NAME

etter.conf - Ettercap configuration file

DESCRIPTION

etter.conf is the configuration file that determines ettercap behaviour. It is always loaded at startup and it configures some attributes used at runtime.

The file contains entries of the form:

[section]
entry = value
...

Each entry defines a variable that can be customized. Every value MUST be an integer. Sections are used only to group together some variables.

NOTE: if you omit a variable in the conf file, it will be initialized with the value 0. It is strongly discouraged to not initialize critical variables such as "arp_poison_delay" or "connection_timeout".

The following is a list of available variables:

[privs]

ec_uid This variable specifies the UID to which privileges are dropped at startup. After the socket at link layer has been opened the privileges are dropped to a specific uid different from root for security reasons. etter.conf is the only file that is read with root privs. Be sure that the specified uid has enough privs to read other files (etter.*) You can bypass this variable by setting the environment variable EC_UID.

[mitm]

arp_storm_delay The value represents the milliseconds to wait between two consecutive packets during the initial ARP scan. You can increment this value to be less aggressive at startup. The randomized scan plus a high delay can fool some types of ARP scan detectors.

arp_poison_smart With this variable set, only 3 inital poisoned ARP messages are sent to the victims. This poisoned status is kept up by ettercap with responding to ARP requests from victims that want to refresh their ARP cache. This makes the ARP poisoning very stealthy but may be unreliable on shared media such as WiFi.

arp_poison_warm_up When the poisoning process starts, the inter-packet delay is low for the first 5 poisons (to be sure the poisoning process has been successful). After the first 5 poisons, the delay is incremented (to keep up the poisoning). This variable controls the delay for the first 5 poisons. The value is in seconds.
The same delay is used when the victims are restored to the original associations (RE-ARPing) when ettercap is closed.

arp_poison_delay This variable controls the poisoning delay after the first 5 poisons. The value is expressed in seconds. You can increase this value (to try to fool the IDS) up to the timeout of the ARP cache (which depends on the poisoned operating system).

arp_poison_icmp Enable the sending of a spoofed ICMP message to force the targets to make an arp request. This will create an arp entry in the host cache, so ettercap will be able to win the race condition and poison the target. Useful against targets that do not accept gratuitous arp if the entry is not in the cache.

arp_poison_reply Use ARP replies to poison the targets. This is the classic attack.

arp_poison_request Use ARP request to poison the targets. Useful against targets that cache even arp request values.

arp_poison_equal_mac
Set this option to 0 if you want to skip the poisoning of two hosts with the same mac address. This may happen if a NIC has one or more aliases on the same network.

dhcp_lease_time This is the lease time (in seconds) for a dhcp assignment. You can lower this value to permit the victims to receive a correct dhcp reply after you have stopped your attack. Using higher timeouts can seriously mess up your network after the attack has finished. On the other hand some clients will prefer a higher lease time, so you have to increase it to win the race condition against the real server.

port_steal_delay This is the delay time (in milliseconds) between stealing packets for the "port" mitm method. With low delays you will be able to intercept more packets, but you will generate more traffic. You have to tune this value in order to find a good balance between the number of intercepted packets, re-transmitted packets and lost packets. This value depends on full/half duplex channels, network drivers and adapters, network general configuration and hardware.

port_steal_send_delay This is the delay time (in microseconds) between packets when the "port" mitm method has to re-send packets queues. As said for port_steal_delay you have to tune this option to the lowest acceptable value.

ndp_poison_warm_up
This option operates similar to the arp_poison_warm_up option. When the poisoning process starts, this option controls the NDP poison delay for the first 5 poisons (to be sure the poisoning process has been successful). After the first 5 poisons, the delay is incremented (to keep up the poisoning). This variable controls the delay for the first 5 poisons. The value should be lower than the **ndp_poison_delay**. The value is in seconds.
The same delay is used when the victims are restored to the original associations when ettercap is closed.

ndp_poison_delay This option is similar to the arp_poison_delay option. It controls the delay in seconds for sending out the poisoned NDP packets to poison victim's neighbor cache. This value may be increased to hide from IDSs. But increasing the value increases as well the probability for failing race conditions during neighbor discovery and to miss some packets.

ndp_poison_send_delay
This option controls the delay in microseconds between poisoned NDP packets are sent. This value may be increased to hide from IDSs. But increasing the value increases as well the probability for failing race conditions during neighbor discovery and to miss some packets.

ndp_poison_icmp Enable the sending of a spoofed ICMPv6 message to motivate the targets to perform neighbor discovery. This will create an entry in the host neighbor cache, so ettercap will be able to win the race condition and poison the target. Useful against targets that do not accept neighbor advertisements if the entry is not in the cache.

ndp_poison_equal_mac
Set this option to 0 if you want to skip the NDP poisoning of two hosts with the same mac address. This may happen if a NIC has one or more aliases on the same network.

icmp6_probe_delay This option defines the time in seconds ettercap waits for active IPv6 nodes to respond to the ICMP probes. Decreasing this value could lead to miss replies from active IPv6 nodes, hence miss them in the host list. Increasing the value usually has no impact; normally nodes can manage to answer during the default delay.

NOTE: The ndp and icmp6 options are only available if ettercap has been built with IPv6 support

[connections]

connection_timeout Every time a new connection is discovered, ettercap allocates the needed structures. After a customizable timeout, you can free these structures to keep the memory usage low. This variable represents this timeout. The value is expressed in seconds. This timeout is applied even to the session tracking system (the protocol state machine for dissectors).

connection_idle The number of seconds to wait before a connection is marked as IDLE.

connection_buffer This variable controls the size of the buffer linked to each connection. Every sniffed packet is added to the buffer and when the buffer is full the older packets are deleted to make room for newer ones. This buffer is useful to view data that went on the cable before you select and view a specific connection. The higher this value, the higher the ettercap memory occupation. By the way, the buffer is dynamic, so if you set a buffer of 100.000 byte it is not allocated all together at the first packet of a connection, but it is filled as packets arrive.

connect_timeout The timeout in seconds when using the connect() syscall. Increase it if you get a "Connection timeout" error. This option has nothing to do with connections sniffed by ettercap. It is a timeout for the connections made by ettercap to other hosts (for example when fingerprinting remote host).

[stats]

sampling_rate Ettercap keeps some statistics on the processing time of the bottom half (the sniffer) and top half (the protocol decoder). These statistics are made on the average processing time of sampling_rate packets. You can decrease this value to have a more accurate real-time picture of processing time or increase it to have a smoother picture. The total average will not change, but the worst value will be heavily influenced by this value.

[misc]

close_on_eof When reading from a dump file and using console or daemon UI, this variable is used to determine what action has to be done on EOF. It is a boolean value. If set to 1 ettercap will close itself (useful in scripts). Otherwise the session will continue waiting for user input.

store_profiles Ettercap collects in memory a profile for each host it detects. Users and passwords are collected there. If you want to run ettercap in background logging all the traffic, you may want to disable the collecting in memory to save system memory. Set this option to 0 (zero) to disable profiles collection. A value of 1 will enable collection for all the hosts, 2 will collect only local hosts and 3 only remote hosts (a host is considered remote if it does not belong to the netmask).

aggressive_dissectors Some dissectors (such as SSH and HTTPS) need to modify the payload of the packets in order to collect passwords and perform a decryption attack. If you want to disable the "dangerous" dissectors all together, set this value to 0.

skip_forwarded If you set this value to 0 you will sniff even packets forwarded by ettercap or by the kernel. It will generate duplicate packets in conjunction with the arp mitm method (for example). It could be useful while running ettercap in unoffensive mode on a host with more than one network interface (waiting for the multiple-interface feature...)

checksum_warning If you set the value to 0 the messages about incorrect checksums will not be displayed in the user messages windows (nor logged to a file with −m).
Note that this option will not disable the check on the packets, but only prevent the message to be displayed (see below).

checksum_check This option is used to completely disable the check on the checksum of the packets that ettercap receives. The check on the packets is performed to avoid ettercap spotting thru bad checsum packets (see Phrack 60.12). If you disable the check, you will be able to sniff even bad checksummed packet, but you will be spotted if someone is searching for you...

[dissectors]

protocol_name This value represents the port on which the protocol dissector has to be bound. A value of 0 will disable the dissector. The name of the variable is the same of the protocol name. You can specify a non standard port for each dissector as well as multiple ports. The syntax for multiport selection is the following: port1,port2,port3,...
NOTE: some dissectors are conditionally compiled . This means that depending on

the libraries found in your system some dissectors will be enabled and some others will not. By default etter.conf contains all supported dissectors. if you got a "FATAL: Dissector "xxx" does not exists (etter.conf line yy)" error, you have to comment out the yy line in etter.conf.

[curses]

color You can customize the colors of the curses GUI.
Simply set a field to one of the following values and look at the GUI aspect :)
Here is a list of values: 0 Black, 1 Red, 2 Green, 3 Yellow, 4 Blue, 5 Magenta, 6 Cyan, 7 White

[strings]

utf8_encoding specifies the encoding to be used while displaying the packets in UTF-8 format. Use the ‘iconv −−list‘ command for a list of supported encodings.

remote_broswer This command is executed by the remote_browser plugin each time it catches a good URL request into an HTTP connection. The command should be able to get 2 parameters:

%host the Host: tag in the HTTP header. Used to create the full request into the browser.

%url The page requested inside the GET request.

redir_command_on You must provide a valid command (or script) to enable tcp redirection at the kernel level in order to be able to use SSL dissection. Your script should be able to get 3 parameters:

%iface The network interface on which the rule must be set

%port The source port of the packets to be redirected (443 for HTTPS, 993 for imaps, etc).

%rport
The internally bound port to which ettercap listens for connections.

NOTE: this script is executed with an execve(), so you cannot use pipes or output redirection as if you were in a shell. We suggest you to make a script if you need those commands.

redir_command_off This script is used to remove the redirect rules applied by ’redir_command_on’. You should note that this script is called atexit() and thus it has not high privileges. You should provide a setuid program or set ec_uid to 0 in order to be sure that the script is executed successfully.

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

ettercap(8) ettercap_curses(8) ettercap_plugins(8) etterlog(8) etterfilter(8) ettercap−pkexec(8)