

Building a dynamic blacklist with Netfilter's Recent module

KEEP OUT!

Netfilter's Recent module builds a temporary blacklist to keep intruders off your network.

BY MICHAEL SCHWARTZKOPFF

When an Intrusion Detection System (IDS) recognizes an attack, it issues a strict “keep out” order to block the intruder’s access to services. Unfortunately, other systems on the network might not benefit from this block. The Recent module [1] by Netfilter [2] dynamically updates the firewall access rules to create a temporary “bad guy” list. You can then configure the firewall rules so that an IP address that breaks a rule is temporarily prevented from any form of access. Recent also comes with special features for fighting port scans, and you can combine the Recent module with an external application such as Snort [3] to create a fast and effective framework for detecting and closing out network attacks.

Recent Affairs

Firewalls are much more than just filter list managers. Stateful inspection has been a basic requirement for many years, and sophisticated devices dynamically adapt rulesets to keep an attacker

from all access points. The Recent module brings some of this high-end functionality to the Netfilter firewall system.

Recent supports a `--set` option that adds the source address of the packet currently being inspected to a list. If the firewall detects typical attack patterns, it can autonomously add entries to the Recent list. `--rcheck` tells Netfilter to check to see whether the source address for the current packet is already on the list. Ad-

ministrators can also set a `--seconds` option to specify how long the address is banned. The `--update` option does the same as `--rcheck` but resets the timestamp in the process.

The example in Listing 1 shows a typical use of the Recent module. Line 1 makes sure that packets from existing connections are allowed to pass (stateful inspection). Line 2 checks to see whether the Recent module already knows the source and, if so, blocks the source IP address. In a full script, the rules for permitted communication would follow.

Blockade

The command in line 4 of Listing 1 fills the Recent list with the IP addresses of all attackers that attempt to access port 135/TCP from an external interface. TCP port 135 is used by many worms to infiltrate Windows systems by exploiting a DCOM RPC vulnerability. The last line specifies `DROP` as the default policy. If the same computer tries again

Listing 1: Firewall Rules

```
01 iptables -A FORWARD -m state
   --state ESTABLISHED,RELATED -j
   ACCEPT
02 iptables -A FORWARD -m recent
   --update -j DROP
03 [other rules]
04 iptables -A FORWARD -i
   External Interface -p tcp
   --dport 135 -m recent --set -j
   DROP
05 iptables -A FORWARD -j DROP
```

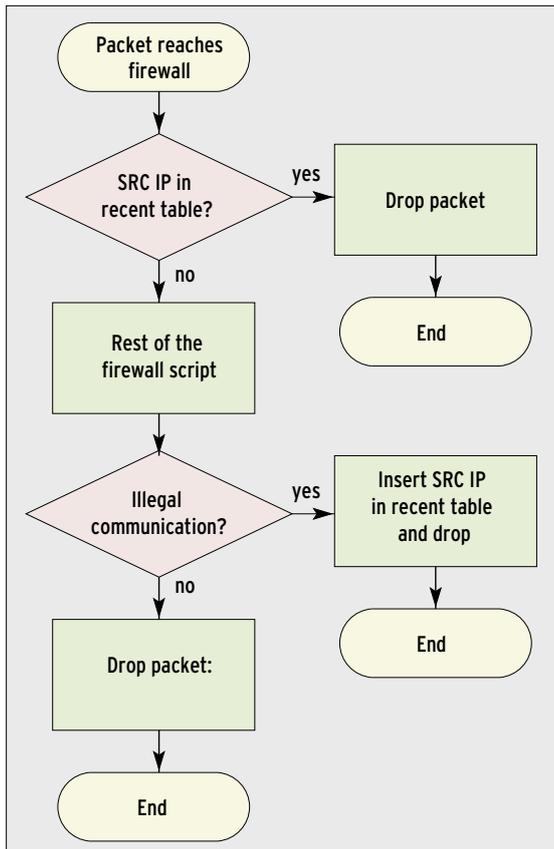


Figure 1: When a packet reaches the firewall, the Recent module checks to see whether the source is known. If it is, it drops the packet. If not, the remainder of the script is executed. If the script detects prohibited behavior, it adds the culprit to the Recent list.

within 60 seconds (default timeout), the firewall blocks the attempt regardless of the target port or target address (line 2).

At the same time, the Recent module resets the timestamp (because of the `--update` option), and the 60-second timeout starts again. The flow diagram in Figure 1 shows the path the packets take through the firewall script.

Inside the Module

The Recent module can manage multiple dynamic lists – for example, one for the mail server and one for all other systems. The firewall script uses the `--name` option to specify the list. Without this parameter, the module will use the `DEFAULT` list.

Lists are accessible via the `proc` directory, `/proc/net/ipt_recent/<list>`, in user space. This directory is where the Recent module stores the addresses and timestamps for any packets it has detected. It is also possible to add entries with a user-space script:

```
echo <Address> > /proc/net/ipt_recent/DEFAULT
```

A script is useful if you need to block communications to and from a specified IP address. You can also delete numbers from the list in the same way:

```
echo <-Address> > /proc/net/ipt_recent/DEFAULT
```

The Recent module typically manages a maximum of 100 entries per list. When new entries appear, they replace older entries, even if the timeout has not yet expired. This can be an issue with fast Internet connections, but you can easily change the defaults: `modprobe ipt_recent ip_list_tot=1000` increases the maximum number of entries to 1000 when you restart the module.

Port Scan Defense

One of Recent's most practical features is its ability to make Internet-based port scans more difficult for attackers. Port scans tend to traverse through a block of consecutive IP addresses. To catch the scanner, the Recent module helps you in utilizing an unused address at the start and end of the block and prohibit all communication with these IPs. Recent will then regard any attempts to talk to these addresses as a potential attack and add the source address to your blacklist.

If the attacker then continues the scan, they will not receive any responses, even if they try to scan a port that would normally be permitted on a host in the middle of your address block. This approach has the advantage of taking effect when the first packet arrives, rather than after a specific time period or after a specific number of packets.

Listing 2a: Snort Configuration

```
01 output alert_syslog: LOG_AUTH LOG_ALERT
```

Because there is no legitimate communication with these address traps, the method is quite secure. However, this technique does not protect you against IP spoofing (see the “DoS Attacks” box), and it does not detect port scans that start in the middle of the address block or scan random addresses.

Experience Counts

As a practical example, I'll look at SSH, which I like to use for remote management. Because I can't know in advance which address I will use to access the firewall (think vacation in Greece), I allow all addresses. A quick check of the logfiles reveals that the firewall has registered 207 accounts of illegal access (portscans or Rumpelstiltskin attacks).

The Recent module triggered 187 times and prevented a connection. Only 10 connections were established, and all of them failed to negotiate the SSH authentication phase. The IPTables Limit module triggered 10 times; its task is to restrict the connection to a sensible rate – attackers failed to discover an open port in 95 percent of the cases.

The author of the Recent module actually suggests adding `-m recent --set` to augment the last firewall blocking rule (default policy). This would add any unknown packets to the blacklist, but drastic measures of this kind are typically counterproductive. In fact, it makes much more sense to be careful to block

DoS Attacks

If you do not define your Recent module rules carefully, you can easily open up a vector for Denial of Service attacks. An attacker could send a UDP packet with a spoofed source address to the dummy target. The firewall would detect an attack and block the presumed culprit, and this might lead to your best customer losing the ability to order online.

To prevent this problem, you need to be very careful when specifying the conditions under which your firewall will add a host to the blacklist (`-m recent --set`). It makes sense to evaluate only TCP packets after completing the handshake because IP spoofing becomes far more difficult after the handshake is complete. You should also take steps to avoid the possibility of blacklisting critical external machines, such as your provider's secondary MX or VPN addresses used by your own staff.

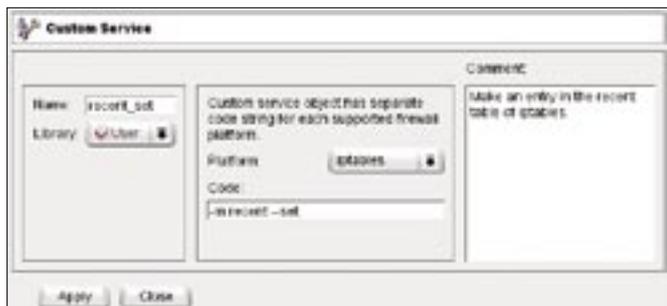


Figure 2: Custom services in Firewall Builder give administrators control of the Recent module via the GUI. In this example, `-m recent --set` is mapped to the GUI equivalent `recent_set`.

only malevolent communication attempts. (See the “DoS Attacks” box.)

More Convenient with Firewall Builder

Firewall Builder [4], one of the best GUIs for Linux firewalls, makes it easy to create intuitive rules for a firewall, build the firewall script, and install the script on the firewall. With a couple of tricks, you can add rules for the Recent module. The first step of the process is to add some new services; in the GUI, select *User* | *Services* | *Custom* (see Figure 2).

You need to define a custom *recent_update* service in the same way; this

Listing 2b: Syslog-NG

```
01 filter f_snort {
02   facility(auth) and
03   match("snort") and
04   match("Priority: 1");
05 }
06 destination snort {
07   program("/usr/local/sbin/
08   blocker.pl");
09 }
10 log {
11   source(src);
12   filter(f_snort);
13   destination(snort);
14 }
```

address. Two NAT rules pass the start and end addresses from the external address block to this machine. Figure 3 shows you how this is reflected in the firewall rules.

Firewall Builder has a few hidden surprises. To handle complex rules with multiple sources and targets, the program generates a number of rules with their own chains that query the objects sequentially. In this case, a call to the Recent module might add the address in question to the blacklist right at the start, although you might discover that the rule does not apply at a later stage. Of course, all further communication has already been interrupted by this point. All you can do is keep the rule so simple that fwbuilder does not use a new chain and manually check rulesets generated by Firewall Builder.

External Applications

The Recent module lets you use external applications to modify the blacklist. This feature is interesting because IDS applications such as Snort often know much more than the firewall about what kind of communications to allow.

The Snort configuration in Listing 2a ensures that the IDS will report all events as Syslog alerts, tag them with the *LOG_AUTH* facility, and give them a priority of *LOG_ALERT*. Syslog-NG [5] filters all events that contain the strings

maps to `-m recent --update`. More cautious spirits would maybe like to add `-i external interface` to their definitions.

To support the traps at the start and end of our IP address block, we can quickly set up a dummy machine with an internal

snort and *Priority: 1* (line 2 in Listing 2b) and sends them to a Perl script (line 5). The blocking script is shown in Listing 2c. If the log entries contain a signature of 1810 with a priority of 1 (successful attack via SSH, line 4), the script will add the attacker’s IP address to the Recent module’s *BLOCK* list (line 5).

In this scenario, the firewall script has to process the *BLOCK* before the lines with *ESTABLISHED,RELATED* because the connection has already been established when the block is imposed.

Use Your Imagination

These simple examples are indicative of the kind of flexibility you can achieve by adding the Recent module to your firewall configuration. In case of zero-day attacks or patches that do not work perfectly, evaluating application logfiles or integrating the output from an IDS can help you build a very effective intrusion prevention system. ■

Listing 2c: Blocker Script

```
01 #!/usr/bin/perl
02 # /usr/local/sbin/blocker.pl
03 while (<>) {
04   if ( /\.*\[1:1810:12\].* ->
05     ((\d{1,3}\.){3}(\d{1,3})/ ) / )
06     {
07       system "echo $1 > /proc/
08       net/iptables/BLOCK";
09     }
10 }
```

INFO

- [1] IPTables Recent module: http://www.snowman.net/projects/iptables_recent/
- [2] Netfilter (IPTables): <http://www.netfilter.org>
- [3] Snort: <http://www.snort.org>
- [4] Firewall Builder: <http://www.fwbuilder.org>
- [5] Syslog-NG: <http://freshmeat.net/projects/syslog-ng/>

Policy	NAT	Routing	Source	Destination	Service	Interface	Chain	Action	Options	Comment
0			red:192.168.1.0	Any	Private_Update	outside	red	Deny		Bad guys stay outside
1			Any	192.168.1.0	SSH	All	red	Deny		Deny access for backdoor or poor mans VPN
2			red:192.168.1.0	Any	Any	All	red	Deny		Internal clients do everything
3			Any	Any	Private_Set	outside	red	Deny		Bad guys are marked
4			Any	Any	Any	All	red	Deny		Drop all

Figure 3: This ruleset for a simple firewall autonomously detects attacks and adds their source IP addresses to the blacklist. To do so, it uses a custom service to control the Recent module.

THE AUTHOR

Dr. Michael Schwartzkopff works as a security and network consultant for Multinet Services GmbH and specializes in SNMP. The Linux bug first bit Michael back in 1994 while he was experimenting with a Yggdrasil distribution.