Situational Awareness with IP’s

If you’ve been given an IP, and you need to do some threat intel on it, you can get a pretty good feel for the type of host it is, where it is, and what it does.
IPInfo

Usually if I get given an IP, I’ll do a lookup with ipinfo.

curl ipinfo.io/54.90.107.240


```
{
  "ip": "54.90.107.240",
  "hostname": "ec2-54-90-107-240.compute-1.amazonaws.com",
  "city": "Virginia Beach",
  "region": "Virginia",
  "country": "US",
  "loc": "36.8512,-76.1692",
  "org": "AS14618 Amazon.com, Inc.",
  "postal": "23465",
  "readme": "https://ipinfo.io/missingauth"
}
```


IPInfo will return JSON with details all about the host, the great thing about this is that you can easily script it by piping into jq.

I tend to abuse bash for loops for this kind of thing, say you have a text file full of IP’s:

```
for ip in $(cat ips.txt); do echo -n "$ip: "; curl -s ipinfo.io/$ip | jq .org; done
```

```
54.90.107.240: "AS14618 Amazon.com, Inc."
54.90.107.120: "AS14618 Amazon.com, Inc."
54.90.107.241: "AS14618 Amazon.com, Inc."
54.90.107.242: "AS14618 Amazon.com, Inc."
54.90.107.243: "AS14618 Amazon.com, Inc."
```

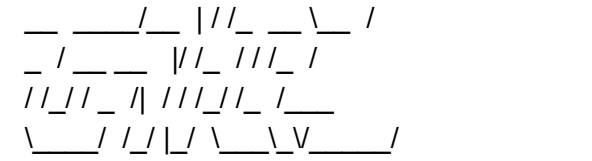
GreyNoise.io 16

You can do the same with Greynoise, if you don’t know already, Greynoise.io 17 is a badass service that hosts thousands of listeners all over the internet silently listening. When devices scan the internet for different ports, services, HTTP requests and the like, Greynoise takes note and indexes them.

The idea behind Greynoise is to ingest all the noise on the internet, so that you can filter it out.

If you have an API key, you can use the greynoise client from https://github.com/GreyNoise-Intelligence/GNQL 13.

greynoise 54.90.107.240



```
┌──────────────────────────────────────┐
│      result 1 of 1       │
└──────────────────────────────────────┘
```

```
        OVERVIEW:
----------------------------
IP: 54.90.107.240
Classification: unknown
First seen: 2018-10-19
Last seen: 2018-10-19
Actor: unknown
Tags: ['Web Crawler', 'HTTP Alt Scanner']

        METADATA:
----------------------------
Location: Ashburn, United States (US)
Organization: Amazon Technologies Inc.
rDNS: ec2-54-90-107-240.compute-1.amazonaws.com
ASN: AS14618
OS: unknown
Category: hosting

        RAW DATA:
----------------------------
Port/Proto: 8443/TCP

[Paths]
/
```

And of course, you can loop this around all day with bash for loops and the -o json option. Shodan

You are probably aware of Shodan, I had to mention this for those who still don't know, as it's such a valuable tool.

Shodan scans all the hosts on the internet, all the time. This means you can preform a lookup of a host and see what they have.

shodan host 216.58.210.206

```
216.58.210.206
Hostnames:              mrs04s09-in-f206.1e100.net;lhr48s11-in-f14.1e100.net
City:                   Mountain View
Country:                United States
Organization:           Google
Updated:                2019-08-17T19:28:38.408716
Number of open ports:   2

Ports:
     80/tcp
    443/tcp
          |-- SSL Versions: TLSv1, TLSv1.1, TLSv1.2, TLSv1.3
```

Email Recon

A quick little trick I picked up is preforming recon on email addresses extremely quickly using EmailRep.

curl emailrep.io/john.smith@gmail.com


```
{
  "email": "john.smith@gmail.com",
  "reputation": "high",
  "suspicious": false,
  "references": 91,
  "details": {
    "blacklisted": false,
    "malicious_activity": false,
    "malicious_activity_recent": false,
    "credentials_leaked": true,
    "credentials_leaked_recent": false,
    "data_breach": true,
    "last_seen": "07/27/2019",
    "domain_exists": true,
    "domain_reputation": "n/a",
```

```
    "new_domain": false,
    "days_since_domain_creation": 8773,
    "suspicious_tld": false,
    "spam": false,
    "free_provider": true,
    "disposable": false,
    "deliverable": true,
    "accept_all": false,
    "valid_mx": true,
    "spoofable": true,
    "spf_strict": true,
    "dmarc_enforced": false,
    "profiles": [
      "lastfm",
      "pinterest",
      "foursquare",
      "aboutme",
      "spotify",
      "twitter",
      "vimeo"
    ]
  }
}
```


SSH Tunelling

If you've ever exposed a CobaltStrike team server port externally, and told people about it, you'll get a lot of hate (source: 1337 hacker slacks). What's the solution? SSH Tunelling.

If you have SSH access to a host, you can tunnel ports (map remote ports to local ones), dynamically create SOCKS proxies, and a lot of really cool things.
Mapping remote port to local port

ssh -L localport:127.0.0.1:remoteport user@host

A good way to think about the syntax of SSH tunnels is to split it into two parts (when I saw this it blew my mind.)

ssh -L 127.0.0.1:8080:127.0.0.1:80 user@host

This will open local port 8080, mapped to port 80 on the remote server. Luckily for us, SSH is kind and let's us infer the first host as local.
Opening a SOCKS proxy that routes from your server

SSH -D 8080 user@host

This will open a socks proxy on local port 8080, you can modify your proxychains.conf to accept this port, and then use proxychains before every command to route traffic through that host.
Vagrant

This is a cool little trick I learned, and it has really made me productive and has generally made things easier.

Like Docker, vagrant can spin up instances of operating systems and drop you into an interactive shell.

My favourite is using Ubuntu:

vagrant init hashicorp/precise32
vagrant up
vagrant ssh
cd /vagrant/

You'll be dropped into an Ubuntu Precise shell!
Conclusion

In conclusion, a lot of cool little tricks can really make your life easier as a pentester. Small one liners, a reference article like this, and you may actually look like you know what you're doing.

I hope you enjoyed this braindump :D, have an amazing day 0x00ers!

- Pry0cc