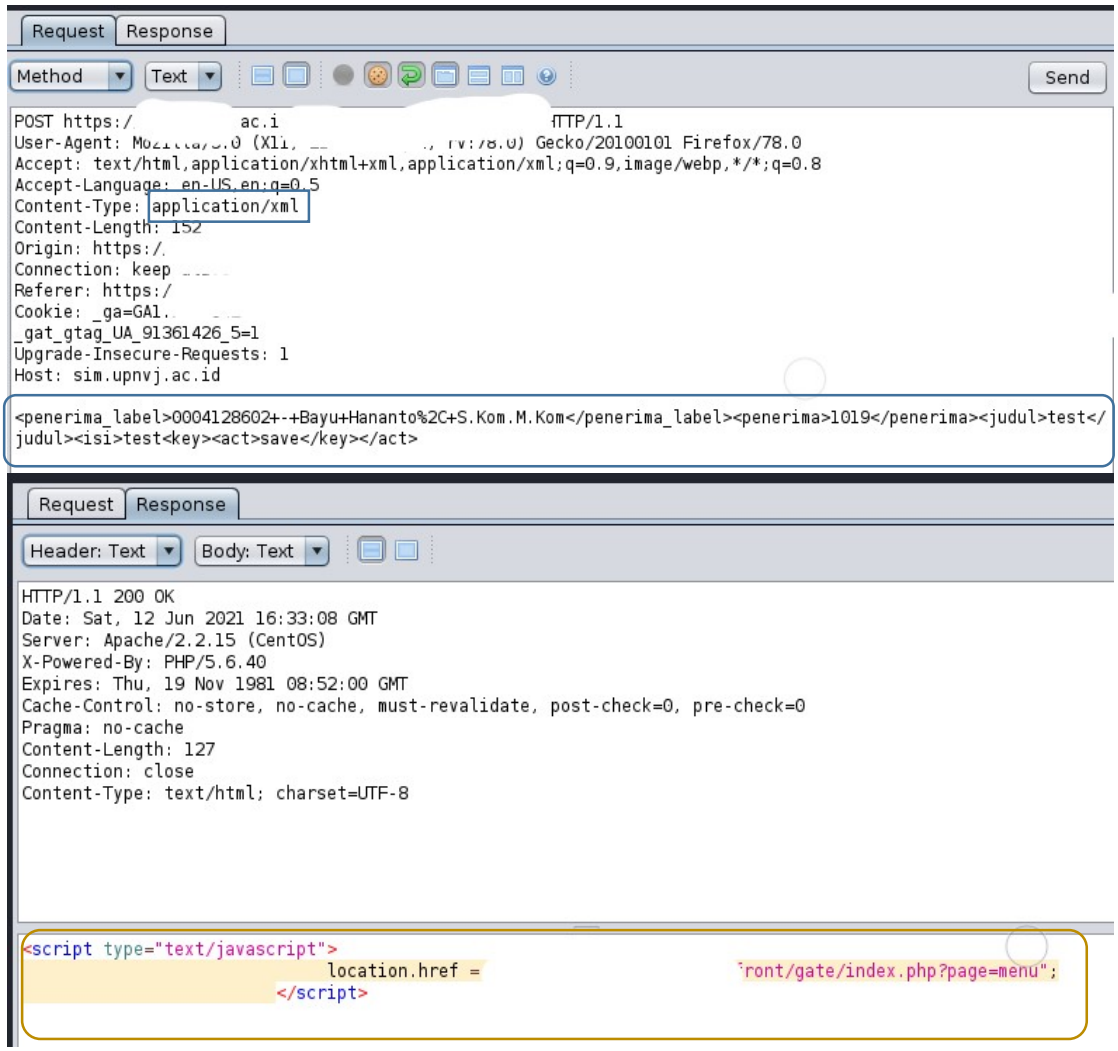
Berikut salah satu hasil *scanning zaproxy* pada saat proses mengirim pesan pada *section* kolom *text* yang ditemukan pada *website* SIM xxx dan mendapatkan *feedback* dari *database* dan data ditampilkan langsung setelah *submit button* tambah.

Berikut hasil *request* dan *respon* dari proses *entry* data yang baru ditambahkan.



Gambar 4. 12 XML File Scanning Result

Dapat dilihat dari hasil *request* tidak ditemukannya *tag* XML. Untuk dapat melakukan XXE, tag XML harus ada, dan apabila tidak ditemukan, dapat dilakukan dengan merubahnya menjadi file XML seperti yang ada dibawah ini



Gambar 4. 13 Percobaan Perubahan Content Type to XML

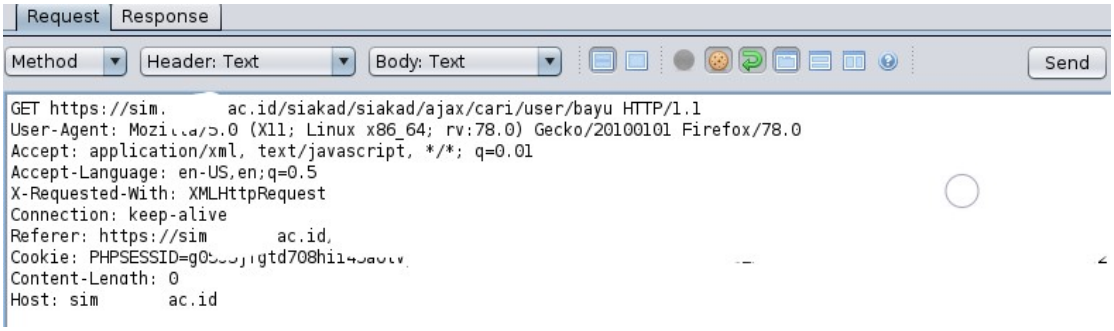
dan dapat dilihat dari *respon* diatas bahwa perubahan *text* menjadi XML berhasil dilihat dari hasil 200 OK diatas. namun tidak mendapatkan respon apapun dari *website* SIM xxx sehingga dapat dikatakan bahwa XXE attack tidak dapat dilakukan pada *website* SIM xxx.

4.3.5 Broken Access Control

Missing function level access control

Merupakan proses merubah hak akses secara ilegal tanpa diketahui oleh pihak *administrator website*.

Dikarenakan pada URL yang ditampilkan pada SIM xxx tidak adanya parameter ID yang muncul, percobaan akses ID lain secara manual tidak bisa dilakukan, untuk itu, perlu dilakukan *scanning* pencarian ID lain dengan mencoba *gennerate* ID *user* yang ada untuk mencari apabila terdapat ID *level user* yang lainnya menggunakan *Fuzzer* pada OWASP ZAP



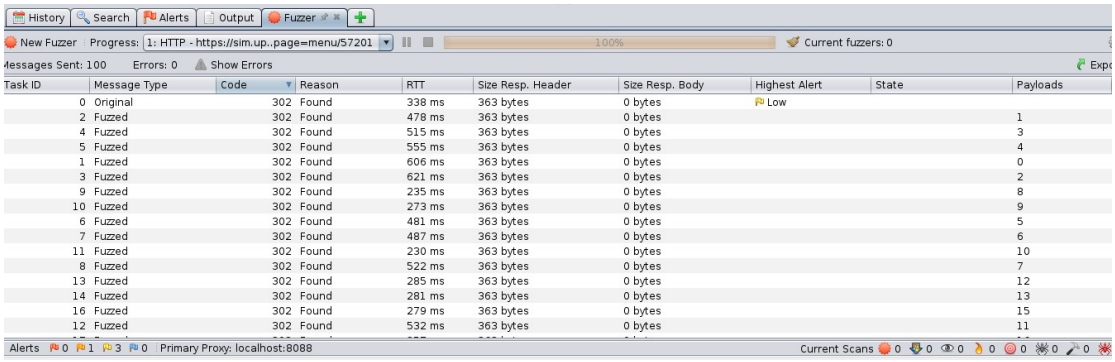
Gambar 4. 14 Scanning Access Control with OWASP ZAP

Tampilan diatas merupakan *scanning request* yg didapat dari *zaproxy* pada saat *login*

Gambar 4. 15 ID User Found Scanning Access Control with OWASP ZAP

Gambar diatas merupakan tampilan *respon* setelah *login*. Gambar diatas menampilkan beberapa informasi yang dapat digunakan untuk melakukan

missing function level access control atau bisa disebut perubahan akses *control* secara ilegal. Seperti diketahui gambar diatas terdapat ID *user* yang bisa digunakkan untuk melakukan *fuzzer.* Bertujuan untuk mendapatkan ID *user* lainnya yang mempunyai *role* yang berbeda apabila ada. Dan berikut hasil *fuzzer* setelah di*generate* untuk mendapatkan ID *role user* lainnya



Gambar 4. 16 Hasil Fuzzer ID Access Control with OWASP ZAP

Hasil gambar diatas menunjukkan bahwa tidak adanya ID lain yang ditemukan. Tidak dapat melakukan *missing function level access control* pada *website* SIM xxx.

Dari kedua jenis tipe *broken access control* yang ada pada *website* SIM xxx diatas dapat disimpulkan bahwa tidak adanya celah untuk melakukan *broken access control attack* pada website SIM xxx.

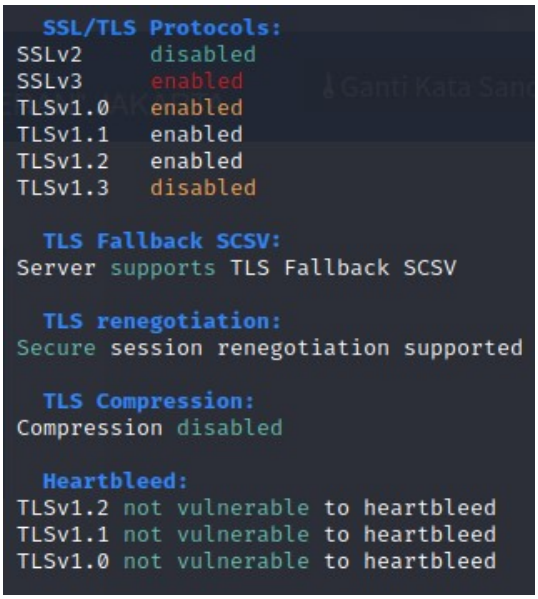
4.3.6 Security Misconfiguration

a. Configuration port ssl

Beberapa *developer* terkadang dapat melakukan kesalahan dalam mengkonfigurasi *port* ssl/https yang dapat menyebabkan kesalahan fatal dan dapat menimbulkan celah untuk melakukan berbagai macam serangan pada *port* ssl/https tersebut. Berikut *command* yang digunakan untuk melihat status *protocols* yang ada pada *port* ssl/https.

Sslscan http:Alamat_Website

Dan berikut hasil dari *Command* diatas



Gambar 4. 17 SSLScan

Dari gambar dilihat dapat dilihat bahwa SSLv3 statusnya *enabled*. SSL, dan penerusnya TLS, adalah protokol *kriptografi* yang dirancang untuk menyediakan keamanan komunikasi melalui Internet. Di ranah web, mereka menyediakan HTTPS, tetapi mereka juga digunakan untuk protokol aplikasi lain. SSLv1 tidak pernah dirilis secara publik, dan SSLv2 dengan cepat ditemukan tidak aman. SSLv3 telah dibuat, dan, bersama dengan TLSv1/1.1/1.2 yang lebih baru, saat ini masih digunakan untuk mengamankan lapisan *transport* Internet.

Seperti yang terjadi pada SSLv2, baru-baru ini *Google Engineering* menunjukkan bahwa SSLv3 rusak (dengan teknik eksploitasi yang dikenal sebagai *POODLE*) dan tidak boleh digunakan lagi. Ada tambalan, tetapi itu tidak mengurangi masalah sepenuhnya karena hanya akan berfungsi jika kedua sisi koneksi telah ditambal. SSLv3 hampir berusia 18 tahun, tetapi dukungan untuknya tetap tersebar luas. *Klien* dan *server* harus menonaktifkan SSLv3 sesegera mungkin.

b. Multiple environments

Library bug yang digunakan untuk *exploit open port* yang tersedia pada *server web* SIM xxx adalah *heartbleed*. *Heartbleed* merupakan *bug* keamanan di *library* kriptografi *Open*SSL, yang merupakan implementasi protokol *Transport Layer Security* (TLS) yang banyak digunakan. *Heartbleed* dapat dieksploitasi terlepas dari apakah *instance Open*SSL yang rentan berjalan sebagai *server* atau *klien* TLS. Ini dihasilkan dari validasi input yang tidak tepat (karena pemeriksaan batas yang hilang) dalam penerapan ekstensi *heartbleed* TLS.

Untuk dapat menggunakan *heartbleed bug* pada *open*ssl, harus mengaktifkan metasploit framework terlebih dahulu, lalu ketik *search openssl* dan masukkan *command openssl_heartbleed* seperti berikut

Use auxiliary/scanner/ssl/openssl_heartbleed

Set RHOSTS 103.xxx.xx.x

Set RPORT 443

Set VERBOSE TRUE

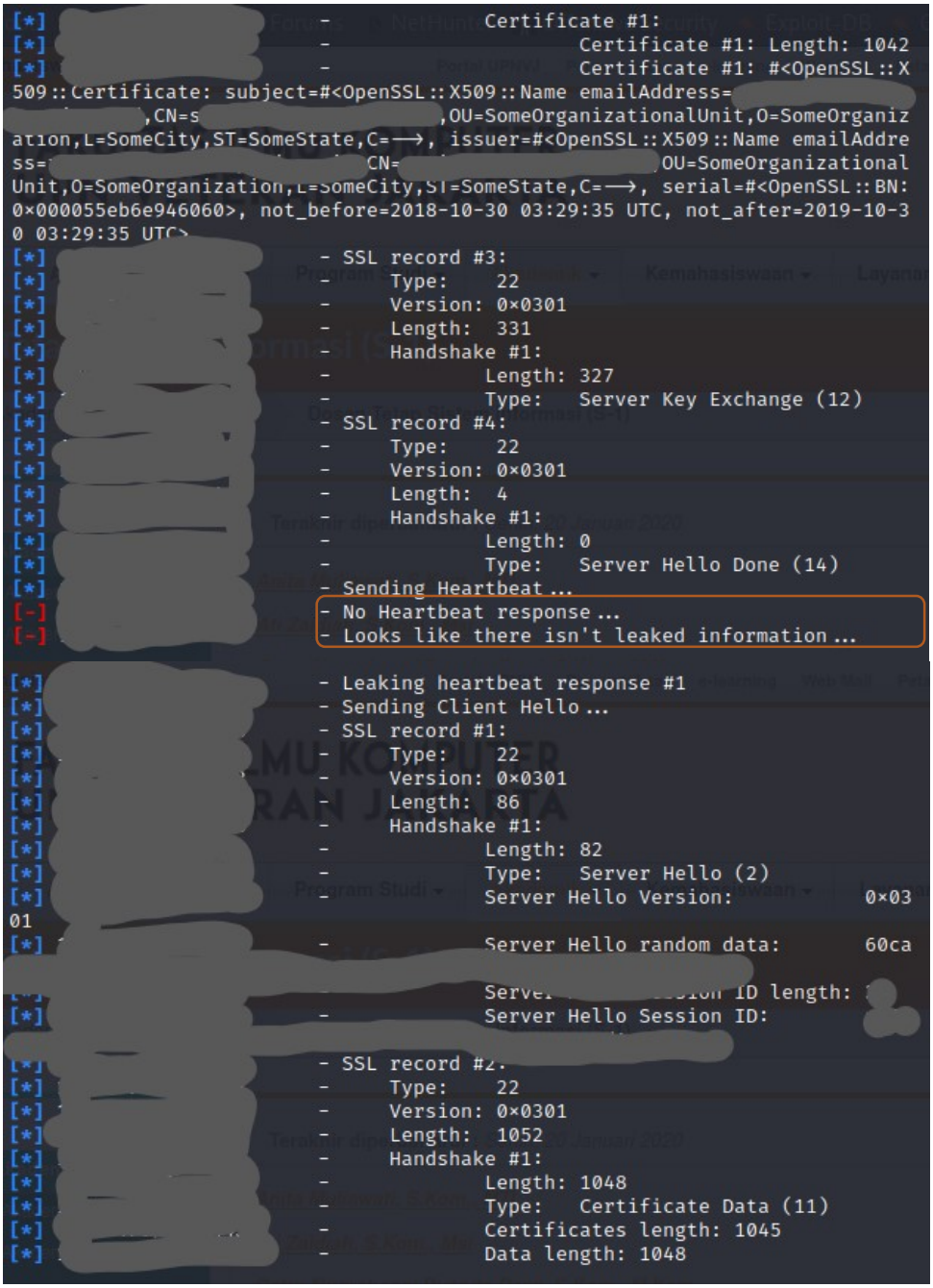
Exploit

Set R*HOSTS* : untuk *setting* target *hos*t yang dituju

set R*POR*T : berfungsi untuk *setting* target *Port* yang dituju dan

set *VERBOSE* : berfungsi untuk menampilkan hasil *detail exploit* yang didapat.

Berikut hasil dari percobaan *exploit* menggunakan komponen *library heartbleed* pada *metasploit framework* diatas.



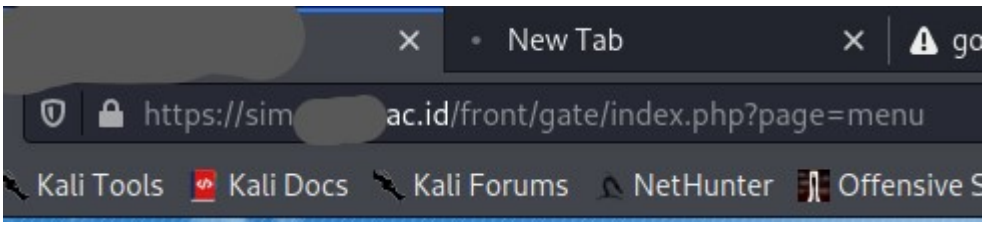
Gambar 4. 18 Hearbleed Result

Dari hasil *exploit* menggunakan *bug heartbleed* diatas mengatakan bahwa *"looks like there isn't leaked information"* yang berarti tidak adanya informasi rahasia yang dapat mengancam melalui *port* ssl/http. Dapat diambil kesimpulan bahwa tidak adanya kesalahan konfigurasi pada server SIM xxx.

c. Incorrect permission

Terkadang *developer* bisa lupa dalam memberikan akses terhadap file tertentu yang tidak sah. Seperti halnya percobaan yang dilakukan berikut. Mencoba mengakses *file* htpasswd yang merupakan *private bug bounty* program yang berisi kata sandi *hash* dari pengguna tertentu yang dapat di *crack.* Atau setidaknya dapat mengakses beberapa aset yang digunakan sebagai username dan kata sandi yang sama

berikut hasil yang didapat dari percobaan private *bug bounty* diatas



Gambar 4. 19 Httpasswd

Dapat dilihat dari *respon* diatas kembali lagi ke halaman sebelumnya yang berarti user tidak dapat mengakses *file* tertentu yang tidak sah pada *website* SIM xxx dan konfigurasi izin akses yang ada pada *website* SIM xxx sudah baik.

4.3.7 Cross-Site Scripting (XSS)

XSS cukup menyisipkan skrip atau HTML ke dalam halaman web. Secara umum, saat menguji kemungkinan serangan XSS, Anda perlu memverifikasi keaslian input Anda, dan penguji perlu mengetahui hal ini saat Anda memvalidasi *output* situs web Anda. Juga, jika tinjauan kode sedang dilakukan, penting untuk mengetahui bagaimana input cocok dengan *output*

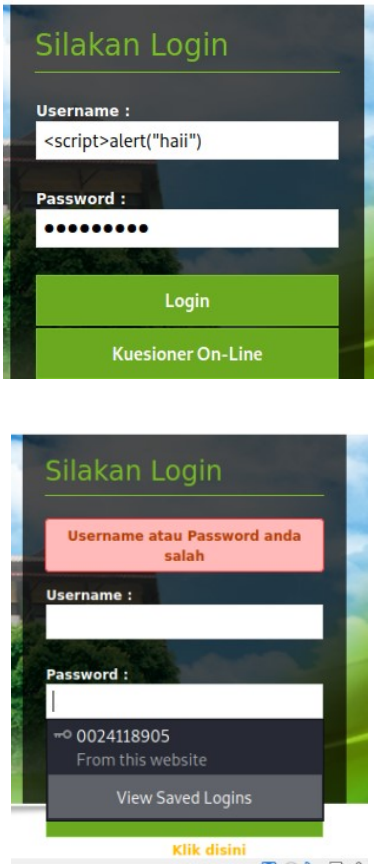
1. **Manual Input Script**

Berikut script yang coba dimasukkan pada *website* SIM xxx

```
<Script>alert(“haii”)

</script>
```

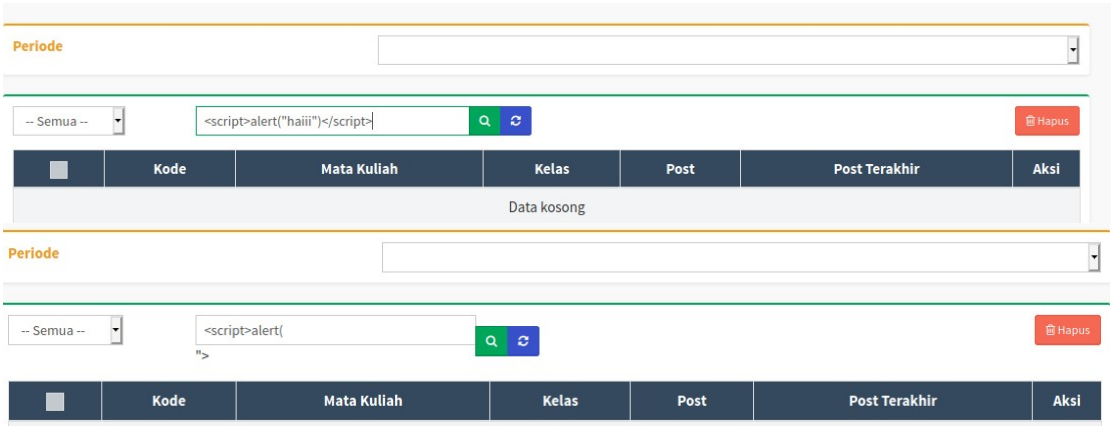
Berikut hasil percobaan XSS yang dilakukan pada halaman login *website* dan beberapa fitur *search* SIM xxx



Gambar 4. 20 XSS Login Page

Gambar diatas merupakan percobaan XSS *Attack* pada *login form website* SIM xxx. Dapat dilihat bahwa *script* yang dikirim tidak mendapatkan *respon* dan

tidak menampilkan apapun pada *login page* tersebut.

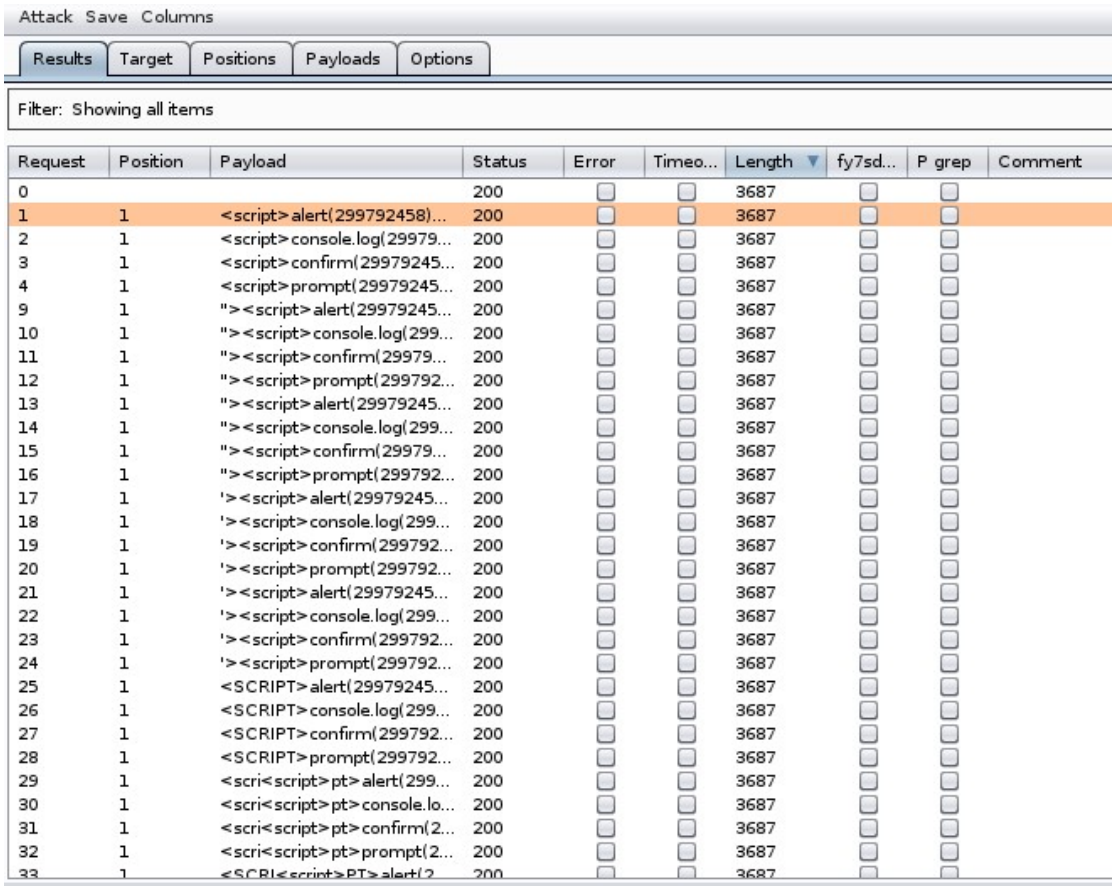


Gambar 4. 21 XSS Fitur Search

Gambar diatas merupakan hasil dari percobaan *script* XSS yang dilakukan pada beberapa *form search* yang ada pada *website* SIM xxx. Dan dapat dilihat bahwa tidak adanya *output* yang muncul sehingga dari percobaan XSS pada *website* SIM xxx tidak ada celah untuk melakukan XXS dan dapat dikatakan *website* SIM xxx aman dari XSS *attack*.

2. Input Script with Tools Burpsuite

Percobaan XSS *attack* lainnya dapat dilakukan dengan menggunakan *script* dan *key* yang tersedia pada *Burpsuite.* Alasan dilakukannya proses ini adalah *script* dan *key* yang tersedia pada burpsuite lebih bervariatif yang membuat kemungkinan celah ditemukan pada *website* lebih besar.

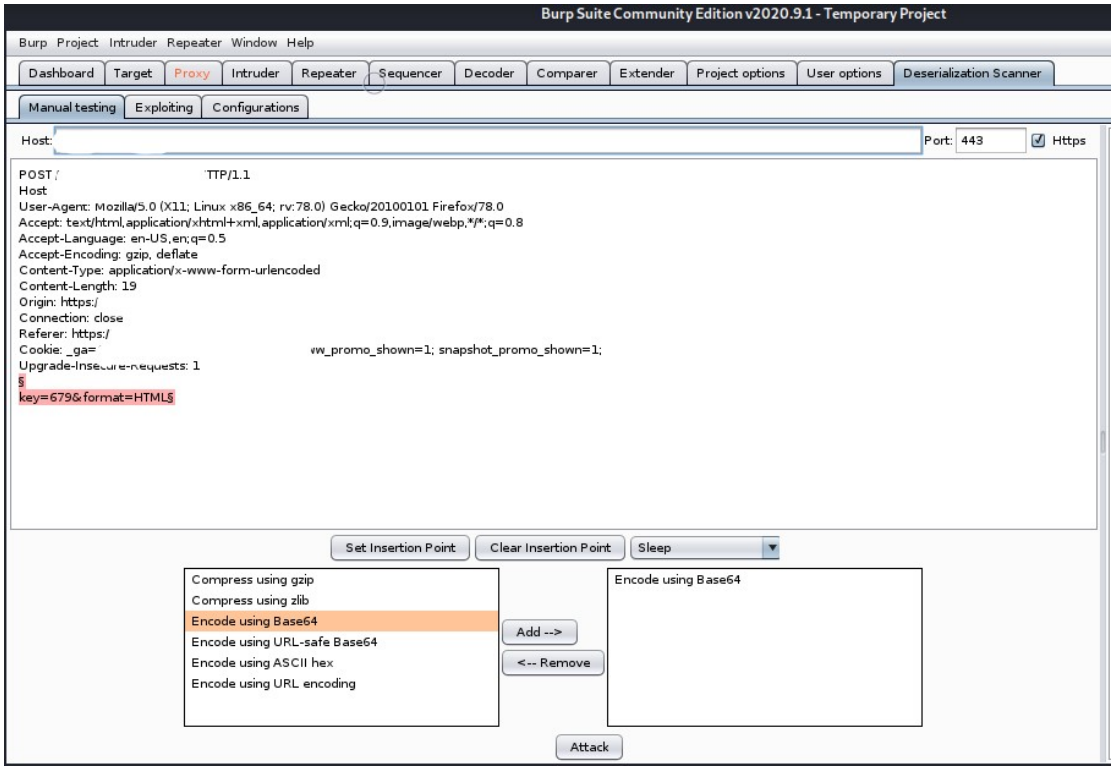


Gambar 4. 22 XSS with Tool Brupsuite

Dari gambar diatas dapat dilihat bahwa meskipun berhasil mengirimkan *request script* ke *website* SIM xxx, tetapi tidak adanya *respon* yang menampilkan feedback percobaan XSS *attack* tersebut. Yang berarti percobaan XXS *attack* tersebut meskipun telah menggunakan berbagai macam *script* dan *key* yang tersedia pada *tools burpsuite* tersebut masih gagal atau tidak berhasil. Artinya *website* SIM xxx aman dari segala bentuk serangan XSS.

4.3.8 Insecure Deserialization

Percobaan *insecure deserialization exploit* ini menggunakan *burpsuite extend java deserialization.jar*



Results:

- Apache Commons Collections 3 (Sleep): NOT vulnerable. Response time: 375 milliseconds
- Spring Alternate Payload (Sleep): NOT vulnerable. Response time: 258 milliseconds
- Apache Commons Collections 4 (Sleep): NOT vulnerable. Response time: 246 milliseconds
- Javassist/Weld (Sleep): NOT vulnerable. Response time: 272 milliseconds
- JSON (Sleep): NOT vulnerable. Response time: 268 milliseconds
- Apache Commons Collections 3 Alternate payload 2 (Sleep): NOT vulnerable. Response time: 221 milliseconds
- ROME (Sleep): NOT vulnerable. Response time: 240 milliseconds
- Mozilla Rhino (Sleep): NOT vulnerable. Response time: 391 milliseconds
- Apache Commons Collections 4 Alternate payload (Sleep): NOT vulnerable. Response time: 231 milliseconds
- Java 8 (up to Jdk8u20) (Sleep): NOT vulnerable. Response time: 225 milliseconds
- Java 6 and Java 7 (up to Jdk7u21) (Sleep): NOT vulnerable. Response time: 225 milliseconds
- Apache Commons Collections 3 Alternate payload 4 (Sleep): NOT vulnerable. Response time: 228 milliseconds
- Hibernate 5 (Sleep): NOT vulnerable. Response time: 244 milliseconds
- Commons BeanUtils (Sleep): NOT vulnerable. Response time: 224 milliseconds
- Apache Commons Collections 3 Alternate payload 3 (Sleep): NOT vulnerable. Response time: 230 milliseconds
- JBoss Interceptors (Sleep): NOT vulnerable. Response time: 241 milliseconds
- Spring (Sleep): NOT vulnerable. Response time: 242 milliseconds
- Vaadin (Sleep): NOT vulnerable. Response time: 241 milliseconds
- Mozilla Rhino Alternate payload (Sleep): NOT vulnerable. Response time: 336 milliseconds
- Apache Commons Collections 3 Alternate payload (Sleep): NOT vulnerable. Response time: 231 milliseconds

END

Gambar 4. 23 Insecure Deserialization Test with Burpsuite

Dapat dilihat dari hasil *testing* diatas bahwa tidak adanya *vulne*r/celah untuk dapat melakukan *insecure deserialization attack.* Yang berarti *website* SIM xxx aman dari kemungkinan ancaman *deserialization attack*

4.3.9 Using Components with Known Vulnerabilities

Komponen yang digunakan merupakan komponen *open port* yang ditemukan pada hasil *scanning* N*map* di proses *information gathering* sebelumnya. Percobaan *exploit* yang dilakukan merupakan *port* 5443 atau *postgre*SQL.

Sama halnya dengan *exploit* yang digunakan menggunakan *library heartbleed* diatas, exploit *postgre*SQL yang dilakukan berikut juga menggunakan *metasploit framework* menggunakan *command* dibawah ini

Search postgresql

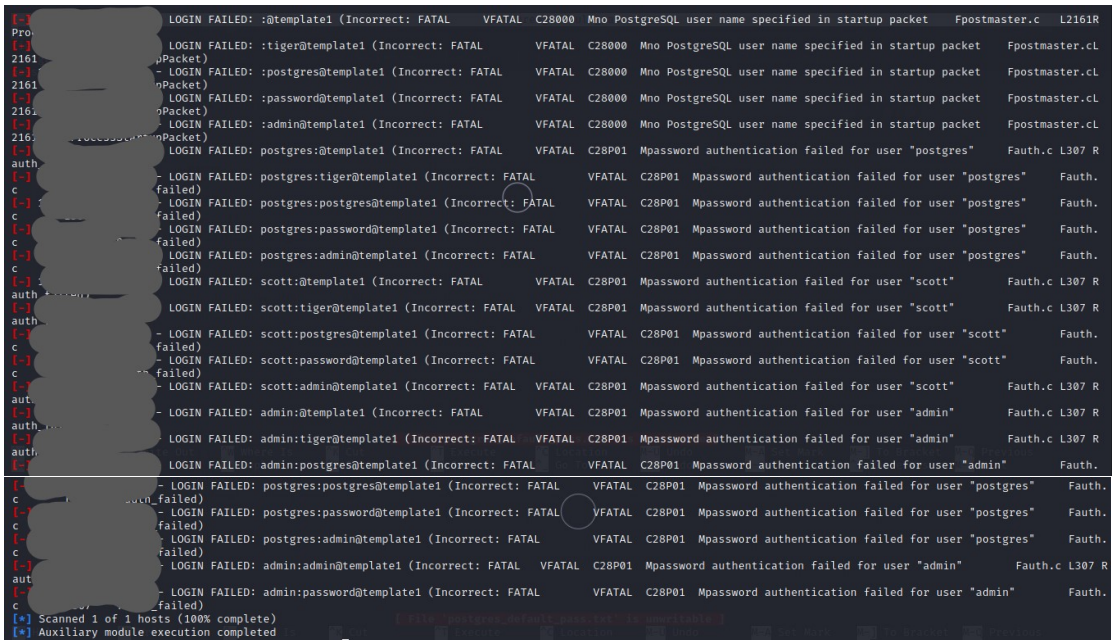
Use auxiliary/scanner/postgresql/postgresql_login

ket :

Search : mencari *package* yang tersedia

Use : *Library package* yang akan digunakan

Dan berikut hasil dari percobaan *bruteforce* yang dilakukan menggunakan *postgre*SQL library pada *metasploit framework*



Gambar 4. 24 Component Vulnerabilities with Postgresql Library at Metasploit Framework

Hasil dari percobaan *exploit postgreSQL* pada gambar diatas, tidak ditemukannya celah *vulne*r/kelemahan pada *open port* 5443 server *web* SIM xxx. Yang berarti *open port* 5443 sudah *secure* dari kemungkinan ancaman serangan.

4.3.10 Insufficient Logging & Monitoring

Untuk melakukan *tracking logging monitoring* perlu melakukan *exploit* untuk masuk ke server. Berikut command yang digunakan untuk *exploit* server pada *open port* 22 menggunakan *scanner openSSL_login* pada *metasploit framework.*

Use auxiliary/scanner/ssh/ssh_login
set USER_FILE /usr/share/wordlists/rockyou.txt
set PASS_FILE /usr/share/worlists/rockyou.txt
set RHOSTS 103.xxx.xx.x
set VERBOSE true
Exploit

Ket :

Use : *Library package* yang akan digunakan

Set USER_FILE : mencari *file username* yang akan digunakan

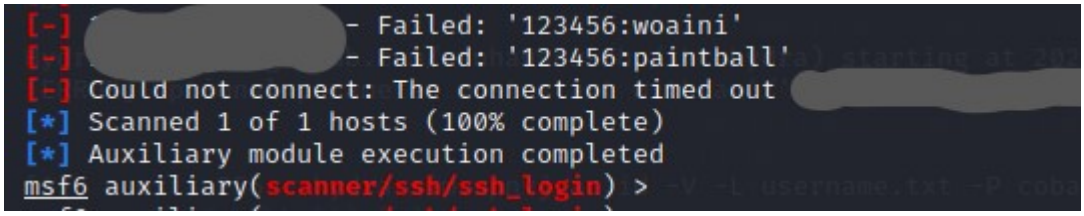
Set PASS_FILE : mencari *file password* yang akan digunakan

Set RHOSTS : setting tujuan *hosts* yang akan di *attack*

Set VERBOSE : untuk menampilkan *detail attack* yang dilakukan

Exploit : *Run* proses

Berikut hasil yang didapat dari percobaan *command* diatas

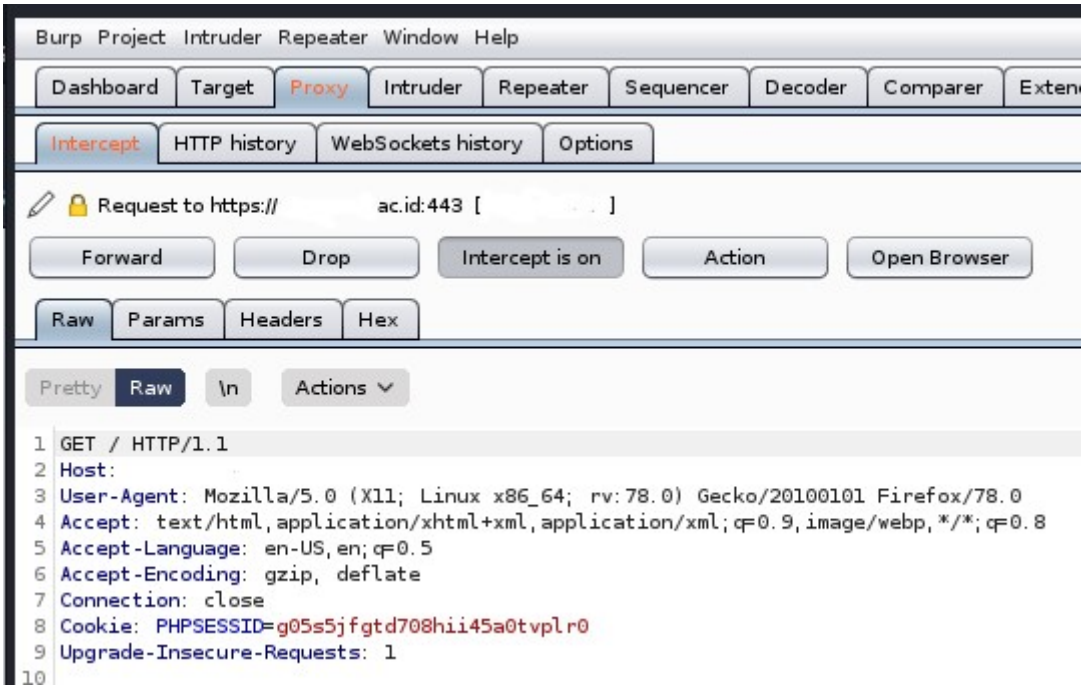


Gambar 4. 25 Logging & Monitoring Attack

Dapat dilihat dari gambar diatas bahwa hasil yang didapat adalah "*could not connect: the connection timed out (103.xxx.xx.x)*" Artinya percobaan untuk eksploit server tidak dapat dilakukan dan percobaan untuk melakukan *insufficient logging* and *monitoring* tidak berhasil.

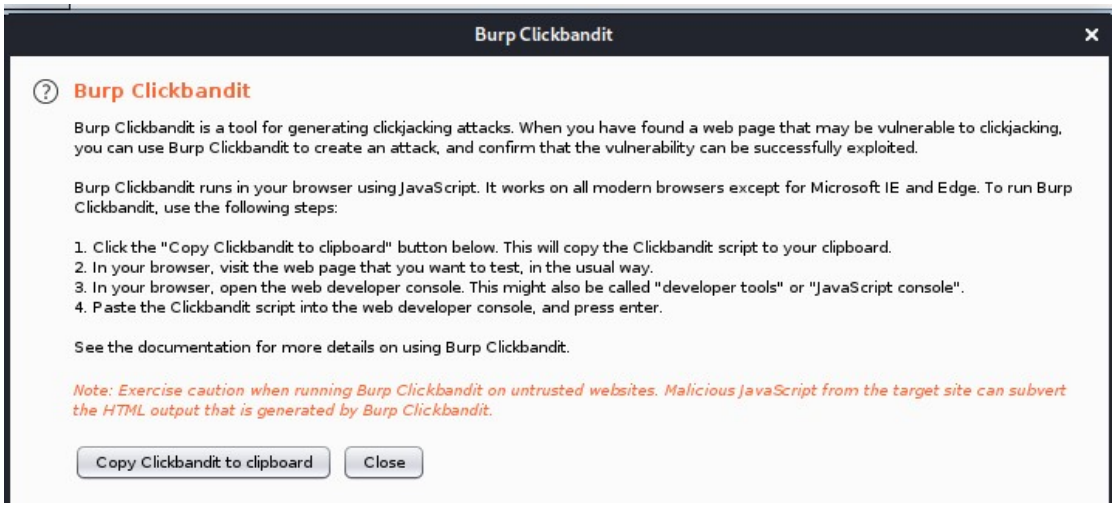
4.3.11 Clickjacking Testing hasil dari vullnerabillity scanning

Clickjacking script yang digunakan adalah script dari *tools Burpsuite* seperti gambar berikut



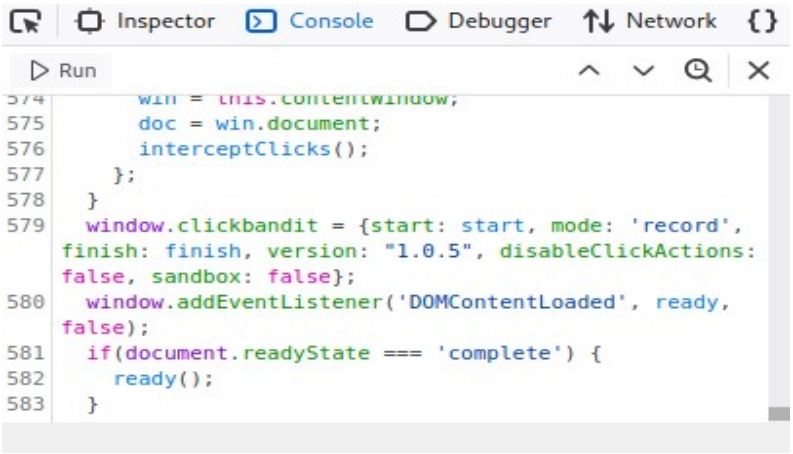
Gambar 4. 26 Hasil Scanning Burpsuite

Gambar diatas merupakan hasil *scanning* yang didapat pada halaman *login website* SIM xxx menggunakan *Tools Burpsuite.*



Gambar 4. 27 Tampilan Copy Burp Clickbandit

Dan gambar diatas merupakan *option copy burp clickbandit* yang akan digunakan untuk melakukan *clickjacking attack*



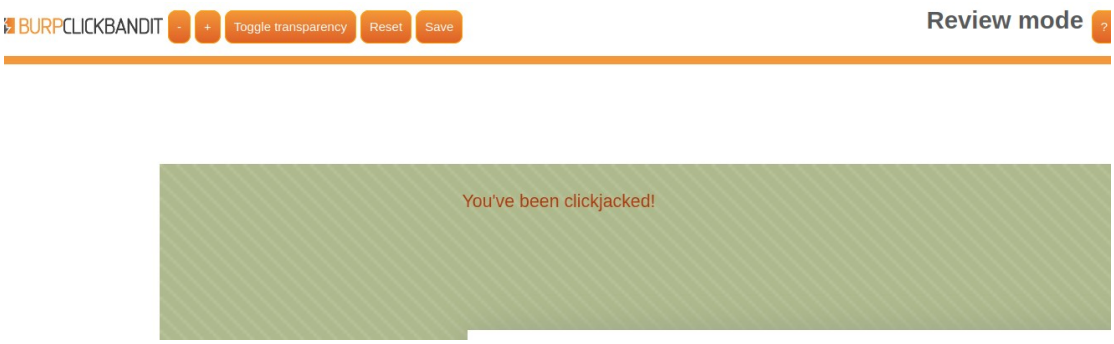
Gambar 4. 28 Script Burp Clickbandit

Setelah melakukan *copy script clickbandit* pada *burpsuite* sebelumnya, selanjutnya pergi ke halaman *login page* dan *button* yang ada pada *website* SIM xxx lalu *click* kanan dan *click console* untuk *paste script clickbandit* yang sudah di*copy* sebelumnya. Lalu *click Run* dan akan muncul tampilan seperti berikut.

Gambar 4. 29 Tampilan Pilihan Clickjacking Attack

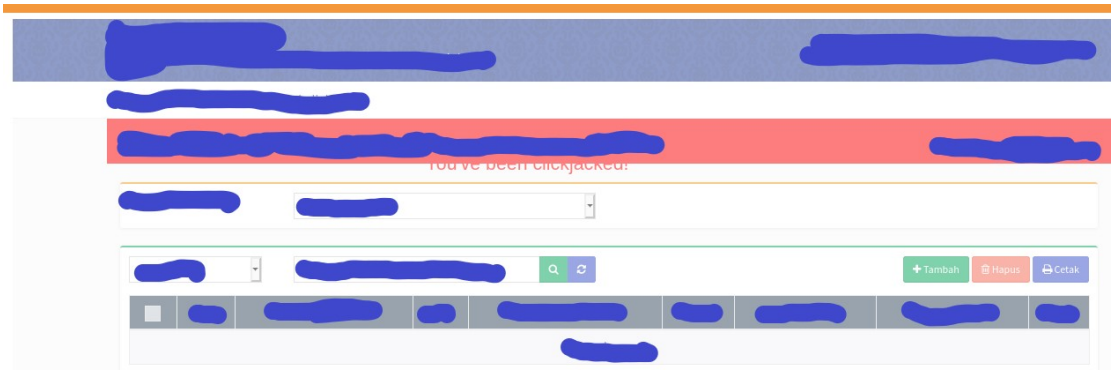
Selanjutnya *click* tombol yang ingin di *attack* yaitu tombol *submit* pada login. Lalu *click save*. Selanjutnya muncul tampilan seperti berikut

1. *login button*



Gambar 4. 30 Hasil Clickjacking1

2. *Button* pada setiap tombol pada *website* SIM xxx



Gambar 4. 31 Hasil Clickjacking2

Seperti hasil yang didapat dari *vullnerabillity scan* menggunakan OWASP ZAP sebelumya, terdapat celah untuk melakukan *clickjacking*. *Clickjacking* termasuk kerentanan yang cukup serius dan dikategorikan *medium warnin*g. Apabila *clickjacking* dilakukan oleh *hacker* bisa berdampak cukup serius, hacker dapat memanfaatkan *clickjacking* untuk mengambil *username* dan *password* dari *click button* yang sudah terkena *attack clickjacking*.

4.4 Report

Level Risk/Score	Metode	status	Hasil temuan	Saran
Critical 8/10	SQL Injection	Tidak Ditemukan	All tested parameters do not appear to be injectable	-
High 7/10	Broken Authentication	Ditemukan	1 of 1 target successfully completed, 41 valid password found hydra	Password dan username yang digunakan user terbilang cukup rentan dan mudah untuk ditebak. Disarankan untuk para user dapat mengubah username maupun password yang lebih kuat dan tidak mudah ditebak untuk menghindari brute force attack yang dapat menimbulkan broken authentication pada website SIM xxx
High 7/10	Sensitive Data Exposure	Ditemukan	DIRECTORY: https://Alamat_Website.AC.ID/includes/ + https://Alamat_Website.AC.ID/info.php (code:200|SIZE:82396)	Info.php berisi berbagai macam informasi yang sangat cukup serius dan sangat berguna untuk para hacker. Disarankan agar info.php dapat Hide untuk public
High 7/10	XML External Entities (XXE)	Tidak ditemukan	No response	-
Medium 6/10	Broken access control:	Tidak ditemukan	1. No response 2. Code 302	-
Medium 6/10	Security Misconfiguration 1. SSL setting info 2. Scan openssl heartbleed	1. Ditemukan 2. Tidak ditemukan	1. SSLV3 enabled 2. No heartbeat response Looks like it isn't leaked information	1. SSLV3 harus disabled 2. -

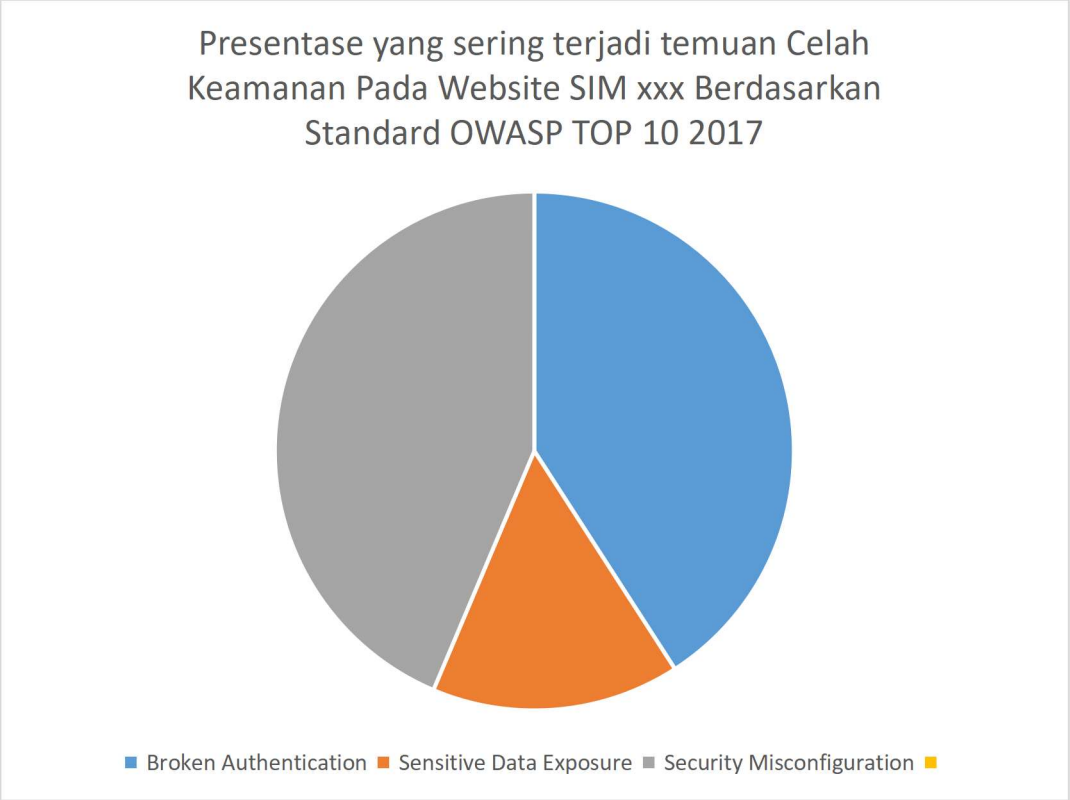
Medium 6/10	Cross-site-scripting (XSS)	Tidak ditemukan	Length 3867 Length 496	-
Medium 5/10	Insecure Deserialization	Tidak ditemukan	Not vulnerable	-
Low 4,7/10	Using components with vulnerabilities	Tidak ditemukan	103.xxx.xx.x:5432 - LOGIN FAILED	-
Low 4/10	Insuficient logging and monitoring	Tidak ditemukan	could not connect: the host (103.xxx.xx.x:22) was unreachable	-
Medium	Clickjacking from vulnerabillity scanning	Ditemukan	You've been clickjacked	pada sisi client dapat ditambahkan addons no script dan sementara pada sisi server bisa menggunakan *Frame Killer, X frame options*.

Tabel 4. 2 ***Reporting***

Ancaman-ancaman pada *website* yang terjadi pada tahun 2017 sudah didata oleh OWASP (*Open Web Application Security Project*) dan sudah tercatat pada OWASP Top 10 Security – 2017. Pada OWASP Top 10 Security, terdapat beberapa ancaman dan tingkat resiko dari dampak serangan yang telah diklasifikasikan oleh OWASP. Tingkat ancaman yang diberi nilai sudah dihitung dengan kalkulator khusus dari NIST (*National Institute of Standards and Technology*) yang disebut CVSS (*Common Vunlnerability Scoring System*) dengan rentang *score* 0.0 sampai 10.0. Ancaman tersebut telah digambarkan pada Tabel 4.2

No	Ancaman	CVSS Score	Persentase yang Sering Terjadi	Status
1	Injection	8.0	35%	Tidak Ditemukan
2	Broken Authentication	7.0	74%	Ditemukan
3	Sensitive Data Exposure	7.0	28%	Ditemukan
4	XML External Entities (XXE)	7.0	2%	Tidak Ditemukan
5	Broken Access Control	6.0	53%	Tidak ditemukan
6	Security Misconfiguration	6.0	79%	Ditemukan
7	Cross-site Scripting (XSS)	6.0	77%	Tidak Ditemukan
8	Insecure Deserialization	5.0	2%	Tidak Ditemukan
9	Using Component with Known Vulnerabilities	4.7	28%	Tidak Ditemukan
10	Insufficient Logging & Monitoring	4.0	2%	Tidak Ditemukan

Tabel 4. 3 **Klasifikasi Ancaman pada tahun 2017**
Sumber: https://www.owasp.org/images/0/0a/OWASP_Top_10_2017_GM_%28en%29.pdf



Grafik Gambar 4. 32 Hasil Temuan Berdasarkan Metode Owasp