

NAME

ettercap-plugins NG-0.7.3 – A collection of plugins for ettercap

DESCRIPTION

Ettercap(8) supports loadable modules at runtime. They are called plugins and they come within the source tarball. They are automatically compiled if your system supports them or until you specify the `--disable-plugins` option to the configure script.

Some of older ettercap plugins (roper, banshee, and so on) have not been ported in the new version. By the way, you can achieve the same results by using new filtering engine.

If you use interactive mode, most plugins need to "Start Sniff" before using them.

To have a list of plugins installed in your system do that command:

```
ettercap -P list
```

The following is a list of available plugins:

arp_cop

It reports suspicious ARP activity by passively monitoring ARP requests/replies. It can report ARP poisoning attempts, or simple IP-conflicts or IP-changes. If you build the initial host list the plugin will run more accurately.

example :

```
ettercap -TQP arp_cop //
```

autoadd

It will automatically add new victims to the ARP poisoning mitm attack when they come up. It looks for ARP requests on the lan and when detected it will add the host to the victims list if it was specified in the TARGET. The host is added when an arp request is seen from it, since communicating hosts are alive :)

chk_poison

It performs a check to see if the arp poisoning module of ettercap was successful. It sends spoofed ICMP echo packets to all the victims of the poisoning pretending to be each of the other targets. If we can catch an ICMP reply with our MAC address as destination it means that the poisoning between those two targets is successful. It checks both ways of each communication. This plugin makes sense only where poisoning makes sense. The test fails if you specify only one target in silent mode. You can't run this plugin from command line because the poisoning process is not started yet. You have to launch it from the proper menu.

dns_spoof

This plugin intercepts DNS query and reply with a spoofed answer. You can chose to which address the plugin has to reply by modifying the etter.dns file. The plugin intercepts A, PTR and MX request. If it was an A request, the name is searched in the file and the ip address is returned (you can use wildcards in the name). If it was a PTR request, the ip is searched in the file and the name is returned (except for those name containing a wildcard). In case of MX request a special reply is crafted. The host is resolved with a fake host 'mail.host' and the additional record contains

the ip address of 'mail.host'. The first address or name that matches is returned, so be careful with the order.

dos_attack

This plugin runs a d.o.s. attack against a victim IP address. It first "scans" the victim to find open ports, then starts to flood these ports with SYN packets, using a "phantom" address as source IP. Then it uses fake ARP replies to intercept packets for the phantom host. When it receives SYN-ACK from the victim, it replies with an ACK packet creating an ESTABLISHED connection. You have to use a free IP address in your subnet to create the "phantom" host (you can use `find_ip` for this purpose). You can't run this plugin in unoffensive mode.

This plugin is based on the original Naptha DoS attack (http://razor.bindview.com/publish/advisories/adv_NAPTHA.html)

example :

```
ettercap -TQP dos_attack
```

dummy

Only a template to demonstrate how to write a plugin.

find_conn

Very simple plugin that listens for ARP requests to show you all the targets an host wants to talk to. It can also help you finding addresses in an unknown LAN.

example :

```
ettercap -TQzP find_conn
```

```
ettercap -TQu -i eth0 -P find_conn
```

find_ettercap

Try to identify ettercap packets sent on the LAN. It could be useful to detect if someone is using ettercap. Do not rely on it 100% since the tests are only on particular sequence/identification numbers.

find_ip

Find the first unused IP address in the range specified by the user in the target list. Some other plugins (such as `gre_relay`) need an unused IP address of the LAN to create a "fake" host. It can also be useful to obtain an IP address in an unknown LAN where there is no dhcp server. You can use `find_conn` to determine the IP addressing of the LAN, and then `find_ip`. You have to build host list to use this plugin so you can't use it in unoffensive mode. If you don't have an IP address for your interface, give it a bogus one (e.g. if the LAN is 192.168.0.0/24, use 10.0.0.1 to avoid conflicting IP), then launch this plugin specifying the subnet range. You can run it either from the command line or from the proper menu.

example :

```
ettercap -TQP find_ip //
```

```
ettercap -TQP find_ip /192.168.0.1-254/
```

fi nger

Uses the passive fi ngerprint capabilities to fi ngerprint a remote host. It does a connect() to the remote host to force the kernel to reply to the SYN with a SYN+ACK packet. The reply will be collected and the fi ngerprint is displayed. The connect() obey to the connect_timeout parameter in etter.conf(5). You can specify a target on command-line or let the plugin ask the target host to be fi ngerprinted. You can also specify multiple target with the usual multi-target specification (see ettercap(8)). if you specify multiple ports, all the ports will be tested on all the IPs.

example :

```
ettercap -TzP fi nger /192.168.0.1/22
ettercap -TzP fi nger /192.168.0.1-50/22,23,25
```

fi nger_submit

Use this plugin to submit a fi ngerprint to the ettercap website. If you found an unknown fi ngerprint, but you know for sure the operating system of the target, you can submit it so it will be inserted in the database in the next ettercap release. We need your help to increase the passive fi ngerprint database. Thank you very much.

example :

```
ettercap -TzP fi nger_submit
```

gre_relay

This plugin can be used to sniff GRE-redirected remote traffi c. The basic idea is to create a GRE tunnel that sends all the traffi c on a router interface to the ettercap machine. The plugin will send back the GRE packets to the router, after ettercap "manipulation" (you can use "active" plugins such as smb_down, ssh decryption, fi lters, etc... on redirected traffi c) It needs a "fake" host where the traffi c has to be redirected to (to avoid kernel's responses). The "fake" IP will be the tunnel endpoint. Gre_relay plugin will impersonate the "fake" host. To fi nd an unused IP address for the "fake" host you can use fi nd_ip plugin. Based on the original Tunnelx technique by Anthony C. Zboralski published in <http://www.phrack.org/show.php?p=56&a=10> by HERT.

gw_discover

This plugin try to discover the gateway of the lan by sending TCP SYN packets to a remote host. The packet has the destination IP of a remote host and the destination mac address of a local host. If ettercap receives the SYN+ACK packet, the host which own the source mac address of the reply is the gateway. This operation is repeated for each host in the 'host list', so you need to have a valid host list before launching this plugin.

example :

```
ettercap -TP gw_discover /192.168.0.1-50/
```

isolate

The isolate plugin will isolate an host form the LAN. It will poison the victim's arp cache with its own mac address associated with all the host it tries to contact. This way the host will not be able to contact other hosts because the packet will never reach the wire.

You can specify all the host or only a group. the targets specification work this way: the target1 is the victim and must be a single host, the target2 can be a range of addresses and represent the hosts that will be blocked to the victim.

examples :

```
ettercap -TzqP isolate /192.168.0.1/ //
```

```
ettercap -TP isolate /192.168.0.1/ /192.168.0.2-30/
```

link_type

It performs a check of the link type (hub or switch) by sending a spoofed ARP request and listening for replies. It needs at least one entry in the host list to perform the check. With two or more hosts the test will be more accurate.

example :

```
ettercap -TQP link_type /192.168.0.1/
ettercap -TQP link_type //
```

pptp_chapms1

It forces the pptp tunnel to negotiate MS-CHAPv1 authentication instead of MS-CHAPv2, that is usually easier to crack (for example with LC4). You have to be in the "middle" of the connection to use it successfully. It hooks the ppp dissector, so you have to keep them active.

pptp_clear

Forces no compression/encryption for pptp tunnels during negotiation. It could fail if client (or the server) is configured to hang off the tunnel if no encryption is negotiated. You have to be in the "middle" of the connection to use it successfully. It hooks the ppp dissector, so you have to keep them active.

pptp_pap

It forces the pptp tunnel to negotiate PAP (cleartext) authentication. It could fail if PAP is not supported, if pap_secret file is missing, or in case windows is configured with "automatic use of domain account". (It could fail for many other reasons too). You have to be in the "middle" of the connection to use it successfully. It hooks the ppp dissector, so you have to keep them active.

pptp_reneg

Forces re-negotiation on an existing pptp tunnel. You can force re-negotiation for grabbing passwords already sent. Furthermore you can launch it to use pptp_pap, pptp_chapms1 or pptp_clear on existing tunnels (those plugins work only during negotiation phase). You have to be in the "middle" of the connection to use it successfully. It hooks the ppp dissector, so you have to keep them active.

rand_flood

Floods the LAN with random MAC addresses. Some switches will fail open in repeating mode, facilitating sniffing. The delay between each packet is based on the port_steal_send_delay value in etter.conf.

It is useful only on ethernet switches.

example :

```
ettercap -TP rand_fbod
```

remote_browser

It sends to the browser the URLs sniffed thru HTTP sessions. So you are able to see the webpages in real time. The command executed is configurable in the etter.conf(5) file. It sends to the browser

only the GET requests and only for webpages, ignoring single request to images or other amenities. Don't use it to view your own connection :)

reply_arp

Simple arp responder. When it intercepts an arp request for a host in the targets' lists, it replies with attacker's MAC address.

example :

```
ettercap -TQzP reply_arp /192.168.0.1/
ettercap -TQzP reply_arp //
```

repoison_arp

It solicits poisoning packets after broadcast ARP requests (or replies) from a poisoned host. For example: we are poisoning Group1 impersonating Host2. If Host2 makes a broadcast ARP request for Host3, it is possible that Group1 caches the right MAC address for Host2 contained in the ARP packet. This plugin re-poisons Group1 cache immediately after a legal broadcast ARP request (or reply).

This plugin is effective only during an arp-poisoning session.

In conjunction with reply_arp plugin, repoison_arp is a good support for standard arp-poisoning mitm method.

example :

```
ettercap -T -M arp:remote -P repoison_arp /192.168.0.10-20/ /192.168.0.1/
```

scan_poisoner

Check if someone is poisoning between some host in the list and us. First of all it checks if two hosts in the list have the same mac address. It could mean that one of those is poisoning us pretending to be the other. It could generate many false-positives in a proxy-arp environment. You have to build hosts list to perform this check. After that, it sends icmp echo packets to each host in the list and checks if the source mac address of the reply differs from the address we have stored in the list for that ip. It could mean that someone is poisoning that host pretending to have our ip address and forwards intercepted packets to us. You can't perform this active test in unoffensive mode.

example :

```
ettercap -TQP scan_poisoner //
```

search_promisc

It tries to find if anyone is sniffing in promisc mode. It sends two different kinds of malformed arp request to each target in the host list and waits for replies. If a reply arrives from the target host, it's more or less probable that this target has the NIC in promisc mode. It could generate false-positives. You can launch it either from the command line or from the plugin menu. Since it listens for arp replies it is better that you don't use it while sending arp request.

example :

```
ettercap -TQP search_promisc /192.168.0.1/
ettercap -TQP search_promisc //
```

smb_clear

It forces the client to send smb password in clear-text by mangling protocol negotiation. You have to be in the "middle" of the connection to successfully use it. It hooks the smb dissector, so you have to keep it active. If you use it against a windows client it will probably result in a failure. Try it against a *nix smbclient :)

smb_down

It forces the client to not to use NTLM2 password exchange during smb authentication. This way, obtained hashes can be easily cracked by LC4. You have to be in the "middle" of the connection to successfully use it. It hooks the smb dissector, so you have to keep it active.

stp_mangler

It sends spanning tree BPDUs pretending to be a switch with the highest priority. Once in the "root" of the spanning tree, ettercap can receive all the "unmanaged" network traffic.

It is useful only against a group of switches running STP.

If there is another switch with the highest priority, try to manually decrease your MAC address before running it.

example :

```
ettercap -TP stp_mangler
```

SEE ALSO

ettercap(8) ettercap_curses(8) etterlog(8) etterfilter(8) etter.conf(5)