Hacker tools: Amass – hunting for subdomains

BY INTIGRITI · JUNE 8, 2021 · LAST UPDATED ON MARCH 24, 2026

Welcome to our *hacker tools* series. In the past weeks, we discussed some useful tools to help you with your bug bounty career. This week we will discuss Amass, the well-known subdomain discovery tool.

Amass is a tool that uses passive and active information gathering techniques to compile a nice list of an organization's externally exposed assets. What differs Amass from the rest is the ability to integrate external services through API keys.

The OWASP Amass project is actively being maintained on the git page https://github.com/OWASP/Amass.

The installation

All information can be found on the Github page of the OWASP Amass project: https://github.com/OWASP/Amass. We will go through the installation process together for faster deployment.

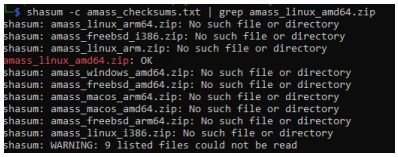
Go to the release page and download the latest package for your system and the checksum file at: https://github.com/OWASP/Amass/releases/

```
My case:
checksum:  download/v3.13.0/amass_checksums.txt
file:      download/v3.13.0/amass_linux_amd64.zip

wget github.com/OWASP/Amass/releases/<checksum>
wget github.com/OWASP/Amass/releases/<file>
```

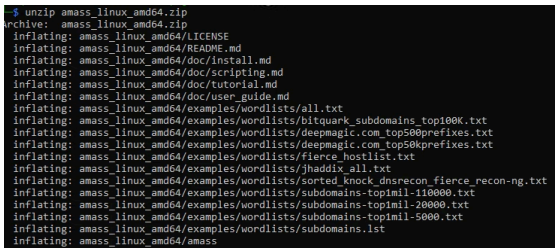
To check if our downloaded file matches the checksum we need to execute the *shasum* command. This will output an OK message on the file you downloaded, indicating this is valid.

```
shasum -c amass_checksums.txt | grep amass_linux_amd64.zip
```



Extract the package, in my case the Zip file.

```
unzip amass_linux_amd64.zip
```



Go into the directory and run Amass to check the installation.

```
./amass
```

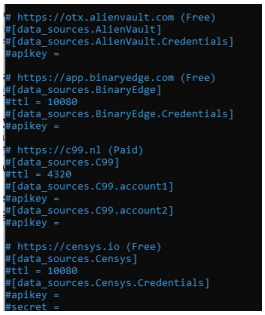
Configuring API keys:

Amass uses lots of external sources to get you the information you want. For some services, this involves using API keys. Let's configure Amass with some API keys so we can make full use of the tool.

First, download the example configuration file. This can be found at https://github.com/owasp-amass/amass/blob/e3e98f9dee4305e920da9b4094d880acc44b6907/examples/config.ini

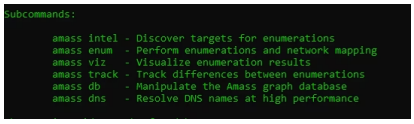
In order to use API keys, you need to register on the corresponding sites in the config file (found in the [data source] section), then request the API key and paste this key into the config file. Some services are free and some have paid plans. It's up to you what you configure.

Now that everything is set up we are ready to use the tool. To use your config file the parameter (**-config**) must be used.



The Basics

Amas has a set of subcommands, each with its own options. We will go over them and see what every set can do.



If you have configured your config file with API keys and other options, you can add this by using the (**-config**) flag.

A full list of examples is available on the user guide at:
https://github.com/OWASP/Amass/blob/master/doc/user_guide.md

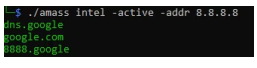
Amass intel Module

Our first subcommand is the Intel command, this module will focus on collecting opensource intelligence and is good for finding root domains and additional subdomains. To view all options in this section run the *amass intel* command.
Some interesting ones are:

SSL grabbing:

We can grab domain names from SSL certificates with the (**-active**) flag. This in combination with an IP range can give interesting results.

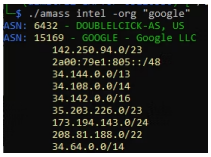
```
./amass intel -active -addr 8.8.8.8
```



Finding ASN nr's:

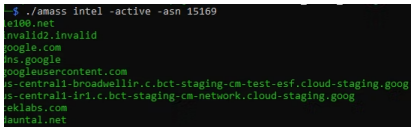
An autonomous system number is a unique identifier that is globally available and allows its autonomous system to exchange routing information with other systems. If we find this number, we can extract more information.

```
./amas intel -org “google”
```



Now that we have the ASN nr we can look for more domains.

```
./amass intel -active -asn 15169
```



Setting a default timeout:

Amass can run for a long time when executed on large scopes. To limit your search time we can set a timeout. This value is in minutes.

```
./amass intel -timeout 60 -d google.com
```

These were a few options available from the intel subcommand. You also can chain those together and mix them up to get as many results as possible.

Amass enum Module:

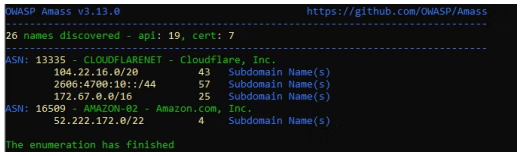
This module is probably the most used feature from Amass. Enum will try to find subdomains from the root domains you provide. Check all options with *./amass enum* or check out the user guide at https://github.com/OWASP/Amass/blob/master/doc/user_guide.md

With the enum module, we can do passive and active scanning. The passive scanning is way quicker but doesn’t validate the subdomains found. When using the (**-passive**) flag, not all options are available.

```
./amass enum -passive -d owasp.org -src -config config.ini
```

When using Amass in active mode, this will take longer but will give more accurate results. This in combination with some parameter tweaking can give good results. The most basic enum command only needs a domain. I will provide the config file and the -src flag to show where Amass gets its information.

```
./amass enum -active -d owasp.org -src -config config.ini
```



Wordlists:

You can feed your own custom wordlists with the (**-aw**) flag for better results.

```
./amass enum -aw <PATH> -d owasp.org
```

Feeding root domain names:

With the list of root domain names we gathered from the intel module, we can feed these to Amass with the (**-df**) flag in a file format. Keep in mind these scans can take a long time.

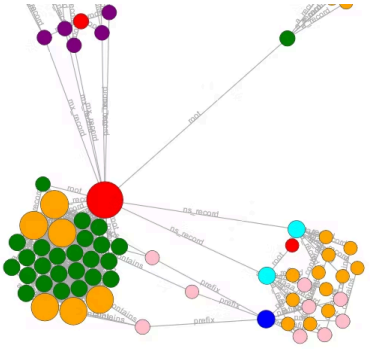
```
./amass enum -df domains.txt
```

There are many more options to explore in the enum module, check out the user pages for more detailed examples.

Amass viz Module

Most hunters will not use this module, as it will generate a visualization of links found between domains, but it is nice to quickly show you. There are different outputs available and one of them is a nice interactive HTML page, showing all the connections. For more options run *./amass viz*.

```
./amass viz -d3 -d owasp.org
```



Amass track Module

Organizations change, new domains and subdomains are added every day. Amass has a nice module to track those changes and report them back to you. When you do a scan with Amass, it stores this onto your computer. When you later do this scan again, you can discover newly added assets. This is very powerful if you would be to automate this process. Run *./amas track* to see the options.

```
./amass track -d owasp.org
```

Amass db Module

The db module is basically a log from all the scans you did in the past. You can retrieve previous scans and see the results. Here a few examples

All scans:

Show a list of all scans done by Amass

```
./amass db -list
```

Specific scan results:

When you want to view a specific scan from a previous run, you need the (**-show**) flag.

```
./amass db -show -d owasp.org
```

Conclusion

Amass can discover lots of hidden assets that give new attack vectors. The lists of newly discovered domains can be used to chain your workflow with other tools. But to make sure you have full use of the tool you need to configure as many API keys as possible. I hope you enjoyed our article and have a nice day discovering all those new subdomains.