NAME

ettercap - multipurpose sniffer/content filter for man in the middle attacks

***** IMPORTANT NOTE ******

Since ettercap NG (formerly 0.7.0), all the options have been changed. Even the target specification has been changed. *Please read carefully this man page.*

SYNOPSIS

ettercap [*OPTIONS*] [*TARGET1*] [*TARGET2*]

If IPv6 is enabled:
TARGET is in the form *MAC/IPs/IPv6/PORTs*
Otherwise,
TARGET is in the form *MAC/IPs/PORTs*
where IPs and PORTs can be ranges (e.g. /192.168.0.1−30,40,50/20,22,25)

DESCRIPTION

Ettercap was born as a sniffer for switched LAN (and obviously even "hubbed" ones), but during the development process it has gained more and more features that have changed it to a powerful and flexible tool for man-in-the-middle attacks. It supports active and passive dissection of many protocols (even ciphered ones) and includes many features for network and host analysis (such as OS fingerprint).

It has two main sniffing options:

UNIFIED, this method sniffs all the packets that pass on the cable. You can choose to put or not the interface in promisc mode (−p option). The packet not directed to the host running ettercap will be forwarded automatically using layer 3 routing. So you can use a mitm attack launched from a different tool and let ettercap modify the packets and forward them for you.
The kernel ip_forwarding is always disabled by ettercap. This is done to prevent a forward of a packet twice (one by ettercap and one by the kernel). This is an invasive behaviour on gateways. So we recommend you to use ettercap on the gateways ONLY with the UNOFFENSIVE MODE ENABLED. Since ettercap listens only on one network interface, launching it on the gateway in offensive mode will not allow packets to be rerouted back from the second interface.

BRIDGED, it uses two network interfaces and forward the traffic from one to the other while performing sniffing and content filtering. This sniffing method is totally stealthy since there is no way to find that someone is in the middle on the cable. You can look at this method as a mitm attack at layer 1. You will be in the middle of the cable between two entities. Don't use it on gateways or it will transform your gateway into a bridge. HINT: you can use the content filtering engine to drop packets that should not pass. This way ettercap will work as an inline IPS ;)

You can also perform man in the middle attacks while using the unified sniffing. You can choose the mitm attack that you prefer. The mitm attack module is independent from the sniffing and filtering process, so you can launch several attacks at the same time or use your own tool for the attack. The crucial point is that the packets have to arrive to ettercap with the correct mac address and a different ip address (only these packets will be forwarded).

The most relevant ettercap features are:

SSH1 support : you can sniff User and Pass, and even the data of an SSH1 connection. ettercap is the first software capable to sniff an SSH connection in FULL-DUPLEX

SSL support : you can sniff SSL secured data... a fake certificate is presented to the client and the session is decrypted.

Characters injection in an established connection : you can inject characters to the server (emulating commands) or to the client (emulating replies) maintaining the connection alive !!

Packet filtering/dropping: You can set up a filter script that searches for a particular string (even hex) in the TCP or UDP payload and replace it with yours or drop the entire packet. The filtering engine can match

any field of the network protocols and modify whatever you want (see etterfilter(8)).

Remote traffic sniffing through tunnels and route mangling: You can play with linux cooked interfaces or use the integrated plugin to sniff tunneled or route-mangled remote connections and perform mitm attacks on them.

Plug-ins support : You can create your own plugin using the ettercap's API.

Password collector for : TELNET, FTP, POP, RLOGIN, SSH1, ICQ, SMB, MySQL, HTTP, NNTP, X11, NAPSTER, IRC, RIP, BGP, SOCKS 5, IMAP 4, VNC, LDAP, NFS, SNMP, HALF LIFE, QUAKE 3, MSN, YMSG (other protocols coming soon...)

Passive OS fingerprint: you scan passively the lan (without sending any packet) and gather detailed info about the hosts in the LAN: Operating System, running services, open ports, IP, mac address and network adapter vendor.

Kill a connection: from the connections list you can kill all the connections you want

TARGET SPECIFICATION

There is no concept of SOURCE nor DEST. The two targets are intended to filter traffic coming from one to the other and vice-versa (since the connection is bidirectional).

TARGET is in the form *MAC/IPs/PORTs*.
NOTE: If IPv6 is enabled, *TARGET* is in the form *MAC/IPs/IPv6/PORTs*.

If you want you can omit any of its parts and this will represent an ANY in that part.
e.g.
"//80" means ANY mac address, ANY ip and ONLY port 80
"/10.0.0.1/" means ANY mac address, ONLY ip 10.0.0.1 and ANY port

MAC must be unique and in the form 00:11:22:33:44:55

IPs is a range of IP in dotted notation. You can specify range with the − (hyphen) and single ip with , (comma). You can also use ; (semicolon) to indicate different ip addresses.
e.g.
"10.0.0.1−5;10.0.1.33" expands into ip 10.0.0.1, 2, 3, 4, 5 and 10.0.1.33

PORTs is a range of PORTS. You can specify range with the − (hyphen) and single port with , (comma).
e.g.
"20−25,80,110" expands into ports 20, 21, 22, 23, 24, 25, 80 and 110

NOTE:
you can reverse the matching of the TARGET by adding the −R option to the command line. So if you want to sniff ALL the traffic BUT the one coming or going to 10.0.0.1 you can specify "./ettercap −R /10.0.0.1/"

NOTE:
TARGETs are also responsible of the initial scan of the lan. You can use them to restrict the scan to only a subset of the hosts in the netmask. The result of the merging between the two targets will be scanned. remember that not specifying a target means "no target", but specifying "//" means "all the hosts in the subnet".

PRIVILEGES DROPPING

ettercap needs root privileges to open the Link Layer sockets. After the initialization phase, the root privs are not needed anymore, so ettercap drops them to UID = 65535 (nobody). Since ettercap has to write (create) log files, it must be executed in a directory with the right permissions (e.g. /tmp/). If you want to drop privs to a different uid, you can export the environment variable EC_UID with the value of the uid you want to drop the privs to (e.g. export EC_UID=500) or set the correct parameter in the etter.conf file.

SSL MITM ATTACK

SSL mitm attack is dependent on TCP traffic redirection to a custom listener port of ettercap. The redir_command_on and redir_command_off configuration variables take care of this (see "etter.conf(5)").
However, when ettercap starts, traffic for any source and any destination targeted for the redirectable services will be redirected to ettercap and the SSL stream will be intercepted.
This may not be the desired behaviour. Therefore you can adjust the redirect rules after ettercap has been started using the selected user interface.

While performing the SSL mitm attack, ettercap substitutes the real ssl certificate with its own. The fake certificate is created on the fly and all the fields are filled according to the real cert presented by the server. Only the issuer is modified and signed with the private key contained in the 'etter.ssl.crt' file. If you want to use a different private key you have to regenerate this file. To regenerate the cert file use the following commands:

openssl genrsa −out etter.ssl.crt 1024
openssl req −new −key etter.ssl.crt −out tmp.csr
openssl x509 −req −days 1825 −in tmp.csr −signkey etter.ssl.crt −out tmp.new
cat tmp.new >> etter.ssl.crt
rm −f tmp.new tmp.csr

NOTE: SSL mitm is not available (for now) in bridged mode.

NOTE: You can use the −−certificate/−−private−key long options if you want to specify a different file rather than the etter.ssl.crt file.

OPTIONS

Options that make sense together can generally be combined. ettercap will warn the user about unsupported option combinations.

SNIFFING AND ATTACK OPTIONS

ettercap NG has a new unified sniffing method. This implies that ip_forwarding in the kernel is always disabled and the forwarding is done by ettercap. Every packet with destination mac address equal to the host's mac address and destination ip address different for the one bound to the iface will be forwarded by ettercap. Before forwarding them, ettercap can content filter, sniff, log or drop them. It does not matter how these packets are hijacked, ettercap will process them. You can even use external programs to hijack packet. You have full control of what ettercap should receive. You can use the internal mitm attacks, set the interface in promisc mode, use plugins or use every method you want.

IMPORTANT NOTE: if you run ettercap on a gateway, remember to re-enable the ip_forwarding after you have killed ettercap. Since ettercap drops its privileges, it cannot restore the ip_forwarding for you.

−M, **−−mitm <METHOD:ARGS>**

MITM attack
This option will activate the man in the middle attack. The mitm attack is totally independent from the sniffing. The aim of the attack is to hijack packets and redirect them to ettercap. The sniffing engine will forward them if necessary.
You can choose the mitm attack that you prefer and also combine some of them to perform different attacks at the same time.
If a mitm method requires some parameters you can specify them after the colon. (e.g. −M dhcp:ip_pool,netmask,etc)

The following mitm attacks are available:

arp ([remote],[oneway])

This method implements the ARP poisoning mitm attack. ARP requests/replies are sent to the victims to poison their ARP cache. Once the cache has been poisoned the victims will send all packets to the attacker which, in turn, can modify and forward them to the real destination.

In silent mode (−z option) only the first target is selected, if you want to poison multiple target in silent mode use the −j option to load a list from a file.

You can select empty targets and they will be expanded as 'ANY' (all the hosts in the LAN). The target list is joined with the hosts list (created by the arp scan) and the result is used to determine the victims of the attack.

The parameter "remote" is optional and you have to specify it if you want to sniff remote ip address poisoning a gateway. Indeed if you specify a victim and the gw in the TARGETS, ettercap will sniff only connection between them, but to enable ettercap to sniff connections that pass thru the gw, you have to use this parameter.

The parameter "oneway" will force ettercap to poison only from TARGET1 to TARGET2. Useful if you want to poison only the client and not the router (where an arp watcher can be in place).

Example:

the targets are: /10.0.0.1−5/ /10.0.0.15−20/
and the host list is: 10.0.0.1 10.0.0.3 10.0.0.16 10.0.0.18

the associations between the victims will be:
1 and 16, 1 and 18, 3 and 16, 3 and 18

if the targets overlap each other, the association with identical ip address will be skipped.

NOTE: if you manage to poison a client, you have to set correct routing table in the kernel specifying the GW. If your routing table is incorrect, the poisoned clients will not be able to navigate the Internet.

icmp (MAC/IP)

This attack implements ICMP redirection. It sends a spoofed icmp redirect message to the hosts in the lan pretending to be a better route for internet. All connections to internet will be redirected to the attacker which, in turn, will forward them to the real gateway. The resulting attack is a HALF-DUPLEX mitm. Only the client is redirected, since the gateway will not accept redirect messages for a directly connected network. BE SURE TO NOT USE FILTERS THAT MODIFY THE PAYLOAD LENGTH. you can use a filter to modify packets, but the length must be the same since the tcp sequences cannot be updated in both ways.
You have to pass as argument the MAC and the IP address of the real gateway for the lan. Obviously you have to be able to sniff all the traffic. If you are on a switch you have to use a different mitm attack such as arp poisoning.

NOTE: to restrict the redirection to a given target, specify it as a TARGET

Example:

−M icmp:00:11:22:33:44:55/10.0.0.1

will redirect all the connections that pass thru that gateway.

dhcp (ip_pool/netmask/dns)

This attack implements DHCP spoofing. It pretends to be a DHCP server and tries to win the race condition with the real one to force the client to accept the attacker's reply. This way ettercap is able to manipulate the GW parameter and hijack all the outgoing traffic generated by the clients.
The resulting attack is a HALF-DUPLEX mitm. So be sure to use appropriate filters (see above in the ICMP section).

You have to pass the ip pool to be used, the netmask and the ip of the dns server. Since ettercap tries to win the race with the real server, it DOES NOT CHECK if the ip is already assigned. You have to specify an ip pool of FREE addresses to be used. The ip pool has the same form of the target specification.

If the client sends a dhcp request (suggesting an ip address) ettercap will ack on that ip and modify only the gw option. If the client makes a dhcp discovery, ettercap will use the first unused ip address of the list you have specified on command line. Every discovery consumes an ip address. When the list is over, ettercap stops offering new ip addresses and will reply only to dhcp requests.
If you don't want to offer any ip address, but only change the router information of dhcp request/ack, you can specify an empty ip_pool.

BIG WARNING: if you specify a list of ip that are in use, you will mess your network! In general, use this attack carefully. It can really mess things up! When you stop the attack, all the victims will be still convinced that ettercap is the gateway until the lease expires...

Example:

−M dhcp:192.168.0.30,35,50−60/255.255.255.0/192.168.0.1
reply to DHCP offer and request.

−M dhcp:/255.255.255.0/192.168.0.1
reply only to DHCP request.

port ([remote],[tree])

This attack implements Port Stealing. This technique is useful to sniff in a switched environment when ARP poisoning is not effective (for example where static mapped ARPs are used).

It floods the LAN (based on port_steal_delay option in etter.conf) with ARP packets. If you don't specify the "tree" option, the destination MAC address of each "stealing" packet is the same as the attacker's one (other NICs won't see these packets), the source MAC address will be one of the MACs in the host list. This process "steals" the switch port of each victim host in the host list. Using low delays, packets destined to "stolen" MAC addresses will be received by the attacker, winning the race condition with the real port owner. When the attacker receives packets for "stolen" hosts, it stops the flooding process and performs an ARP request for the real destination of the packet. When it receives the ARP reply it's sure that the victim has "taken back" his port, so ettercap can resend the packet to the destination as is. Now we can re-start the flooding process waiting for new packets.

If you use the "tree" option, the destination MAC address of each stealing packet will be a bogus one, so these packets will be propagated to other switches (not only the directly connected one). This way you will be able to steal ports on other switches in the tree (if any), but you will generate a huge amount of traffic (according to port_steal_delay). The "remote" option has the same meaning as in "arp" mitm method.

When you stop the attack, ettercap will send an ARP request to each stolen host giving back their switch ports.
You can perform either HALF or FULL DUPLEX mitm according to target selection.

NOTE: Use this mitm method only on ethernet switches. Use it carefully, it could produce performances loss or general havoc.

NOTE: You can NOT use this method in only-mitm mode (−o flag), because it hooks the sniffing engine, and you can't use interactive data injection.

NOTE: It could be dangerous to use it in conjunction with other mitm methods.

NOTE: This mitm method doesn't work on Solaris and Windows because of the lipcap

and libnet design and the lack of certain ioctl(). (We will feature this method on these OSes if someone will request it...)

Example:

The targets are: /10.0.0.1/ /10.0.0.15/
You will intercept and visualize traffic between 10.0.0.1 and 10.0.0.15, but you will receive all the traffic for 10.0.0.1 and 10.0.0.15 too.

The target is: /10.0.0.1/
You will intercept and visualize all the traffic for 10.0.0.1.

ndp ([remote],[oneway])

NOTE: This MITM method is only supported if IPv6 support has been enabled.

This method implements the NDP poisoning attack which is used for MITM of IPv6 connections. ND requests/replies are sent to the victims to poison their neighbor cache. Once the cache has been poisoned the victims will send all IPv6 packets to the attacker which, in turn, can modify and forward them to the real destination.

In silent mode (−z option) only the first target is selected, if you want to poison multiple target in silent mode use the −j option to load a list from a file.

You can select empty targets and they will be expanded as 'ANY' (all the hosts in the LAN). The target list is joined with the hosts list (created by the arp scan) and the result is used to determine the victims of the attack.

The parameter "remote" is optional and you have to specify it if you want to sniff remote ip address poisoning a gateway. Indeed if you specify a victim and the gw in the TARGETS, ettercap will sniff only connection between them, but to enable ettercap to sniff connections that pass thru the gw, you have to use this parameter.

The parameter "oneway" will force ettercap to poison only from TARGET1 to TARGET2. Useful if you want to poison only the client and not the router (where an arp watcher can be in place).

Example:

Targets are: //fe80::260d:afff:fe6e:f378/ //2001:db8::2:1/
Ranges of IPv6 addresses are not yet supported.

NOTE: if you manage to poison a client, you have to set correct routing table in the kernel specifying the GW. If your routing table is incorrect, the poisoned clients will not be able to navigate the Internet.

NOTE: in IPv6 usually the link-local address of the router is being used as the gateway address. Therefore you need to set the link-local address of the router as one target and the global-unicast address of the victim as the other in order to set up a successful IPv6 MITM attack using NDP poisoning.

−o, **−−only−mitm**

This options disables the sniffing thread and enables only the mitm attack. Useful if you want to use ettercap to perform mitm attacks and another sniffer (such as wireshark) to sniff the traffic. Keep in mind that the packets are not forwarded by ettercap. The kernel will be responsible for the forwarding. Remember to activate the "ip forwarding" feature in your kernel.

−f, **−−pcapfilter <FILTER>**

Set a capturing filter in the pcap library. The format is the same as tcpdump(1). Remember that this kind of filter will not sniff packets out of the wire, so if you want to perform a mitm attack, ettercap will not be able to forward hijacked packets.
These filters are useful to decrease the network load impact into ettercap decoding module.

−B, **−−bridge <IFACE>**

BRIDGED sniffing
You need two network interfaces. ettercap will forward form one to the other all the traffic it sees. It is useful for man in the middle at the physical layer. It is totally stealthy since it is passive and there is no way for an user to see the attacker.
You can content filter all the traffic as you were a transparent proxy for the "cable".

OFF LINE SNIFFING

−r, **−−read <FILE>**

OFF LINE sniffing
With this option enabled, ettercap will sniff packets from a pcap compatible file instead of capturing from the wire.
This is useful if you have a file dumped from tcpdump or wireshark and you want to make an analysis (search for passwords or passive fingerprint) on it.
Obviously you cannot use "active" sniffing (arp poisoning or bridging) while sniffing from a file.

−w, **−−write <FILE>**

WRITE packet to a pcap file
This is useful if you have to use "active" sniffing (arp poison) on a switched LAN but you want to analyze the packets with tcpdump or wireshark. You can use this option to dump the packets to a file and then load it into your favourite application.

NOTE: dump file collect ALL the packets disregarding the TARGET. This is done because you may want to log even protocols not supported by ettercap, so you can analyze them with other tools.

TIP: you can use the −w option in conjunction with the −r one. This way you will be able to filter the payload of the dumped packets or decrypt WEP-encrypted WiFi traffic and dump them to another file.

USER INTERFACES OPTIONS

−T, **−−text**

The text only interface, only printf ;)
It is quite interactive, press 'h' in every moment to get help on what you can do.

−q, **−−quiet**

Quiet mode. It can be used only in conjunction with the console interface. It does not print packet content. It is useful if you want to convert pcap file to ettercap log files.

example:

ettercap −Tq −L dumpfile −r pcapfile

−s, **−−script <COMMANDS>**

With this option you can feed ettercap with command as they were typed on the keyboard by the user. This way you can use ettercap within your favourite scripts. There is a special command you can issue thru this command: s(x). this command will sleep for x seconds.

example:

ettercap −T −s 'lq' will print the list of the hosts and exit
ettercap −T −s 's(300)olqq' will collect the infos for 5 minutes, print the list of the local profiles and exit

−C, **−−curses**
Ncurses based GUI. See ettercap_curses(8) for a full description.

−G, **−−gtk**
The nice GTK2 interface (thanks Daten...).

−D, **−−daemonize**
Daemonize ettercap. This option will detach ettercap from the current controlling terminal and set it as a daemon. You can combine this feature with the "log" option to log all the traffic in the background. If the daemon fails for any reason, it will create the file "./ettercap_daemonized.log" in which the error caught by ettercap will be reported. Furthermore, if you want to have a complete debug of the daemon process, you are encouraged to recompile ettercap in debug mode.

GENERAL OPTIONS

−b, **−−broadcast**
Tells Ettercap to process packets coming from Broadcast address.

−i, **−−iface <IFACE>**
Use this <IFACE> instead of the default one. The interface can be unconfigured (requires libnet >= 1.1.2), but in this case you cannot use MITM attacks and you should set the unoffensive flag.

−I, **−−iflist**
This option will print the list of all available network interfaces that can be used within ettercap. The option is particularly useful under windows where the name of the interface is not so obvious as under *nix.

−Y, **−−secondary <interface list>**
Specify a list of (or single) secondary interfaces to capture packets from.

−A, **−−address <ADDRESS>**
Use this <ADDRESS> instead of the one autodetected for the current iface. This option is useful if you have an interface with multiple ip addresses.

−n, **−−netmask <NETMASK>**
Use this <NETMASK> instead of the one associated with the current iface. This option is useful if you have the NIC with an associated netmask of class B and you want to scan (with the arp scan) only a class C.

−R, **−−reversed**
Reverse the matching in the TARGET selection. It means not(TARGET). All but the selected TARGET.

−t, **−−proto <PROTO>**
Sniff only PROTO packets (default is TCP + UDP).
This is useful if you want to select a port via the TARGET specification but you want to differentiate between tcp or udp.
PROTO can be "tcp", "udp" or "all" for both.

−6, **−−ip6scan**
Send ICMPv6 probes to discover active IPv6 nodes on the link. This options sends a ping request to the all-nodes address to motivate active IPv6 hosts to respond. You should not use this option if you try to hide yourself. Therefore this option is optional.

NOTE: This option is only available if IPv6 support has been enabled.

−z, **−−silent**
Do not perform the initial ARP scan of the LAN.

NOTE: you will not have the hosts list, so you can't use the multipoison feature. you can only select two hosts for an ARP poisoning attack, specifying them through the TARGETs

−p, **−−nopromisc**
Usually, ettercap will put the interface in promisc mode to sniff all the traffic on the wire. If you want to sniff only your connections, use this flag to NOT enable the promisc mode.

−S, **−−nosslmitm**
Usually, ettercap forges SSL certificates in order to intercept https traffic. This option disables that behavior.

−u, **−−unoffensive**
Every time ettercap starts, it disables ip forwarding in the kernel and begins to forward packets itself. This option prevent to do that, so the responsibility of ip forwarding is left to the kernel.
This options is useful if you want to run multiple ettercap instances. You will have one instance (the one without the −u option) forwarding the packets, and all the other instances doing their work without forwarding them. Otherwise you will get packet duplicates.
It also disables the internal creation of the sessions for each connection. It increases performances, but you will not be able to modify packets on the fly.
If you want to use a mitm attack you have to use a separate instance.
You have to use this option if the interface is unconfigured (without an ip address.)
This is also useful if you want to run ettercap on the gateway. It will not disable the forwarding and the gateway will correctly route the packets.

−j, **−−load−hosts <FILENAME>**
It can be used to load a hosts list from a file created by the −k option. (see below)

−k, **−−save−hosts <FILENAME>**
Saves the hosts list to a file. Useful when you have many hosts and you don't want to do an ARP storm at startup any time you use ettercap. Simply use this options and dump the list to a file, then to load the information from it use the −j <filename> option.

−P, **−−plugin <PLUGIN>**

Run the selected PLUGIN. Many plugins need target specification, use TARGET as always. Use multiple occurrences of this parameter to select multiple plugins.
In console mode (−C option), standalone plugins are executed and then the application exits. Hook plugins are activated and the normal sniffing is performed.
To have a list of the available external plugins use "list" (without quotes) as plugin name (e.g. ./ettercap −P list).

NOTE: you can also activate plugins directly from the interfaces (always press "h" to get the inline help)

More detailed info about plugins and about how to write your own are found in the man page ettercap_plugins(8)

−F, **−−filter <FILE>**

Load the filter from the file <FILE>. The filter must be compiled with etterfilter(8). The utility will compile the filter script and produce an ettercap−compliant binary filter file. Read the etterfilter(8) man page for the list of functions you can use inside a filter script. Any number of filters can be loaded by specifying the option multiple times; packets are passed through each filter in the order specified on the command line. You can also load a script without enabling it by appending :0 to the filename.
NOTE: these filters are different from those set with −−pcapfilter. An ettercap filter is a content filter and can modify the payload of a packet before forwarding it. Pcap filter are used to capture only certain packets.
NOTE: you can use filters on pcapfile to modify them and save to another file, but in this case you have to pay attention on what you are doing, since ettercap will not recalculate checksums, nor split packets exceeding the mtu (snaplen) nor anything like that.

−W, **−−wifi−key <KEY>**

You can specify a key to decrypt WiFi packets (WEP or WPA). Only the packets decrypted successfully will be passed to the decoders stack, the others will be skipped with a message.
The parameter has the following syntax: type:bits:t:string. Where 'type' can be: wep, wpa-pws or wpa-psk, 'bits' is the bit length of the key (64, 128 or 256), 't' is the type of the string ('s' for string and 'p' for passphrase). 'string' can be a string or an escaped hex sequences.

example:
−−wifi−key wep:128:p:secret
−−wifi−key wep:128:s:ettercapwep0
−−wifi−key 'wep:64:s:\x01\x02\x03\x04\x05'
−−wifi−key wpa:pwd:ettercapwpa:ssid
−−wifi−key wpa:psk:
663eb260e87cf389c6bd7331b28d82f5203b0cae4e315f9cbb7602f3236708a6

−a, **−−config <CONFIG>**

Loads an alternative config file instead of the default in /etc/etter.conf. This is useful if you have many preconfigured files for different situations.

−−certificate <FILE>

Tells Ettercap to use the specified certificate file for the SSL MiTM attack.

−−private−key <FILE>

Tells Ettercap to use the specified private key file for the SSL MiTM attack.

VISUALIZATION OPTIONS

−e, **−−regex <REGEX>**

Handle only packets that match the regex.
This option is useful in conjunction with −L. It logs only packets that match the posix regex REGEX.
It impacts even the visualization of the sniffed packets. If it is set only packets matching the regex will be displayed.

−V, **−−visual <FORMAT>**

Use this option to set the visualization method for the packets to be displayed.

FORMAT may be one of the following:

hex Print the packets in hex format.

example:

the string "HTTP/1.1 304 Not Modified" becomes:

```
0000: 4854 5450 2f31 2e31 2033 3034 204e 6f74  HTTP/1.1 304 Not
0010: 204d 6f64 6966 6965 64                   Modified
```

ascii Print only "printable" characters, the others are displayed as dots '.'

text Print only the "printable" characters and skip the others.

ebcdic Convert an EBCDIC text to ASCII.

html Strip all the html tags from the text. A tag is every string between < and >.

example:

<title>This is the title</title>, but the following <string> will not be displayed.

This is the title, but the following will not be displayed.

utf8 Print the packets in UTF-8 format. The encoding used while performing the conversion is declared in the etter.conf(5) file.

−d, **−−dns**

Resolve ip addresses into hostnames.

NOTE: this may seriously slow down ettercap while logging passive information. Every time a new host is found, a query to the dns is performed. Ettercap keeps a cache for already resolved host to increase the speed, but new hosts need a new query and the dns may take up to 2 or 3 seconds to respond for an unknown host.

HINT: ettercap collects the dns replies it sniffs in the resolution table, so even if you specify to not resolve the hostnames, some of them will be resolved because the reply was previously sniffed. think about it as a passive dns resolution for free... ;)

−E, **−−ext−headers**
Print extended headers for every displayed packet. (e.g. mac addresses)

−Q, **−−superquiet**
Super quiet mode. Do not print users and passwords as they are collected. Only store them in the profiles. It can be useful to run ettercap in text only mode but you don't want to be flooded with dissectors messages. Useful when using plugins because the sniffing process is always active, it will print all the collected infos, with this option you can suppress these messages.
NOTE: this options automatically sets the −q option.

example:

ettercap −TzQP finger /192.168.0.1/22

LOGGING OPTIONS

−L, **−−log <LOGFILE>**
Log all the packets to binary files. These files can be parsed by etterlog(8) to extract human readable data. With this option, all packets sniffed by ettercap will be logged, together with all the passive info (host info + user & pass) it can collect. Given a LOGFILE, ettercap will create LOGFILE.ecp (for packets) and LOGFILE.eci (for the infos).

NOTE: if you specify this option on command line you don't have to take care of privileges since the log file is opened in the startup phase (with high privs). But if you enable the log option while ettercap is already started, you have to be in a directory where uid = 65535 or uid = EC_UID can write.

NOTE: the logfiles can be compressed with the deflate algorithm using the −c option.

−l, **−−log−info <LOGFILE>**
Very similar to −L but it logs only passive information + users and passwords for each host. The file will be named LOGFILE.eci

−m, **−−log−msg <LOGFILE>**
It stores in <LOGFILE> all the user messages printed by ettercap. This can be useful when you are using ettercap in daemon mode or if you want to track down all the messages. Indeed, some dissectors print messages but their information is not stored anywhere, so this is the only way to keep track of them.

−c, **−−compress**
Compress the logfile with the gzip algorithm while it is dumped. etterlog(8) is capable of handling both compressed and uncompressed log files.

−o, **−−only−local**
Stores profiles information belonging only to the LAN hosts.

NOTE: this option is effective only against the profiles collected in memory. While logging to a file ALL the hosts are logged. If you want to split them, use the related etterlog(8) option.

−O, **−−only−remote**
Stores profiles information belonging only to remote hosts.

STANDARD OPTIONS

−v, **−−version**
Print the version and exit.

−h, **−−help**
prints the help screen with a short summary of the available options.

EXAMPLES

Here are some examples of using ettercap.

ettercap −Tp

Use the console interface and do not put the interface in promisc mode. You will see only your traffic.

ettercap −Tzq

Use the console interface, do not ARP scan the net and be quiet. The packet content will not be displayed, but user and passwords, as well as other messages, will be displayed.

ettercap −T −j /tmp/victims −M arp /10.0.0.1−7/ /10.0.0.10−20/

Will load the hosts list from /tmp/victims and perform an ARP poisoning attack against the two target. The list will be joined with the target and the resulting list is used for ARP poisoning.

ettercap −T −M arp // //

Perform the ARP poisoning attack against all the hosts in the LAN. BE CAREFUL !!

ettercap −T −M arp:remote /192.168.1.1/ /192.168.1.2−10/

Perform the ARP poisoning against the gateway and the host in the lan between 2 and 10. The 'remote' option is needed to be able to sniff the remote traffic the hosts make through the gateway.

ettercap −Tzq //110

Sniff only the pop3 protocol from every hosts.

ettercap −Tzq /10.0.0.1/21,22,23

Sniff telnet, ftp and ssh connections to 10.0.0.1.

ettercap −P list

Prints the list of all available plugins

FILES

~/.config/ettercap_gtk

Stores persistent information (e.g., window placement) between sessions.

ORIGINAL AUTHORS

Alberto Ornaghi (ALoR) <alor@users.sf.net>
Marco Valleri (NaGA) <naga@antifork.org>

PROJECT STEWARDS

Emilio Escobar (exfil) <eescobar@gmail.com>
Eric Milam (Brav0Hax) <jbrav.hax@gmail.com>

OFFICIAL DEVELOPERS

Mike Ryan (justfalter) <falter@gmail.com>
Gianfranco Costamagna (LocutusOfBorg) <costamagnagianfranco@yahoo.it>
Antonio Collarino (sniper) <anto.collarino@gmail.com>
Ryan Linn <sussuro@happypacket.net>
Jacob Baines <baines.jacob@gmail.com>

CONTRIBUTORS

Dhiru Kholia (kholia) <dhiru@openwall.com>
Alexander Koeppe (koeppea) <format_c@online.de>
Martin Bos (PureHate) <purehate@backtrack.com>
Enrique Sanchez
Gisle Vanem <giva@bgnett.no>
Johannes Bauer <JohannesBauer@gmx.de>
Daten (Bryan Schneiders) <daten@dnetc.org>

SEE ALSO

etter.conf(5) ettercap_curses(8) ettercap_plugins(8) etterlog(8) etterfilter(8) ettercap−pkexec(8)

AVAILABILITY

https://github.com/Ettercap/ettercap/downloads

GIT

git clone git://github.com/Ettercap/ettercap.git
or
git clone https://github.com/Ettercap/ettercap.git

BUGS

Our software never has bugs.
It just develops random features. ;)

KNOWN-BUGS

- ettercap doesn't handle fragmented packets... only the first segment will be displayed by the sniffer. However all the fragments are correctly forwarded.

+ please send bug-report, patches or suggestions to <ettercap−betatesting@lists.sourceforge.net> or visit https://github.com/Ettercap/ettercap/issues.

+ to report a bug, follow the instructions in the README.BUGS file

PHILOLOGICAL HISTORY

"Even if blessed with a feeble intelligence, they are cruel and smart..." this is the description of Ettercap, a monster of the RPG Advanced Dungeons & Dragon.

The name "ettercap" was chosen because it has an assonance with "ethercap" which means "ethernet capture" (what ettercap actually does) and also because such monsters have a powerful poison... and you know, arp poisoning... ;)

The Lord Of The (Token)Ring

(the fellowship of the packet)

"One Ring to link them all, One Ring to ping them,
one Ring to bring them all and in the darkness sniff them."

Last words

"Programming today is a race between software engineers striving to build bigger and better idiot-proof programs, and the Universe trying to produce bigger and better idiots. So far, the Universe is winning." - Rich Cook