securit.ie

When did you last check your stack?

Leave a Comment

Posted on February 4, 2019

The Lazy Hacker

by **Ciaranmcnally**

It has been over a year since I last blogged, as reflected in this post's name. My working situation has improved a lot over the past year which has led to me neglecting the previously enjoyed exercise of bug bounty hunting and blogging. I do very much intend on getting back into these activities at some stage but I'm kept busy for now. I work roughly 7 months of the year on contracts. Mainly taking the form of internal/external penetration tests and vulnerability assessments. I do still sometimes take on incident response jobs, many of which are subcontracted to me against my will 😉 I enjoy the rush of an active attack and witnessing the TTPs (techniques, tactics & procedures) of seasoned criminals but like you I can't and won't talk about any of that here because I'll trigger my thousand yard stare.

Search …

GO

RECENT POSTS

The Lazy Hacker

OSCP Certification

Freelance Security Consulting!

Zero Days CTF – Dublin 2016

Daggercon 2015

ARCHIVES

February 2019

September 2017

June 2016

April 2016

October 2015

So it's a given that I have had 5 months of the year free to develop and encourage my own laziness via automation. That's what this series of blog posts will focus on, hopefully in enough depth to make it a worthwhile read and potentially convince someone I'm worth contracting. I'd like to show the world some of the things I've been working on recently and share my mentality, toolkit and methodologies. I can guarantee a lot of this stuff is being done by others out there, as I like to read a lot and I didn't pick up these ideas or skills from kicking rocks. It takes a lot of active learning and researching to stay on top of the latest security trends and techniques.

Many folks are indeed better than I at being cutting-edge, thanks to their active information circles, hard work and dedication to self-improvement. I still run tools like Nessus against infrastructure and am very fond of burp-suite, these tools compliment and can be used in conjunction with any of my original tools. They capture useful information very quickly, especially on internal environments. Nessus authenticated scans are very handy to have and I won't stop using such tools any time soon. I want to thank everyone in this wonderful security community, who takes the time to openly share their trade-craft, hard lessons learned and research.

All I can share is what has worked for me consistently and maybe inspire some others with ideas or new avenues to explore. Since a lot of my stuff is self-developed, I may also eventually release more tools but the bulk of my toolkit probably won't be shared, it's my bread and butter and should help me find things others miss. Most

September 2015

CATEGORIES

Uncategorized

META

Log in

Entries RSS

Comments RSS

WordPress.org

things can also be easily accomplished with other tools already in the public domain. I develop for myself in python, it means I know my own tools inside out and can prioritize output in a way that the data becomes immediately actionable for me.

A Simplified Methodology

In order to avoid writing a book, I'll try simplify and explain a lot of ideas as generally as I can. The following is a simplified penetration testing methodology. This should be effective against any online business, network or company, of any scale.

1. Reconnaissance, OSINT and Information Gathering
2. Network scanning, Service and application identification, vulnerability identification
3. Exploitation & Escalation

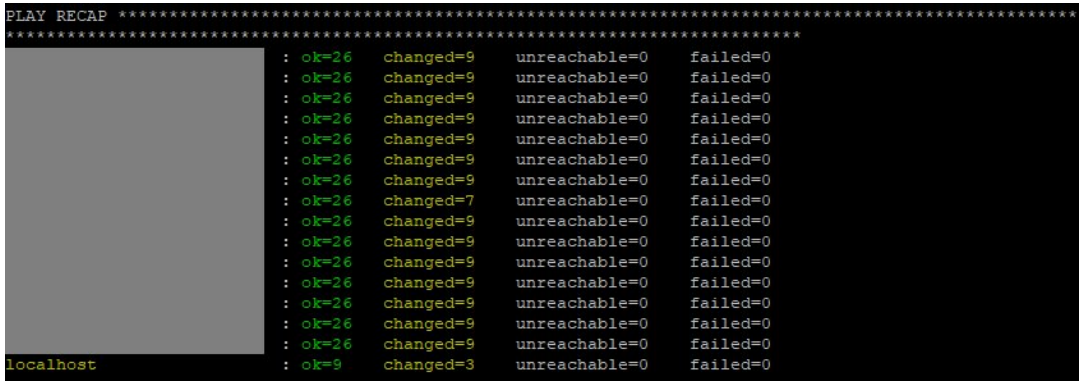
I set out with a plan of automating as much as I could and especially where current tools or techniques were lacking, so at a minimal I was at least speeding up what I do manually. This should allow me to have more time for manual examination and fuzzing which increases my value by increasing the likelihood that more serious issues can be exposed in the time available.

Getting Started

- who is the target? company? acquisitions? owners? developers?
- where/how do they hire? where are they based? what tech do they use?
- what's their major product? what services do they provide?
- 3rd parties they interact with? How do they accomplish their goals?
- where are their networks? how are they managed? by who?
- what issues have impacted them? what attacks have they seen before?
- Company presentations? product demos? any URLs/tech used in them?
- what do they want or need to avoid?

With an understanding and profile of the target, it's time to start enumerating public network information resources. This is the first technical step of my workflow and will be the focus of this post.

Some key concepts I relish throughout, are that all data and technical information captured is useful, can be expanded on, should be stored and is continuously fed in a feedback loop manner from process to process. It is also essential to keep track of when, where and what you are testing. This can help if a client wants to dig into anything further. Keep a log of your own activities and enable logs for tools, it'll save you heartache in the long run and allow you to retrace steps in future.

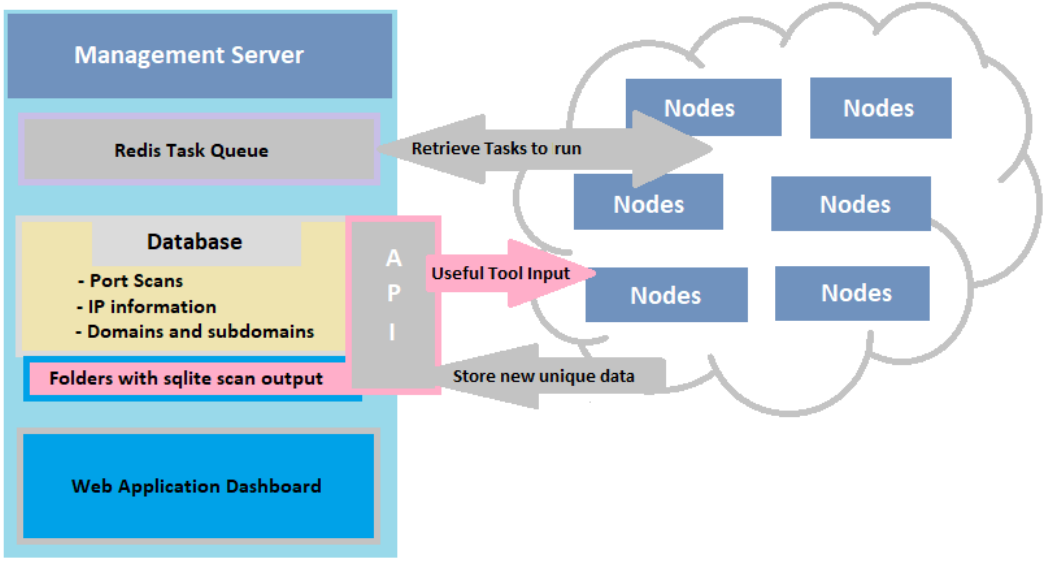


Ansible playbooks are bae

The final concept, is that I want everything I build to be easy to tear down and rebuild. I accomplish this through a series of ansible playbooks. This allows me to rapidly scale my tools out when needed and reproduce my architecture quickly on a fresh set of hosts. Ansible is perfect for me as an alternative to bash scripts and I can manage all of my infrastructure on the command line of a single administrative host.

Capturing the Network

Before digging into the specifics I'd like to share the architecture behind one of my primary tools used for reconnaissance. This is a tool I developed called "*recron*" and is an *automated continuous recon framework*. It is composed of four main parts on the management side, a database – a redis task queue, an API and a web app/dashboard. And then a single orchestration client on any number of worker nodes to run my CLI tools. It scales quite well, I can leave it running over a week with no hiccups and I have no idea what to do with this beautiful beast yet.



recron: network reconnaissance architecture

The basic principle here is that no information is lost once it's collected. Through cronjobs and continuous enumeration, it can be deployed against a target for the duration of a penetration test or as an ongoing monitoring service to manage assets from an offensive perspective in an ongoing way. The tool allows data or information to be manually added and updated, with changes in infrastructure updating automatically on a continuous basis and being stored or tracked over time. It's possible to implement automatic alerting for these changes too. However I mostly just want a clear snapshot of the external network infrastructure as it is.

The continuous monitoring and alerting feature set I use and strive for are the following:

- Subdomain discovery, brute forcing and enumeration
- Network edge-expansion, identifying unexplored IP ranges and domains
- Port scans, banner capturing and enumeration
- Web application enumeration via brute forcing and discovery scans

Service and Application Identification

As an example a domain is added to the database via a CLI tool, a continuous process adds this new domain to the task queue. This information is picked up by the multiple worker nodes and the domain is fed into multiple domain discovery tools like *subfinder* (**https://github.com/subfinder/subfinder**) and IP information is retrieved from various current and historical API's and data-sets like those at *scans.io* (**https://scans.io/**). The unique output of these tools is then added back into the database via the API and the process is repeated.

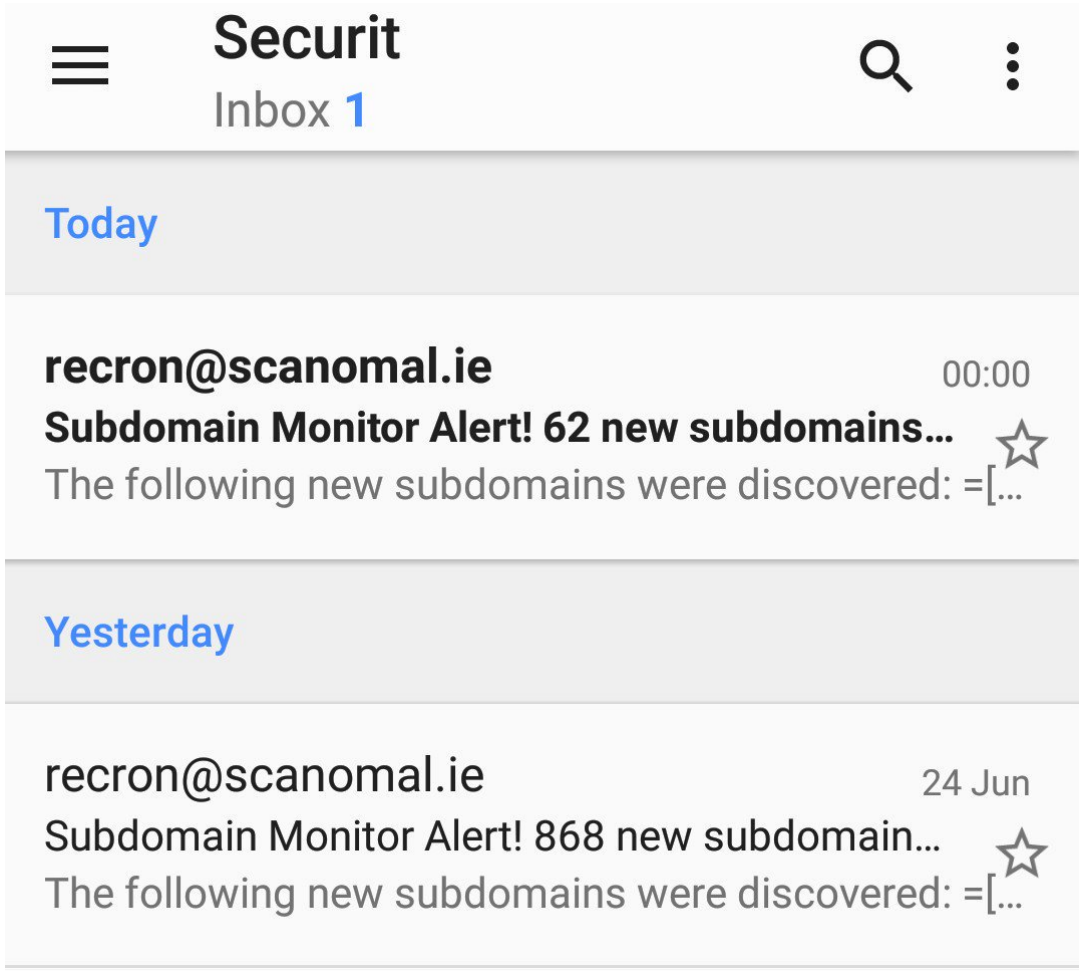
This continuous feedback loop has endless possibilities and allows for the smarter discovery of otherwise hard to find assets. For example one of the tasks uses an excellent tool called *altDNS* (**https://github.com/infosec-au/altdns**) which allows the

generation of alterations and permutations of subdomains. We can feed this tool a single domain/subdomain, a dictionary, all the subdomains discovered thus far for the target via our API and any other common subdomain brute force lists deemed useful. A brute force of this generated list with *massdns* doesn't take longer than 10 minutes (**https://github.com/blechschmidt/massdns**). This ensures the rapid discovery and enumeration of new assets, subdomains or potential virtual hosts, when you increase the number of client nodes.

Other things we can do is brute the subdomains of subdomains, whereby ***example.blah.test.domain.com*** can be brute forced at the following locations:

- *.example.blah.test.domain.com
- *.blah.test.domain.com
- *.test.domain.com
- *.domain.com

For large companies it should be obvious how this increased attack surface is especially useful with regards to bug bountying. The results of this technique speak for itself.



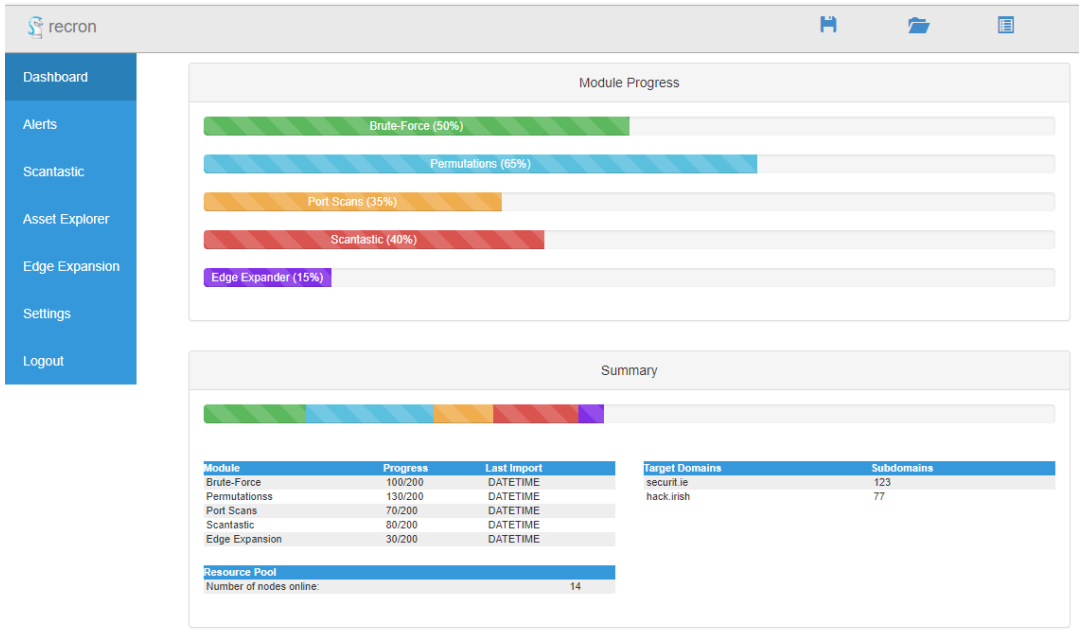
Asset discovery and subdomain recon

The screenshot above is new subdomains discovered on top of what was found with traditional tooling. Another module of *recron* involves network edge expansion. This for example takes all the IP addresses currently in the database and identifies the network ranges that have multiple subdomains already identified within a /24 block, it

then does a reverse lookup of all IPs in that network and checks SSL certificates for any additional domains. Flagging other newly discovered domains as potentially owned by the same target organisation, this is less useful with modern cloud based services but is good for company owned network blocks and can reveal new areas or domains to explore or allow new data to be added into the discovery process.

Reusing all the information stored, presents me with other interesting opportunities. A more recent example I've explored, is taking a list of IP addresses that were resolved from the target company domains. A task then runs a *vhost brute force* against these IPs using a list of all subdomains discovered thus far. This can reveal old/alternative web apps that were never removed from the company owned IPs in question and open up more additional attack surface. This has on multiple occasions revealed web applications for subdomains that didn't have current DNS entries.

Another extremely useful module is one to run network scans against the identified IPs, this is also a continuous or daily process and takes an IP from the database, adding it to the task queue. I usually run *masscan* (**https://github.com/robertdavidgraham/masscan**) and then Nmap against a target with the ***-A*** flag so it also runs additional information gathering scripts. The results are then imported via the API and added to a host allowing for asset exploration via the management dashboard.



An older screenshot of the dashboard with test data

The idea is to make useful information accessible as quickly as possible and available in order to make it actionable. Another component of this *automated recon framework* is a tool I just recently "finished". I based it on a few older tools of mine that dumped directory brute force scans into elasticsearch (called *scantastic*), I wanted to minimize the overhead of my infrastructure however and SQLite works perfectly fine in this case.

https://github.com/maK-/scanomaly

I'm releasing the latest tool that was part of the framework and it's called ***scanomaly***. The intent of this was to combine multiple tools into one flexible tool and store all the data for single targets in a databases. Using the module based system it also allowed me to rapidly script prototypes for new attacks and run them across a network.

This allows me to compare databases and generate alerts based on new content that has appeared on web servers since the last scan and also carry out general fuzzing. There isn't a public user interface for the output of this tool yet but I may just copy some of the stuff from the management flask app in the screenshot above. I want the visual elements to go along with the anomaly detection tools, based on the

response information stored in the responses table. I'll likely do another blog post on how to use this tool with examples once I've the user side of it finished.

This post is getting too long so I'm going to call it before talking about exploitation and escalation …

It's good to be back among the blogging world

Category: **Uncategorized** Tags: **automated**, **automation**, **bug bounty**, **hack**, **hacker**, **hacking**, **hacks**, **incident response**, **information gathering**, **infosec**, **lazy**, **penetration testing**, **PenetrationTest**, **reconnaissance**, **recron**, **scanomaly**, **securit**, **securit consulting**, **security**

← **OSCP Certification**