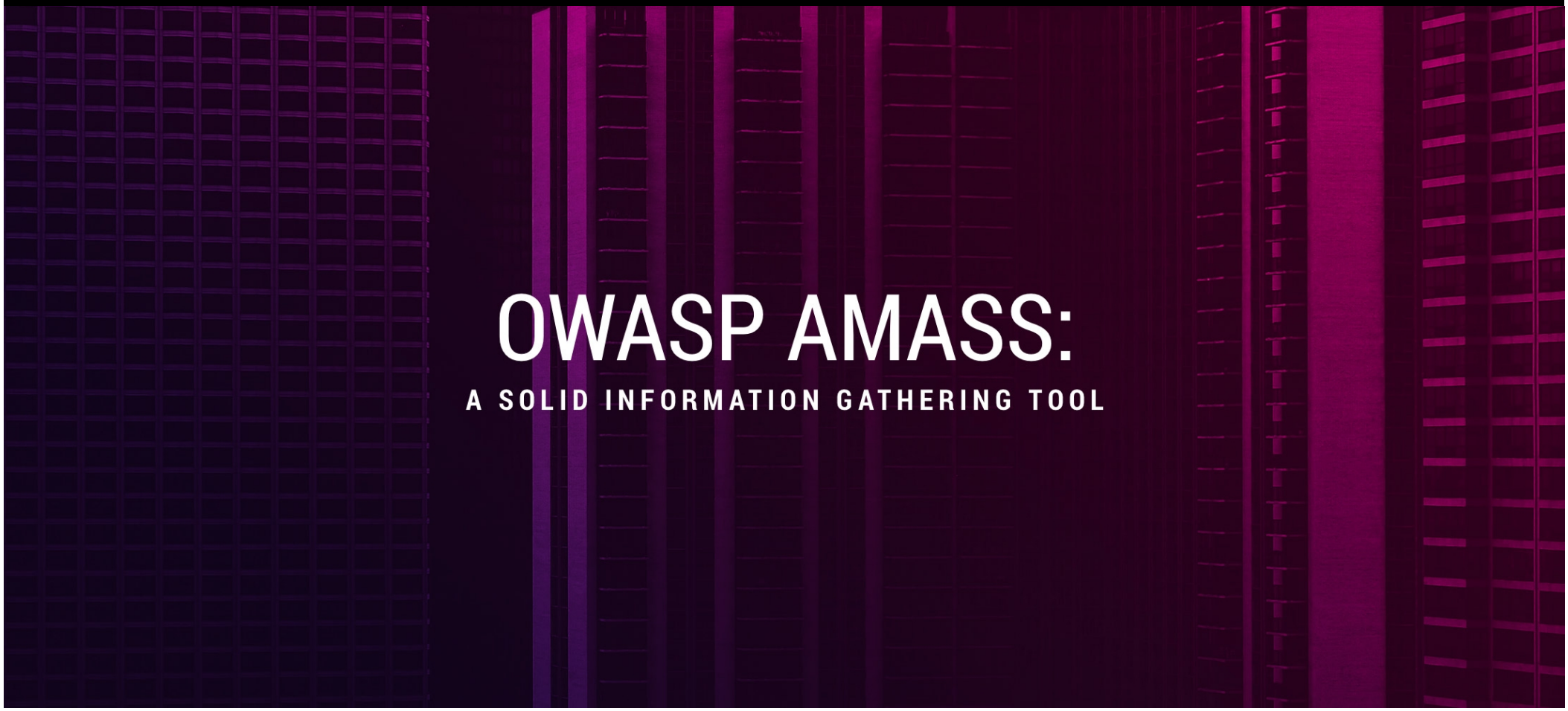

TIPS TOOLS RECONNAISSANCE

SECURITYTRAILS BLOG · APR 02 · NICOLÁS PENCE & THE SECURITYTRAILS TEAM

OWASP Amass: A Solid Information Gathering Tool

Reading time: 15 minutes

Facebook Twitter LinkedIn

Throughout history, human beings have crafted tools as a way to improve people's lives. From stone hammers to metal knives, through advancements from rudimentary medical instruments to breakthroughs made with industrial steam machinery.

SEARCH BLOG

Search...

TABLE OF CONTENTS

From the disruption of transistors and the computer era through today's technology that seems to come straight out of science fiction, like the storage of data in DNA, tools at the very least allow us to get more work done.

Tools afford us time and efficiency, and the security industry is no exception. Security tools are like what optical illusion is for magicians: they yield impressive results in brief periods of time, with a great impact on your audience. These digital instruments open multiple doors to a world of information that would otherwise be difficult to perceive.

Today we're introducing you to Amass, a true information-gathering 'swiss army knife' for your command line tool box. It was originally written by Jeff Foley (currently the Amass Project Leader) and later adopted by the OWASP Foundation.

What is Amass?

Amass is an open source network mapping and attack surface discovery tool that uses information gathering and other techniques such as active reconnaissance and external asset discovery to scrap all the available data. In order to accomplish this, it uses its own internal machinery and it also integrates smoothly with different external services to increase its results, efficiency and power.

This tool maintains a strong focus on DNS, HTTP and SSL/TLS data discovering and scrapping. To do this, it has its own techniques and provides several integrations with different API services like (spoiler alert!) the SecurityTrails API.

It also uses different web archiving engines to scrape the bottom of the internet's forgotten data deposits.

Installation

Let's start by installing this tool in our local environment. While it supports multiple software platforms, more interestingly, it supports different hardware architectures, which means that you could build your own automated box using a small but powerful ARM board—Raspberry PI for instance, or even a mobile phone!

Today our focus is to work on a 64-bit PC with Linux, but if you want to test it first and install it later, we strongly suggest you try out the docker image.

To install it from a pre-compiled binary, go to the **releases** section of their Github page. You can access it here, and a screenshot of available zip packages as well as the source code is shown below:

It’s very important (especially when using security tools) that you check the integrity of those downloaded binaries to be sure there has been no tampering whatsoever between what you intended to download and what you actually ended up downloading onto your hard drive.

To do that, you need to save the file amass_checksums.txt, which includes all the hash checksums corresponding to the binaries required to verify the authenticity of the OS binaries.

For the Amass 3.5.2 version (the latest release available at the time of this writing) the checksum file has the following contents:

As in this analysis we're using Linux in an amd64 CPU architecture, we are only verifying this hash (you can avoid this if you want, but the check will output several "No such file..." errors). To do so, we must first remove all non-corresponding hashes from the file, and invoke the following command:

```
$ shasum -c amass_checksums.txt
amass_v3.5.2_linux_amd64.zip: OK
```

With that result, we can be assured that the binary is correct, and that there were no file modifications while it was downloading.

Simply fetch the desired .zip file (in our case that would be amass_v3.5.3_linux_amd64.zip), uncompress it and enter the newly created folder (amass_v3.5.3_linux_amd64).

In it you will see different files and folders, the executable is called "amass", and when you run it, you'll see this:

This would be the end of the installation, but if you want it to be part of your $PATH, just move the amass binary to your favourite executables folder.

First steps

Let's take a look at the subcommands so we can check out the power of this tool:

```
Subcommands:

  amass intel - Discover targets for enumerations
  amass enum - Perform enumerations and network mapping
  amass viz - Visualize enumeration results
  amass track - Track differences between enumerations
  amass db - Manipulate the Amass graph database
  amass dns - Resolve DNS names at high performance
```

We are going to explain briefly what they do and how to activate them, and note a few of them while trying to dig a little deeper than the ones mentioned in the official tutorial or user guide, for both fun and educational purposes!

Presentation by obfuscation

One particular object within the intel subcommands is the "**-demo**" flag, which allows us to output results in an obfuscated manner. This way, we can make presentations without revealing too much information about our targets.

In this example we are conducting an intelligence gathering action and obtaining an obfuscated output with the use of the -demo flag in the command line. This will replace TLDs and ccTLDs with 'x' characters:

```
$ amass intel -asn 6057 -demo
adsl.xxxxxxxxx.xxx.xx
```

```
ancel.xxx.xx
ir-static.xxxxxxxxx.xxx.xx
algorta.xxx.xx
estudiocontable.xxx.xx
fielex.xxx.xx
ain.xxx.xx
vyt.xxx.xx
com.xx
cx2sa.xxx
easymail.xxx.xx
mor-inv.xxx
catafrey.xxx
kernel.xxx.xx
bglasesores.xxx
sislersa.xxx
arpel.xxx.xx
copab.xxx.xx
spefar.xxx
aitacargas.xx
duprana.xx
sua.xxx.xx
gruporovira.xxx
exterior.xxxxx.xxx
lideco.xxx
seaairforwarders.xxx
flp.xx
acac.xxx.xx
cerrolargo.xxx.xx
esquemas.xxx
```

Autonomous system number inquiry

Autonomous systems are the true guardians of internet communications. They know exactly how your traffic can reach a certain destination by receiving and advertising routing information. If this sounds at all interesting, let us tell you, it is. Let's dive in a little more, in detail.

Every organization connected to the internet with IP ranges to advertise, or that want multihomed broadband connections (for example, "I'm a cloud provider, and want to advertise my own delegated IP range to two different ISPs"), should have an autonomous system (AS for short). An autonomous system number (abbreviated as ASN) is the ID number of this AS given by your region's NIC authority (you can find more information on this topic in our previous article about ASN Lookup).

So what could you possibly do with this command? For one thing, you could dig up all the domain names associated within an entire ASN—that means a complete cloud provider (e.g., Google or Microsoft have their own ASN), an entire mega company (Apple, Akamai and Cloudflare also have their own ASNs), or an entire ISP (internet service providers of all sizes have ASNs, even the medium-sized and smaller ones). Internet exchanges and regional internet registries (IXs and RIRs, respectively) also have associated AS Numbers.

So how can we perform an ASN check? Once you have your target's ASN you can find out what's in there, as in the following example:

```
$ amass intel -asn 28000
labs.lacnic.net
lacnic.net.uy
lactld.org
dev.lacnic.net
lacnic.net
lacnog.org
net.uy
lacnog.lat
ripe.net
```

Here we have just queried the LACNIC (Latin American and Caribbean RIR) ASN; now we'll try with the RIPE (Europe, Middle East and Central Asia RIR) ASN:

```
$ amass intel -asn 25152
root-servers.net
```

Despite the few results, those domains (especially the last one, corresponding to RIPE's ASN) are some of the most important names on the internet (check out our piece on DNS Root Servers).

Great! What else can we do with this tool? Let's find AS Numbers by way of description. This is incredibly useful, as you can get records quickly and avoid gathering data manually, by looking at companies' websites, looking glass

tools, and more.

By following the previous example, we'll be looking at some of the AS Numbers regarding other existing RIRs (hint: ARIN corresponds to North America, AFRINIC to Africa and APNIC services to Pacific Facing Asia and Oceania):

- ARIN Results

```
$ amass intel -org ARIN
3856, PCH-AS - Packet Clearing House
3914, KMHP-AS-ARIN - Keystone Mercy Health Plan
4441, MARFORPACDJOSS - Marine Forces Pacific
6656, STARINTERNET
6702, APEXNCC-AS Gagarina avenue
6942, CLARINET - ClariNet Communications Corporation
7081, CARIN-AS-BLOCK - ISI
7082, CARIN-AS-BLOCK - ISI
7083, CARIN-AS-BLOCK - ISI
7084, CARIN-AS-BLOCK - ISI
7085, CARIN-AS-BLOCK - ISI
9489, KARINET-AS Korea Aerospace Research Institute
10034, GARAK-AS-KR SEOUL AGRICULTURAL & MARINE PRODUCTS CORP.
10056, HDMF-AS Hyundai Marin & Fire Insurance
10065, KMTC-AS-KR Korea Marine Transport
10309, MARIN-MIDAS - County of Marin
10439, CARINET - CariNet
10715, Universidade Federal de Santa Catarina
10745, ARIN-ASH-CHA - ARIN Operations
10927, PCH-SD-NAP - Packet Clearing House
```

```
11179, ARYAKA-ARIN - Aryaka Networks
11187, GWS-ARIN-AS - Global Web Solutions
11228, ARINC - ARINC
11242, Universidade Federal de Santa Catarina
```

- AFRINIC Results

```
$ amass intel -org AFRINIC
33764, AFRINIC-ZA-JNB-AS
37177, AFRINIC-ANYCAST
37181, AFRINIC-Anycast-RFC-5855
37301, AFRINIC-ZA-CPT-AS
37708, AFRINIC-MAIN
131261, AFRINIC-AS-AP Temporary assignment due to software testing
```

- APNIC Results

```
$ amass intel -org APNIC
4608, APNIC-SERVICES Asia Pacific Network Information Centre
4777, APNIC-NSPIXP2-AS Asia Pacific Network Information Centre
9450, WEBEXAPNIC-AS-AP Webex Communications Inc
9838, APNIC-DEBOGON-AS-AP APNIC Debogon Project
17821, APNICTRAINING-ISP ASN for APNICTRAINING LAB ISP
18366, APNIC-ANYCAST-AP ANYCAST AS
18367, APNIC-UNI1-AP UNICAST AS of ANYCAST node(Hongkong)
18368, APNIC-SERVICES APNIC DNS Anycast
18369, APNIC-ANYCAST2 APNIC ANYCAST
```

```
18370, APNIC-UNI4-AP UNICAST AS of ANYCAST node(Other location)
23659, HEITECH-AS-AP APNIC HEITECH ASN
24021, APNICRANDNET-TUI-AU TUI experiment
24555, APRICOT-APNIC-ASN ASN used for conferences in AP region
38271, CSSL-APNIC-2008-IN CyberTech House
38610, APNIC-JP-RD APNIC R&D Centre
38905, CPHAPNIC-AS-AP Consolidated Press Holdings Limited
45163, APNICRANDNET-TUI2-AU TUI experiment
45192, APNICTRAINING-DC ASN for APNICTRAINING LAB DC
55420, SABAHNET-AS-AP APNIC ASN Block
```

This set of outputs shows us which AS Numbers have a description matching our search criteria, which is extremely useful and fast!

Other intel capabilities may include:

- Reverse WHOIS queries
- Active reconnaissance
  On this topic, we could say that the flag -active gives a proactive check against additional sources of information, checks what active SSL/TLS Certificates this server has and provides us with additional information (enabling the -src flag allows us to see which source of information was queried as well as the corresponding result).

```
$  amass intel -addr 8.8.8.8 -src
[Reverse DNS] dns.google
```

```
$  amass intel -active -addr 8.8.8.8 -src[Reverse DNS] dns.google
[Active Cert] dns.google.com
[Active Cert] 8888.google
```

The **intel subcommand** provides different ways to output information, and to make further checks like port scanning, you can take a look at more details in the github documentation page.

Reconnaissance

How about plain and simple DNS record enumeration? Amass provides this using the **enum subcommand**, and it queries multiple different sources of information to check the existence of domain related subdomains.

In the image below you can see the different backends this tool relies on to find information. Some of them are quite peculiar, as in the case of the Pastebin website check.

Now here's an example of how adding an ASN to the query can obtain additional information. In the first query we explicitly added the AS Number, and for the second query it was removed. The results speak for themselves:

```
$ amass enum -asn 28000 -d lacnic.net
```

```
$ amass enum -d lacnic.net
```

To summarize, adding a context (e.g., ASN) could help with getting more information from your query.

Scraping unpublished subdomains

Sometimes subdomains won't show up. They can be a bit inactive, and when queried, their activity goes far below the radar. So how do we get them? Meet subdomain brute forcing. This technique allows us to bring in our custom wordlist and try it against the configured domain name, in an attempt to find or discover unseen subdomains.

In this case, we are going to guess...

```
$ amass enum -brute -src -d ripe.net -demo
```

The output will look similar to this:

Tracking

Is there anything really static on the internet? Well, there are no simple answers to this question. The volatility that DNS records, BGP routes, and IP space have on the internet is astonishing. In this scenario, we will use the **track subcommand**, to see exactly what has changed between our previous checks. The results may surprise you:

Apparently, in a window of roughly six hours, two subdomains' AAAA records were removed.

The track subcommand allows us to match information between checks and output the difference between them, so you can get an idea of how quickly a target is moving.

For more detail, you can add the -history flag, and this will output different time frames and activity.

The following example accounts for the -history flag, so you can see how the output will look:

Wait, there's data storage too?

The short answer is yes. Amass implements a graph database that can be queried after every check to see which records have been modified, and to speed up query results.

The following command will give us a summary on the data we've collected, showing the sources of information ordered by ASN and IP range in conjunction with a list of the domains and subdomains discovered.

```
$ amass db -show
```

To perform further data analysis, it's helpful to get a unique identification number that will let you trace the different checks you have made (it helps when trying to filter and visualize data later). To get this ID you'll need to run the following:

Then, if desired, you can get a summary—or the full output—regarding the investigation. For the sake of brevity, let's just do a summary output of index number **1** corresponding to the **ripe.net** analysis:

Visualization

If you've come this far, you probably have an overall idea of the power of this tool, and you're probably plenty excited about the data outputs and the colourful shell results as well!

But what if you want to extract these findings and take them to the next level? Perhaps you want to be able to show your target ecosystem, and make a nice presentation of it.

If that's the case, then Amass has you covered. Let's meet the **viz subcommand**.

That image you see below could be one of the possible outputs. To gain some insight, the small red dot at the center corresponds to the ripe.net domain, and the satellites connected to it are different objects that represent associated data such as IP addresses, DNS records, etc.

You can zoom in to see what all this is composed of, namely:

- Red dots - domain names
- Green dots - subdomain names
- Yellow dots - reverse pointer records
- Orange dots - IP addresses (v4 & v6)
- Purple dots - mail exchanger records (MX)
- Cyan dots - nameserver records (NS)
- Blue dots - autonomous system numbers (ASNs)
- Pink dots - IP netblocks

As you zoom in on this example, your visualization would look like the one below:

And if you want to look at (almost) the whole ecosystem, you can zoom out to see the “big picture”.

You can make your own visualization, focused on a specific analysis. Now we are taking the index identification number (8) obtained in the previous section and opening the output file that we need to visualize:

```
$ amass viz -enum 8 -d3
```

The animated visualization HTML file will look similar to this:

What about API integrations?

While it may seem like this tool has no need for configuration files, that's not entirely true. When an API interaction is needed, you need to conveniently save your keys so you can use this feature recurrently.

To do that, we are going to set up an **Amass configuration file**, with the necessary information for the APIs to work, and throw down some commands to see how the tool behaves.

We can start by downloading an example should be "configuration file from this link.

Config file locations should be placed like those stated in the following table depending on your deployment:

You can also place a different config file for testing purposes, if you set the -config flag.

When it comes to enabling an API key, it's pretty straightforward. Just uncomment the desired API section and place your SecurityTrails-provided API key:

```
[SecurityTrails]
apikey =  YOUR_API_KEY
```

Then, simply invoke the desired command, using the config file with the API key in it, in our case:

We can see that the SecurityTrails API integration is enabled, now let's see if we can get any results by using it:

Great! You've just learned how to enable an API key and execute a query using it. If you need more information, just look into the config file for more useful data about integrations and third-party services.

Summary

While this tool is an amazing resource for finding data about any target, it's somewhat vague in its documentation and how it actually works. Of course, you're probably thinking, "Why do I need to know how they do it?", but understanding how it gathers data according to a determined method makes it easier to determine how accurate the result is.

You could be obtaining a domain associated to a determined ASN that bears no logical relation at first sight without a proper explanation, but is still listed. If you search in the most common places, such as in WHOIS, DNS A MX TXT or reverse pointer (rDNS or PTR) records, etc., you won't easily find a good reason for its appearance within the output.

Of course, if you're familiar with Golang, you can read first-hand how it's been created, and yes the -src flag can shed light on where data is pulled from, but if you're not up for a source code review, a little more in the way of in-depth documentation (explaining how every check works) would be really nice.

All in all, know this: Amass should definitely be included in your security toolbox.

As we've seen in this article Amass integrates smoothly with our Cybersecurity API, letting users expand their data reconnaissance results in an instant.

However, that's not all we offer, our flagship enterprise-product SurfaceBrowser™ is a complete infosec command-center that can also be the perfect complement for any of your infosec research tasks. Contact our sales team and schedule a call today!

name@company.com Subscribe

‹ PREVIOUS NEXT ›

Related Posts

What is Reverse DNS? Top Tools for Performing a Reverse DNS Lookup

Learn what is Reverse DNS, and the top tools to perform a reverse DNS Lookup from the terminal, using a rDNS API or from a web-based interface.

Top 7 Subdomain Scanner Tools: Find Subdomains in Seconds

Today, we'll explore the top 7 subdomain scanning tools to boost effectiveness in your daily subdomain reconnaissance tasks.

Whois IP: Top 4 tools to perform a WHOIS IP Lookup

Learn what is a WHOIS IP lookup, and the top ways to perform a WHOIS against any IP using classic commands and new WHOIS IP lookup tools.

LATEST FROM OUR BLOG

- NEW All Things Quantum - Quantum Security Series - Part 3
- New SurfaceBrowser™ Features: Company Activity & New Associations
- GOSINT: A Framework for Collecting, Processing, and Exporting Indicator...
- DevSecOps: Ingraining Security in the Software Development Process
- Random and Mysterious - Quantum Security Series - Part 2