OWASP amass USER MANUAL

Author : Özgür Koca, ozgurkoca.com, June 2022

Github : https://github.com/enseitankado/owasp-amass-diagrams

OWASP Amass performs network mapping of attack targets and external asset discovery using open source information gathering and active discovery techniques. Check out the techniques the tool uses to find subdomains here. Amass , an OWASP project, is one of the important blueteam tools used in the field of open source intelligence (OSINT) . Written in the Go language, the main focus of the tool is domain name intelligence and discovery.

The OWASP Amass tool obtains subdomains by scraping data sources, brute-force recursively, scanning web archives, allowing/changing names, and reverse DNS scanning. Additionally, Amass uses IP addresses obtained during resolution to discover associated network blocks and ASNs. All the information is then used to create maps of target networks. The research and discovery techniques Amass uses include:

- **Modification/Permutation** : Primitive changes to found subdomains (dev1 <-> dev2). When it sees a subdomain like customer1.domain.com, it also scans sequential subdomains like customer2.domain.com.
- **Fuzzy tag/string search** : Addition subtraction (dev <-> dav). When it sees a subdomain such as dev.domain.com, it tries to find possible subdomains with simple letter changes.
- **Recursive brute force** : searching APIs and WEB archives. It searches for subdomains from archives such as wayback.com and archiveit.com. The data sources he explored are listed below. Some of these are limited services that need API key.
- **Active techniques** : Downloading HTTP TLS/SSL certificates. Common Name contains subdomain information.

- **Zones** : Monitoring zone transfers and zone walking (NSEC records).
- **Public sources** : Web archives, pastebin, version control and social media websites.
- **Certificate Transparancey (CT) logs** : CT is a framework used to track certificates and verify whether a certificate is safe or malicious. It tries to discover using the certificates defined for the domain.
- **HTTP CSP (Content Security Policy) headers** : When subdomains are included in the policies, they are whitelisted here to protect them from CSS attacks. This creates a data source for amass in whitelists.

Below are the data sources Amass uses for research and discovery. You can run the following command to see the current data sources and availability of amass.

```
amass enum -list
```

Amass can use around 50 data sources, some of them free, some with trial use and some quite expensive services. You can add the -src parameter to your commands to keep track of the most successful data sources. So you can keep track of which source returns results and how often.

TECHNICAL	DATA SOURCE
APIS	360PassiveDNS, Ahrefs, AnubisDB, BinaryEdge, BufferOver, BuiltWith, C99, Chaos, CIRCL, Cloudflare, DNSDB, DNSRepo, Detectify, FOFA, FullHunt, GitHub, GitLab, Greynoise, HackerTarget, Hunter, IntelX, LeakIX, Maltiverse, MHT45 PassiveTotal, PentestTools, Quake, Shodan, SonarSearch, Spamhaus, Spyse, Sublist3rAPI, ThreatBook, ThreatCrowd, ThreatMiner, Twitter, URLScan, VirusTotal, ZETAlytics, ZoomEye
CERTIFICATES	Active pulls (optional), Censys, CertSpotter, Crtsh, Digitorus, FacebookCT, GoogleCT
DNS	Brute forcing, Reverse DNS sweeping, NSEC zone walking, Zone transfers, FQDN alterations/permutations, FQDN Similarity-based Guessing
ORIENTATION	ARIN, BGPTools, BGPView, IPdata, IPinfo, NetworksDB, RADb, Robtex, ShadowServer, TeamCymru
SCRAPING	AbuseIPDB, Ask, Baidu, Bing, DNSDumpster, DuckDuckGo, Gists, HackerOne, HyperStat, IPv4Info, PKey, RapidDNS, Riddler, Searchcode, Searx, SiteDossier, Yahoo
WEB ARCHIVES	AlienVault, AskDNS, DNSlytics, ONYPHE, SecurityTrails, SpyOnWeb, Umbrella, WhoisXMLAPI
WHOIS RECORDS	AlienVault, AskDNS, DNSlytics, ONYPHE, SecurityTrails, SpyOnWeb, Umbrella, WhoisXMLAPI

Amass is a very comprehensive and advanced tool. The tool contains 5 subcommands, intel, enum, viz, track and db. **The intel** command does reconnaissance on the target. Useful to set a starting point for the destination. **enum** maps the target to identify possible attack points. **viz** helps in better analysis by visualizing the results obtained. **track** is used to track and compare changes over time on the target. amass saves all intelligence and scan results in its own database. The **db** subcommand is used to access and query this database. You can refer to the document page to access all the commands and options of amass . In this chapter, I will only exemplify its basic uses.

Amass Subcommands

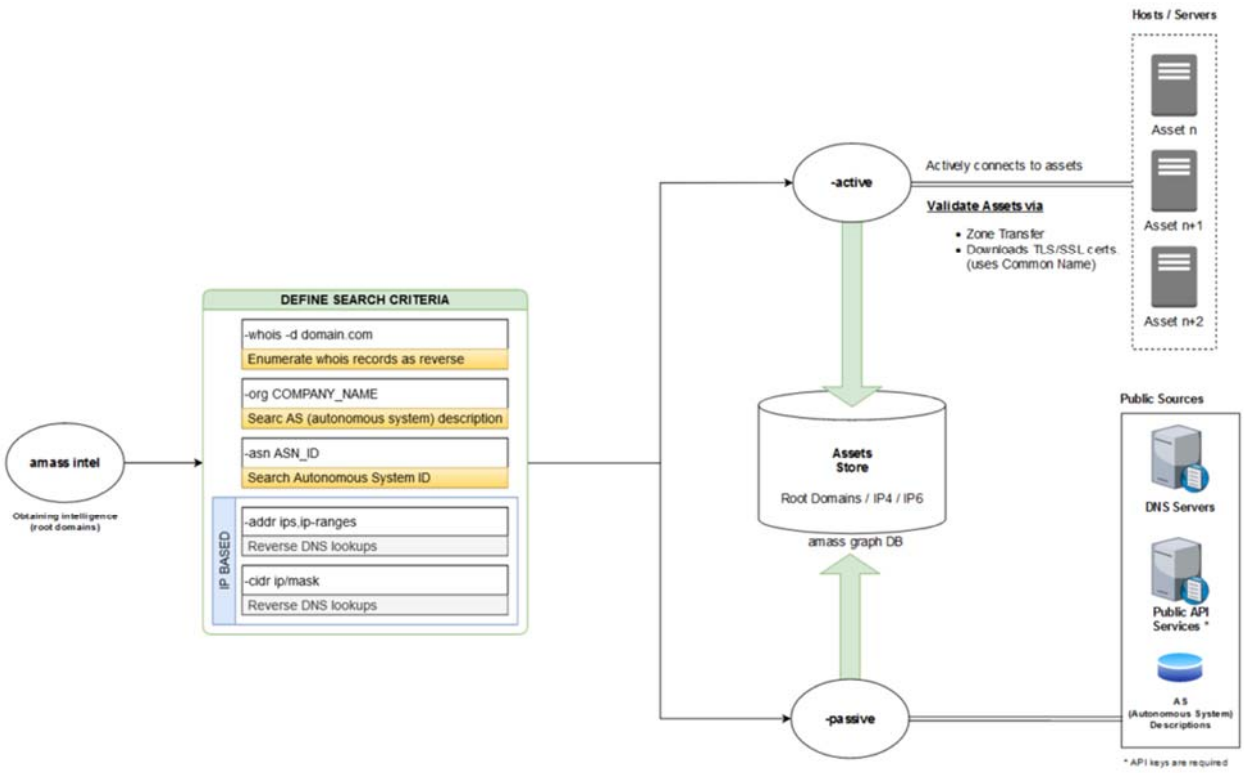
- **amass intel** : Discovers root domains (not subdomains), ASNs, performs reverse WHOIS & DNS queries. It conducts open source intelligence for the investigation of the target organization.
- **amass enum** : Extracts and maps DNS records of open systems using subdomains, fast mode (passive), normal mode, DNS resolution/validation techniques.
- **amass viz** : Creates visual graphics, good for visual inspection. Supports Maltego. Creates a visual diagram of the discoveries and researches made.
- **amass track** : Allows historical comparison between scans and seeing new or updated assets.
- **amass db** : Amass stores all active and passive discoveries in its own database. The db command is used to access these records later.

intel command

Amass intel subcommand/module can help gather open source intelligence about the organization and allows you to find more root domain names associated with the organization. To see the available options for this subcommand, simply type it into the terminal:

```
$ amass intel
[...]
Usage: amass intel [options] [-whois -d DOMAIN] [-addr ADDR -asn ASN -cidr CIDR]
-active
Attempt certificate name grabs
-addr value
IPs and ranges (192.168.1.1-254) separated by commas
-asn value
ASNs separated by commas (can be used multiple times)
-cidr value
CIDRs separated by commas (can be used multiple times)
-org string
Search string provided against AS description information
-whois
All provided domains are run through reverse whois
[...]
```

At this point, it's worth noting that another big advantage of Amass is that all subcommands try to maintain argument consistency. In the following paragraphs, the parameters of the subcommands are included, you can see that most of them are used in common. The diagram below basically shows the working logic of amass intel.



By default, this subcommand will use a set of information gathering techniques and data sources, such as WHOIS and IPv4Info, to obtain organizational intelligence and parent domains, unless explicitly disabled in Amass's configuration file. A sample Amass configuration file is available in the GitHub repository .

```
$ amass intel -whois -d owasp.org
appseceu.com
owasp.com
appsecasiapac.com
appsecnorthamerica.com
appsecus.com
[...]
owasp.org
appsecapac.com
appsecla.org
[...]
```

You can also confirm some of the above results by manually browsing the data sources. In the screenshot below, we did a reverse Whois lookup for "OWASP Foundation" and asked ViewDNS (which is also part of Amass' data sources) for similar domains:

https://viewdns.info/reversewhois/?q=owasp%20foundation

```
Reverse Whois results for OWASP Foundation
==============

There are 15 domains that matched this search query.
These are listed below:
```

Domain Name	Creation Date	Registrar
appseccali.org	2013-07-16	GODADDY.COM, LLC
appseccalifornia.org	2013-07-16	GODADDY.COM, LLC
appsecil.com	2017-09-01	GODADDY.COM, LLC
appsecil.info	2017-09-01	GODADDY.COM, LLC
appsecil.org	2017-09-01	GODADDY.COM, LLC
appsecli.info	2017-09-01	GODADDY.COM, LLC
appsecli.org	2017-09-01	GODADDY.COM, LLC

When searching with Amass intel, you can always run it with more configuration options, for example, you can try zone transfer with the --active argument and have it connect to the relevant server to get the SSL/TLS certificates to extract the information. Just make sure you are authorized to make active searches against the target before doing so.

At this point, it's worth noting that some configuration flags will not work with others, in which case Amass will ignore them.

Amass' findings may not always be accurate for a variety of reasons, for example the data sources used by Amass may not be consistent or up-to-date. Amass tries to further validate the information using DNS queries. While Amass does a good job, users should still run more validation checks on results that don't seem relevant to the target. This can be accomplished using a variety of methods such as:

- Use utilities (eg dig, nslookup) to resolve domains.
- Perform WHOIS searches to verify institutional details.
- Use search findings like main domains in search engines.

```
amass intel -org TURKEY
```

Returns the ASN IDs related to the word Turkey. This type of string searches often does not return enough results. You can also add the -active switch if you want the search to be performed using active methods as well. The above command will search for Turkey in the ASN descriptions and return the associated ASNs.

The org key searches for ASN IDs (Autonomous system) associated with the given expression. The string specified with the -org parameter is searched in AS records and its ASN ID is found. ASN ID is the autonomous system number assigned by the Internet Assigned Numbers Authority (IANA), it is represented by a worldwide unique 16-digit identification number. An AS (Autonomous System) refers to a large network or group of networks with a single routing policy (rfc1930). Identifies them in the ASN ID. For example, the ASN number of TTNET's network is 9121, CloudFlare's 13335, ULAKNET's 8517 and Amazon's 16509. Large routers take advantage of these ASN numbers when routing needs to be done between networks.

Now let's try to find the domain names in an ASN we found. Check out the command below.

```
amass intel -active -src -ip -asn 8517 -p 80,443
```

It will make a discovery using active techniques (-active) on ports 80 and 443 of domains in the autonomous network (ULAKNET) with ASN number 8517. It will use TLS/SSL certificates for active discovery. amass will pull the SSL certificate that will connect to all IP addresses of the specified ASN and list the domain to which the SSL certificate is associated. It will also display the sources (-src) and ip (-ip) addresses of the domains it finds.

```
amass intel -active -ip -src -cidr 193.140.28.0/24 -p 80,443
```

A string range is declared with the -cidr parameter. The number 24 represents the netmask and corresponds to 255.255.255.0. amass will search and discover different sources for IP addresses between 193.140.28.1 and 193.140.28.254, detecting and listing the domain names directed to these IPs. Also, active discovery will be done using TLS/SSL certificates for ports 80 and 443. It will connect to each IP address and pull SSL certificates and list the domain names mentioned in the certificate.

```
amass intel -active -ip -src -cidr 74.6.0.6/16 -addr 74.6.231.20-21 -p 80.8080
```

It will scan computers in network 74.6.0.6/16 using active techniques. It will list the ip address and source of the found domains. It will perform a reverse DNS query on computers in the 74.6.231.20-21 range.

```
amass intel -whois -d yahoo.com
```

It tries to find domain names by reverse searching the WHOIS records of the given domain (-d) or domain list (-df). In this example amass accesses the WHOIS records of yahoo.com and tries to find the root domains of other organizations that may be the same as those WHOIS records. It is useful for finding other domains belonging to an organization/company, but WHOIS records may not always contain accurate information or be kept confidential.

```
amass intel -asn 8517 -whois -d omu.edu.tr
```

This command searches for other domains with the same WHOIS records in the ULAKNET autonomous network with ASN ID 8517.

Below is a table showing the options and usage areas of the intel command.

CHOICE	EXPLANATION	SAMPLE

-ACTIVE	It uses active methods. Zone tries transfer, connects to target server to get SSL/TLS certificates.	amass intel -active -addr 192.168.2.1-64 -p 80,443,8080
-ADDR	Specifies IP ranges. Ranges are separated by commas.	192.168.2.1-64,192.168.3.10-254
-ASN	Reports ASN ID. ASN ID is a unique number assigned to large computer networks by IANNA.	amass intel -asn 13374,14618
-CIDR	Defines a network. The number after the slash defines the netmask.	amass intel -cidr 104.154.0.0/15
-CONFIG	Specifies the location of the scan configuration file.	amass intel -config config.ini
-D	Field names separated by commas.	amass intel -whois -d example.com
-DF	It feeds root domains from a file.	amass intel -whois -df domains.txt
IS	Specifies the path to the directory containing the database where the scan results will be stored.	amass intel -dir PATH -cidr 104.154.0.0/15
-EF	The path to the file with a list of data sources that will not be used when researching.	amass intel -whois -ef exclude.txt -d example.com
-EXLUDE	List of data sources that will not be used in scanning. It is specified by separating it with a comma.	amass intel -whois -exclude crtsh -d example.com
-IF	Path to the file with a list of data sources to use for scanning.	amass intel -whois -if include.txt -d example.com
-INCLUDE	List of data sources to use for scanning. It is written separated by commas.	amass intel -whois -include crtsh -d example.com
-ROPE	It also shows the IP addresses of the discovered domains.	amass intel -ip -whois -d example.com
-IPV4	It also shows the IPv4 addresses of the discovered domains.	amass intel -ipv4 -whois -d example.com
-IPV6	It also shows the IPv6 addresses of the discovered domains.	amass intel -ipv6 -whois -d example.com
-LIST	Lists the data sources to use in discovery.	amass intel -list
-LOG	Reports the path to the log file where errors will be logged.	amass intel -log amass.log -whois -d example.com
-MAX-DNS-QUERIES	Maximum number of concurrent DNS queries.	amass intel -max-dns-queries 200 -whois -d example.com
-HE	The path to the file where the output will be saved.	amass intel -o out.txt -whois -d example.com
-ORGAN	String expression to search for in the AS statement.	amass intel -org Facebook
-P	Comma-separated port numbers.	amass intel -cidr 104.154.0.0/15 -p 443,8080
-R	Declares DNS servers to be used during discovery.	amass intel -r 8.8.8.8,1.1.1.1 -whois -d example.com

-RF	Indicates the path of the file containing the list of DNS servers to be used during discovery.	amass intel -rf data/resolvers.txt -whois -d example.com
-SRC	Indicates the source from which the discovered fields were obtained.	amass intel -src -whois -d example.com
-TIMEOUT	Timeout period to consider during discovery.	amass intel -timeout 30 -d example.com
-WHOIS	WHOIS information searches for other similar domains.	amass intel -whois -d example.com

amass enum command

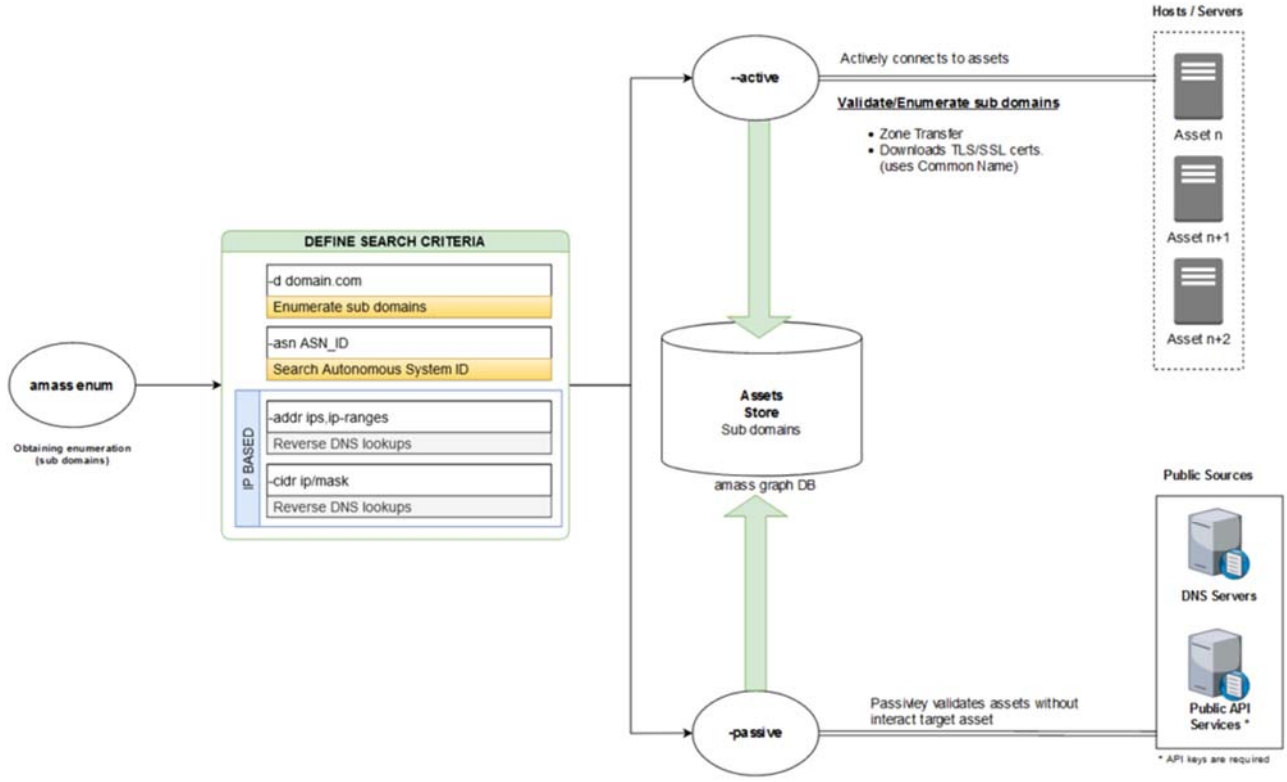
Amass enum can be run in a passive or active mode. Passive mode is much faster, but Amass will not validate DNS information by resolving subdomains. You can run it passively using the "-passive" flag and cannot enable many techniques or configurations such as DNS resolution and validation. Sometimes it is necessary to choose passive mode instead of active mode, for example:

- Because you need to constantly monitor the target scope for changes, or because you are working on a phishing interaction and searching for subdomains, you need to know all possible subdomains that have been used and may be reused in the future.
- You may need to verify DNS information at a later stage and spool results quickly.
- Due to the constraints or requirements of a security engagement, you may only need to perform passive information gathering.

To see the available options for this subcommand, simply type it into the terminal:

```
Usage: amass enum [options] -d DOMAIN

-active
Attempt zone transfers and certificate name grabs
-addr value
IPs and ranges (192.168.1.1-254) separated by commas
-d value
Domain names separated by commas (can be used multiple times)
-cidr value
CIDRs separated by commas (can be used multiple times)
-asn value
ASNs separated by commas (can be used multiple times)
```



In the example below, we are passively searching for subdomains on Owasp.org and asking Amass to display the data sources where it finds each subdomain:

```
$ amass enum -passive -d owasp.org -src
[...]
[ ThreatCrowd] update-wiki.owasp.org
[...]
BufferOver] my.owasp.org
[ crtsh ] www.lists.owasp.org
[ crtsh ] www.ocms.owasp.org
[...]
Querying VirusTotal for owasp.org subdomains
Querying Yahoo for owasp.org subdomains
[...]
```

This subcommand will perform DNS discovery and network mapping while probing. It's worth noting at this point that Amass enum will identify all subdomains, although the Amass intel command helps to collect the IP ranges, ASNs and main domains owned by an organization.

Using Amass in active configuration mode means you will have more accurate results and more assets can be discovered because you can enable all DNS discovery techniques. With "enabled configuration mode", it will search for subdomains from certificate domains (eg: Common Name) by performing zone transfer and port scans of SSL/TLS services .

can generally be considered active as it performs subdomain brute-forcing in multiple ways (word list, masks, etc.) with the "-active" flag enabled . All findings will be validated by Amass using the default or specified analyzers:

All the settings below are related to this subcommand. The following flags are available for configuration:

Choice	***Explanation***	***Sample***
-active	It uses active methods. Zone tries transfer, connects to target server to get SSL/TLS certificates.	amass enum -active -d example.com -p 80,443,8080
-aw	Indicates the path of the alternative word file to generate subdomain alternatives.	amass enum -aw PATH -d example.com
-bl	Reports a blacklist of subdomains that will not be discovered.	amass enum -bl blah.example.com -d example.com
-blf	Specifies the path of the file with the list of blacklisted subdomains.	amass enum -blf data/blacklist.txt -d example.com
-brute	Uses brute force in subdomain exploration.	amass enum -brute -d example.com
-config	Specifies the path to the INI configuration file.	amass enum -config config.ini
-D	Field names separated by commas.	amass enum -d example.com
-df	It feeds root domains from a file.	amass enum -df domains.txt
is	Specifies the path to the directory containing the database where the scan results will be stored.	amass enum -dir PATH -d example.com
-ef	The path to the file with a list of data sources that will not be used when researching.	amass enum -ef exclude.txt -d example.com
-exclude	List of data sources that will not be used in scanning. It is specified by separating it with a comma.	amass enum -ef exclude.txt -d example.com
-if	Path to the file with a list of data sources to use for scanning.	amass enum -ef exclude.txt -d example.com
-include	List of data sources to use for scanning. It is written separated by commas.	amass enum -include crtsh -d example.com
-rope	It also shows the IP addresses of the discovered domains.	amass enum -ip -d example.com
-ipv4	It also shows the IPv4 addresses of the discovered domains.	amass enum -ip -d example.com
-ipv6	It also shows the IPv6 addresses of the discovered domains.	amass enum -ip -d example.com
-json	Defines the path of the output file to be saved in JSON format.	amass enum -json out.json -d example.com
-list	Lists the data sources to use in discovery.	amass enum -list
-log	Reports the path to the log file where errors will be logged.	amass enum -log amass.log -d example.com
-max-dns-queries	Maximum number of concurrent DNS queries.	amass enum -max-dns-queries 200 -d example.com
-dns-qps	Maximum number of DNS queries per second across all domain resolvers (DNS server).	amass enum -dns-qps 200 -d example.com

-rqps	Maximum number of DNS queries per second for each untrusted resolver.	amass enum -dns-qps 200 -d example.com
-trqps	Maximum number of DNS queries per second for each trusted resolver	amass enum -trqps 20 -d example.com
-min-for-recursive	Subdomain labels seen before recursive brute forcing (Default: 1)	amass enum -brute -min-for-recursive 3 -d example.com
-max-depth	Maximum number of subdomain tags for brute forcing	amass enum -brute -max-depth 3 -d example.com
-nf	Path to a file providing already known subdomains (from other tools/sources)	amass enum -nf names.txt -d example.com
-noalts	Disables alternative subdomain generation.	amass enum -noalts -d example.com
-norecursive	Disables recursive brute force.	amass enum -brute -norecursive -d example.com
-He	Specifies the path to the text output file.	amass enum -o out.txt -d example.com
-oA	Path prefix used to name all output files.	amass enum -oA amass_scan -d example.com
-passive	It performs a completely passive execution.	amass enum --passive -d example.com
-p	Comma-separated port numbers.	amass enum -d example.com -p 443,8080
-r	Declares DNS servers to be used during discovery.	amass enum -d example.com -p 443,8080
-en	IP addresses of trusted DNS resolvers (can be used multiple times)	amass enum -en 8.8.8.8,1.1.1.1 -d example.com
-rf	Indicates the path of the file containing the list of DNS servers to be used during discovery.	amass enum -en 8.8.8.8,1.1.1.1 -d example.com
-trf	Specifies the path to a file that provides trusted DNS resolvers.	amass enum -trf data/trusted.txt -d example.com
-src	Indicates the source from which the discovered fields were obtained.	amass enum -src -d example.com
-timeout	Timeout period to consider during discovery.	amass enum -timeout 30 -d example.com
-w	Reports the path of a different wordlist file.	amass enum -brute -w wordlist.txt -d example.com

The most common usage of the enum command is as follows. After this command is given, amass starts to search the subdomains of meb.gov.tr domain name from the internet. It asks all data sources it can and returns a stack of subdomains.

```
amass enum -d meb.gov.tr
amass enum -passive -d yahoo.com -config ./passive-config.ini
```

In this scan, a (-passive) scan is performed without DNS resolution. Assuming passive-config.ini contains a list of passive information resources and API keys. The -passive option returns faster but fewer results than -active.

By using the -v parameter, you can increase the detail level of the amass.log file and better understand the features of amass while applying it. It allows you to detect data sources that are not working properly, services that are blocking your IP address, and DNS resolvers that are not working properly.

amass track command

To monitor a target's attack surface, it shows the differences between reconnaissance of the same target(s). This subcommand only takes advantage of 'output_directory' and the remote database settings specified in the config file.

To see the available options for this subcommand, simply type it into the terminal:

```
Usage: amass track [options] -d domain
-d value
Domain names separated by commas (can be used multiple times)
-history
Show the difference between all enumeration pairs
-last int
The number of recent enumerations to include in the tracking
```

Choice	***Explanation***	***Sample***
-config	Specifies the path to the INI configuration file.	amass track -config config.ini
-D	Field names separated by commas.	amass track -d example.com
-df	It feeds root domains from a file.	amass track -df domains.txt
is	Specifies the path to the directory containing the database where the scan results are stored.	amass track -df domains.txt
-history	It shows the differences between the discoveries.	amass track -history
-last	The number of recent discoveries to include in tracking.	amass track -last 2
-since	Exclude all enumerations before the specified date (format: 01/02 15:04:05 2006 MST)	amass track -since DATE

```
amass track -d yahoo.com -last 2
```

The above command compares the last two scans of the domain yahoo.com.

amass viz command

Creates visual network graphs using the collected information. This visualization is particularly useful for interpreting relationships between entities in large scans that return comprehensive results. This subcommand only takes advantage of the 'output_directory' and remote graphics database settings from the config file.

Files created for visualization are created in the current working directory by default. Here are the switches that can be used to output DNS and infrastructure findings as network graphs:

Choice	*Explanation*	*Sample*
-config	Specifies the path to the INI configuration file.	amass viz -config config.ini -d3
-D	Field names separated by commas.	amass viz -d3 -d example.com
-d3	Generates D3.js v4 formatted HTML simulation file.	amass viz -d3 -d example.com
-df	Defines the path to the file containing the root domains.	amass viz -d3 -df domains.txt
is	Defines the path to the directory containing the Graph database.	amass viz -d3 -dir PATH -d example.com
-enum	Scan results identify a discovery by its index number from the database.	amass viz -enum 1 -d3 -d example.com
-He	Defines the path to the directory where the output files will be saved.	amass viz -d3 -o OUTPATH -d example.com
-oA	Defines a prefix for output files.	amass viz -d3 -oA example -d example.com
-gexf	Outputs Graph Exchange XML Format (GEXF) formatted.	amass viz -gexf -d example.com
-graphistry	Graphistry returns JSON formatted output.	amass viz -graphistry -d example.com
-maltego	Outputs the Maltego Graph Table formatted CSV file.	amass viz -maltego -d example.com

```
amass viz -d yahoo.com -d3 -o yahoo
```

The above command displays all results for the domain yahoo.com in an HTML-formatted file with an associative graph. The chart is created using the d3 javascript library . With the -o option, the output is saved in the yahoo directory. The vizz command can also output in Maltego, XML, and JSON formats.

amass db command

It is a command that allows you to view the discovery data for each scan performed. Performs the viewing and processing of the database. This subcommand only takes advantage of the 'output_directory' and remote database settings from the config file. Usage options for interacting with exploration findings in the database are:

CHOICE	**EXPLANATION**	**SAMPLE**
-CONFIG	Specifies the path to the INI configuration file.	amass db -config config.ini
-D	Field names separated by commas.	amass db -d example.com
-DF	It feeds root domains from a file.	amass db -df domains.txt
IS	Specifies the path to the database directory containing the scan results.	amass db -dir PATH
-ENUM	Retrieves discovery information via an index sequence number from the list.	amass db -enum 1 -show
-IMPORT	Imports a JSON-formatted amass file.	amass db -import PATH
-ROPE	It also shows the IP addresses of the discovered domains.	amass db -show -ip -d example.com

-IPV4	It also shows the IPv4 addresses of the discovered domains.	amass db -show -ipv4 -d example.com
-IPV6	It also shows the IPv6 addresses of the discovered domains.	amass db -show -ipv6 -d example.com
-JSON	Defines the path of the output file to be saved in JSON format.	amass db -names -silent -json out.json -d example.com
-LIST	Lists the data sources to use in discovery.	amass db -list
-NAMES	Bring only discovered domains.	amass db -names -d example.com
-NOCOLOR	Disables color output.	amass db -names -nocolor -d example.com
-HE	Specifies the path to the text output file.	amass db -names -o out.txt -d example.com
-SHOW	Prints the results for the resulting discovery directory + domains.	amass db -show
-SILENT	Disables all output during execution.	amass db -names -silent -json out.json -d example.com
-SRC	Indicates the source from which the discovered fields were obtained.	amass db -show -src -d example.com
-SUMMARY	Prints only the ASN table summary.	amass db -summary -d example.com

To list all the details of your previous scans, simply run amass db -show so you can see the current results without the need for a new scan. If you want to see the details of a particular domain, simply add the -d option:

```
amass db -show -d paypal.com
```

If you prefer nice, clean, simple output, you can print the discovered domains/subdomains using the -names option instead of -show.

```
amass db -dir amass4owasp -d owasp.org -enum 1 -show
```

Displays the results of discovery #1 of the domain name owasp.org in the database registered in the amass4owasp directory.

Generating API keys

There are many data sources that Amass can use. I gave a list of them at the beginning of the article. However, the data sources listed below need the API key.

AlienVault, BinaryEdge, BufferOver, BuiltWith, C99, Censys, Chaos, CIRCL, DNSDB, DNSTable, FacebookCT, GitHub, HackerOne, HackerTarget, NetworksDB, PassiveTotal, RapidDNS, Riddler, SecurityTrails, Shodan, SiteDossier, Spyse, URLScan, Umbrella VirusTotal, WhoisXML, ZETAlytics, Cloudflare

Most of these API keys are free, but most only give limited results unless you have a paid API key. Free ones are better than nothing. You will need to find the website for each of the above services, then sign up and get an API key. This is a very time consuming and tedious task, but good discovery does not come without sacrifice. You can do this. Once you have all your API keys paste them into the amass config file. See here for a comprehensive example configuration file . Below is the content of a configuration file. Those who write Paid are paid. Obtain any of the data sources using the API in the sample file, add them as follows and create your own amass config file. Then save the file with the name amassconfig.ini for example:

```
# https://passivedns.cn (Contact)
[data_sources.360PassiveDNS]
[data_ sources.360PassiveDNS.Credentials ]
apikey = sizeozelapicode

# https://ahrefs.com (Paid)
[data_ sources.Ahrefs ]
ttl = 4320
[data_ sources.Ahrefs.Credentials ]
apikey = sizeozelapicode

# https://otx.alienvault.com (Free)
[data_ sources.AlienVault ]
[data_ sources.AlienVault.Credentials ]
apikey =sizeozelapicode
```

Now, when you use amass, specify the config file with the -config parameter as follows. This will increase the number and quality of your intel and enum discoveries. I strongly recommend you to do this in order to be one step ahead of the others.

```
amass enum -d turkiye.gov.tr -config ./amasconfig.ini
```

output directory

amass keeps log and output files for the last scan session under ~/.config/amass by default. These files are reset when a new scan is made. The log file should be examined at the end of the scan, as it records how the scanning process progresses and possible errors. Similarly, the scan output is stored in this directory in CSV format. It can be useful to use a different repository directory for each scan. You can give the -dir option to the command for this. Check out the example below:

```
amass intel -asn 47524 -src -ip -dir turksat -config amass-apis.ini
```

With the above command, TURKSAT network with ASN (autonomous network) number 47524 will be searched and the discoveries will be saved in the turksat directory.

References

Amass Project Website: https://owasp-amass.com/
Amass Source Code: https://github.com/OWASP/Amass
User guide: https://github.com/OWASP/Amass/blob/master/doc /user_guide.md
Usage examples: https://github.com/OWASP/Amass/blob/master/doc/tutorial.md
Haklukes' user guide: https://hakluke.medium.com/haklukes-guide-to- amass-how-to-use-amass-more-effectively-for-bug-bounties-7c37570b83f7

Copyright