

Re: loading huge number of rules in iptables (blocklist)

Re: loading huge number of rules in iptables (blocklist)

Source: <http://linux.derkeiler.com/Mailing-Lists/Debian/2007-03/msg03793.html>

- *From:* Andrew Sackville-West <andrew@xxxxxxxxxxxxxxxxxxxxxxxx>
 - *Date:* Wed, 21 Mar 2007 12:03:34 -0700
-

On Wed, Mar 21, 2007 at 02:30:06PM -0400, H.S. wrote:

Andrew Sackville-West wrote:

I'm sorry, but what exactly is the purpose here? I did a little poking around and it looks like just a massive list of ip's to block, but for what purpose?

I'm not trying to say that this is not the right solution for whatever your problem is, but it certainly seems very brute force. Hence my questions.

We were discussing some rogue p2p sites which try to connect to bittorrent clients to collect information about the users. The discussion was prompted by a number of posts on slashdot, which led to peerguardian website and kind of took off from there. The purpose is to block/drop traffic from all the ip ranges listed in blocklist provided by peerguardian website. I can give more pointers if this is not sufficient.

okay, I follow... and you want otherwise unfettered p2p operating, but security from these particular sites. ugh. nasty problem.

The result was the experiment to use the massive blocklist and to automate the process in iptables firewall on a router -- needs iptables, bash, curl and maybe python or perl. I am giving it a shot. As I said before, this is the first attempt.

so, is there some other way to use this info besides a massive iptables rule set? I'm in territory I don't understand, so feel free to ignore me :). What about a proxy? instead of a ruleset in the firewall, run the whole thing through a proxy that is set up to read

Re: loading huge number of rules in iptables (blocklist)

Re: loading huge number of rules in iptables (blocklist)

the list of denies. then a simple update of the list can result in new blocking without reloading a whole set of rules. I don't have a clue as to the mechanics of this.

Another possibility, from my reading it appears this is supposed to work with a program called moblock. moblock seems to do some parsing to eliminate duplicates and so forth. I've done a little grepping through the list and can tell that it could be done more efficiently. First, there are duplicates as shown here:

```
andrew@basement:~$ zcat level1.gz | wc -l
151663
andrew@basement:~$ zcat level1.gz | cut -d: -f2 | uniq | wc -l
150695
```

that's 1000 rules gone right there. (the cut eliminates the name, giving just the ip range.

Also, a little scripting could probably concatenate a lot of the ranges. just a cursory look through shows that there are contiguous ranges specified on different lines. I don't have time today to hack at it, but I think you might be able to cut as much as 25% out of the list that way.

for example, just looking at a quick one: there are 7 lines used to specify different ranges that add up to 12.9.27.63–12.9.27.223 being blocked. so maybe my 25% is pessimistic. I think its worth the time. Pick through the whole list and concatenate all the ranges together and save it out somewhere. then when a change comes into the list, diff it with the original and incorporate the changes into your saved out list. This way, you only have to process the whole list one time and then make incremental changes as they come up.

.02

A

Attachment: signature.asc

Description: Digital signature