NAME

ufw − program for managing a netfilter firewall

DESCRIPTION

This program is for managing a Linux firewall and aims to provide an easy to use interface for the user.

USAGE

ufw [**−−dry−run**] **enable|disable|reload**

ufw [**−−dry−run**] **default** allow|deny|reject [incoming|outgoing|routed]

ufw [**−−dry−run**] **logging** on|off|LEVEL

ufw [**−−dry−run**] **reset**

ufw [**−−dry−run**] **status** [verbose|numbered]

ufw [**−−dry−run**] **show** REPORT

ufw [**−−dry−run**] [**delete**] [**insert** NUM] [**prepend**] **allow|deny|reject|limit** [**in|out**] [**log|log−all**] [PORT[/PROTOCOL] | APPNAME] [**comment** COMMENT]

ufw [**−−dry−run**] [**rule**] [**delete**] [**insert** NUM] [**prepend**] **allow|deny|reject|limit** [**in|out** [**on** INTERFACE]] [**log|log−all**] [**proto** PROTOCOL] [**from** ADDRESS [**port** PORT | **app** APPNAME]] [**to** ADDRESS [**port** PORT | **app** APPNAME]] [**comment** COMMENT]

ufw [**−−dry−run**] **route** [**delete**] [**insert** NUM] [**prepend**] **allow|deny|reject|limit** [**in|out on** INTERFACE] [**log|log−all**] [**proto** PROTOCOL] [**from** ADDRESS [**port** PORT | **app** APPNAME]] [**to** ADDRESS [**port** PORT | **app** APPNAME]] [**comment** COMMENT]

ufw [**−−dry−run**] [**−−force**] **delete** NUM

ufw [**−−dry−run**] **app list|info|default|update**

OPTIONS

−−version
show program’s version number and exit

−h, **−−help**
show help message and exit

−−dry−run
don’t modify anything, just show the changes

enable reloads firewall and enables firewall on boot.

disable unloads firewall and disables firewall on boot

reload reloads firewall

default allow|deny|reject DIRECTION
change the default policy for traffic going DIRECTION, where DIRECTION is one of **incoming**, **outgoing** or **routed**. Note that existing rules will have to be migrated manually when changing the default policy. See **RULE SYNTAX** for more on **deny** and **reject**.

logging on|off|LEVEL
toggle logging. Logged packets use the LOG_KERN syslog facility. Systems configured for rsyslog support may also log to /var/log/ufw.log. Specifying a LEVEL turns logging on for the specified LEVEL. The default log level is ’low’. See**LOGGING** for details.

reset Disables and resets firewall to installation defaults. Can also give the **−−force** option to perform the reset without confirmation.

status show status of firewall and ufw managed rules. Use **status verbose** for extra information. In the status output, ’Anywhere’ is synonymous with ’any’, 0.0.0.0/0 (IPv4) and ::/0 (IPv6). Note that when using **status**, there is a subtle difference when reporting interfaces. For example, if the

following rules are added:

```
ufw allow in on eth0 from 192.168.0.0/16
ufw allow out on eth1 to 10.0.0.0/8
ufw route allow in on eth0 out on eth1 to 10.0.0.0/8 from 192.168.0.0/16
ufw limit 2222/tcp comment 'SSH port'
```

ufw status will output:

```
To                         Action      From
--                         ------      ----
Anywhere on eth0           ALLOW       192.168.0.0/16
10.0.0.0/8                 ALLOW OUT   Anywhere on eth1
10.0.0.0/8 on eth1         ALLOW FWD   192.168.0.0/16 on eth0
Anywhere                   LIMIT       Anywhere                # SSH port
```

For the input and output rules, the interface is reported relative to the firewall system as an endpoint, whereas with route rules, the interface is reported relative to the direction packets flow through the firewall.

show REPORT
display information about the running firewall. See **REPORTS**

allow ARGS
add allow rule. See **RULE SYNTAX**

deny ARGS
add deny rule. See **RULE SYNTAX**

reject ARGS
add reject rule. See **RULE SYNTAX**

limit ARGS
add limit rule. See **RULE SYNTAX**

delete RULE|NUM
deletes the corresponding RULE

insert NUM RULE
insert the corresponding RULE as rule number NUM

prepend RULE
prepend the corresponding RULE to the top of the ruleset

RULE SYNTAX

Users can specify rules using either a simple syntax or a full syntax. The simple syntax only specifies the port and optionally the protocol to be allowed or denied on the host.

Both syntaxes support specifying a comment for the rule. For existing rules, specifying a different comment updates the comment and specifying '' removes the comment.

Example rules using the simple syntax:

```
ufw allow 53
```

This rule will allow tcp and udp port 53 to any address on this host. To specify a protocol, append '/protocol' to the port. For example:

```
ufw allow 25/tcp
```

This will allow tcp port 25 to any address on this host. **ufw** will also check /etc/services for the port and protocol if specifying a service by name. Eg:

```
ufw allow smtp
```

ufw supports both ingress and egress filtering and users may optionally specify a direction of either **in** or **out** for either incoming or outgoing traffic. If no direction is supplied, the rule applies to incoming traffic. Eg:

```
ufw allow in http
ufw reject out smtp
ufw reject telnet comment 'telnet is unencrypted'
```

Users can also use a fuller syntax, specifying the source and destination addresses and ports. This syntax is loosely based on OpenBSD's PF syntax. For example:

```
ufw deny proto tcp to any port 80
```

This will deny all traffic to tcp port 80 on this host. Another example:

```
ufw deny proto tcp from 10.0.0.0/8 to 192.168.0.1 port 25
```

This will deny all traffic from the RFC1918 Class A network to tcp port 25 with the address 192.168.0.1.

```
ufw deny proto tcp from 2001:db8::/32 to any port 25
```

This will deny all traffic from the IPv6 2001:db8::/32 to tcp port 25 on this host. IPv6 must be enabled in /etc/default/ufw for IPv6 firewalling to work.

```
ufw deny in on eth0 to 224.0.0.1 proto igmp
```

This will deny all igmp traffic to 224.0.0.1 on the eth0 interface.

```
ufw allow in on eth0 to 192.168.0.1 proto gre
```

This will allow all gre traffic to 192.168.0.1 on the eth0 interface.

```
ufw allow proto tcp from any to any port 80,443,8080:8090 comment 'web app'
```

The above will allow all traffic to tcp ports 80, 443 and 8080−8090 inclusive and adds a comment for the rule. When specifying multiple ports, the ports list must be numeric, cannot contain spaces and must be modified as a whole. Eg, in the above example you cannot later try to delete just the '443' port. You cannot specify more than 15 ports (ranges count as 2 ports, so the port count in the above example is 4).

ufw supports several different protocols. The following are valid in any rule and enabled when the protocol is not specified:

```
tcp
udp
```

The following have certain restrictions and are not enabled when the protocol is not specified:

```
ah      valid without port number
```

esp valid without port number
gre valid without port number
ipv6 valid for IPv4 addresses and without port number
igmp valid for IPv4 addresses and without port number

Rules for traffic not destined for the host itself but instead for traffic that should be routed/forwarded through the firewall should specify the **route** keyword before the rule (routing rules differ significantly from PF syntax and instead take into account netfilter FORWARD chain conventions). For example:

ufw route allow in on eth1 out on eth2

This will allow all traffic routed to eth2 and coming in on eth1 to traverse the firewall.

ufw route allow in on eth0 out on eth1 to 12.34.45.67 port 80 proto tcp

This rule allows any packets coming in on eth0 to traverse the firewall out on eth1 to tcp port 80 on 12.34.45.67.

In addition to routing rules and policy, you must also setup IP forwarding. This may be done by setting the following in /etc/ufw/sysctl.conf:

net/ipv4/ip_forward=1
net/ipv6/conf/default/forwarding=1
net/ipv6/conf/all/forwarding=1

then restarting the firewall:

ufw disable
ufw enable

Be aware that setting kernel tunables is operating system specific and **ufw** sysctl settings may be overridden. See the **sysctl** manual page for details.

ufw supports connection rate limiting, which is useful for protecting against brute−force login attacks. When a limit rule is used, **ufw** will normally allow the connection but will deny connections if an IP address attempts to initiate 6 or more connections within 30 seconds. See http://www.debian−administration.org/articles/187 for details. Typical usage is:

ufw limit ssh/tcp

Sometimes it is desirable to let the sender know when traffic is being denied, rather than simply ignoring it. In these cases, use **reject** instead of **deny**. For example:

ufw reject auth

By default, **ufw** will apply rules to all available interfaces. To limit this, specify **DIRECTION on INTERFACE**, where DIRECTION is one of **in** or **out** (interface aliases are not supported). For example, to allow all new incoming http connections on eth0, use:

ufw allow in on eth0 to any port 80 proto tcp

To delete a rule, simply prefix the original rule with **delete** with or without the rule comment. For example, if the original rule was:

ufw deny 80/tcp

Use this to delete it:

ufw delete deny 80/tcp

You may also specify the rule by NUM, as seen in the **status numbered** output. For example, if you want to delete rule number '3', use:

ufw delete 3

If you have IPv6 enabled and are deleting a generic rule that applies to both IPv4 and IPv6 (eg 'ufw allow 22/tcp'), deleting by rule number will delete only the specified rule. To delete both with one command, prefix the original rule with **delete**.

To insert a rule, specify the new rule as normal, but prefix the rule with the rule number to insert. For example, if you have four rules, and you want to insert a new rule as rule number three, use:

ufw insert 3 deny to any port 22 from 10.0.0.135 proto tcp

Similarly, to add a rule before all other rules matching the rule's IP type, use the prepend rule:

ufw prepend deny from 1.2.3.4

This is particularly useful for dynamic firewalls as found in an IPS. Importantly, if the specified rule is an IPv4 rule, it will be prepended before all other IPv4 rules. If it is an IPv6 rule, it will be prepended before any IPv6 rules.

To see a list of numbered rules, use:

ufw status numbered

ufw supports per rule logging. By default, no logging is performed when a packet matches a rule. Specifying **log** will log all new connections matching the rule, and **log−all** will log all packets matching the rule. For example, to allow and log all new ssh connections, use:

ufw allow log 22/tcp

See **LOGGING** for more information on logging.

EXAMPLES

Deny all access to port 53:

ufw deny 53

Allow all access to tcp port 80:

ufw allow 80/tcp

Allow all access from RFC1918 networks to this host:

ufw allow from 10.0.0.0/8
ufw allow from 172.16.0.0/12

ufw allow from 192.168.0.0/16

Deny access to udp port 514 from host 1.2.3.4:

ufw deny proto udp from 1.2.3.4 to any port 514

Allow access to udp 1.2.3.4 port 5469 from 1.2.3.5 port 5469:

ufw allow proto udp from 1.2.3.5 port 5469 to 1.2.3.4 port 5469

REMOTE MANAGEMENT

When running **ufw enable** or starting **ufw** via its initscript, **ufw** will flush its chains. This is required so **ufw** can maintain a consistent state, but it may drop existing connections (eg ssh). **ufw** does support adding rules before enabling the firewall, so administrators can do:

ufw allow proto tcp from any to any port 22

before running '**ufw enable**'. The rules will still be flushed, but the ssh port will be open after enabling the firewall. Please note that once ufw is 'enabled', **ufw** will not flush the chains when adding or removing rules (but will when modifying a rule or changing the default policy). By default, **ufw** will prompt when enabling the firewall while running under ssh. This can be disabled by using '**ufw −−force enable**'.

APPLICATION INTEGRATION

ufw supports application integration by reading profiles located in /etc/ufw/applications.d. To list the names of application profiles known to **ufw**, use:

ufw app list

Users can specify an application name when adding a rule (quoting any profile names with spaces). For example, when using the simple syntax, users can use:

ufw allow <name>

Or for the extended syntax:

ufw allow from 192.168.0.0/16 to any app <name>

You should not specify the protocol with either syntax, and with the extended syntax, use **app** in place of the **port** clause.

Details on the firewall profile for a given application can be seen with:

ufw app info <name>

where '<name>' is one of the applications seen with the app list command. Users may also specify **all** to see the profiles for all known applications.

Syntax for the application profiles is a simple .INI format:

```
[<name>]
title=<title>
description=<description>
ports=<ports>
```

The 'ports' field may specify a '|'-separated list of ports/protocols where the protocol is optional. A comma-separated list or a range (specified with 'start:end') may also be used to specify multiple ports, in which case the protocol is required. For example:

```
[SomeService]
title=Some title
description=Some description
ports=12/udp|34|56,78:90/tcp
```

In the above example, 'SomeService' may be used in app rules and it specifies UDP port 12, TCP and UDP on port 34 and TCP ports 56 and 78-90 inclusive.

After creating or editing an application profile, users can run:

```
ufw app update <name>
```

This command will automatically update the firewall with updated profile information. If specify 'all' for name, then all the profiles will be updated. To update a profile and add a new rule to the firewall automatically, users can run:

```
ufw app update --add-new <name>
```

The behavior of the **update --add-new** command can be configured using:

```
ufw app default <policy>
```

The default application policy is **skip**, which means that the **update --add-new** command will do nothing. Users may also specify a policy of **allow** or **deny** so the **update --add-new** command may automatically update the firewall. **WARNING:** it may be a security to risk to use a default **allow** policy for application profiles. Carefully consider the security ramifications before using a default **allow** policy.

LOGGING

ufw supports multiple logging levels. **ufw** defaults to a loglevel of 'low' when a loglevel is not specified. Users may specify a loglevel with:

```
ufw logging LEVEL
```

LEVEL may be 'off', 'low', 'medium', 'high' and 'full'. Log levels are defined as:

off disables ufw managed logging

low logs all blocked packets not matching the defined policy (with rate limiting), as well as packets matching logged rules

medium
log level low, plus all allowed packets not matching the defined policy, all INVALID packets, and all new connections. All logging is done with rate limiting.

high log level medium (without rate limiting), plus all packets with rate limiting

full log level high without rate limiting

Loglevels above medium generate a lot of logging output, and may quickly fill up your disk. Loglevel medium may generate a lot of logging output on a busy system.

Specifying 'on' simply enables logging at log level 'low' if logging is currently not enabled.

REPORTS

The following reports are supported. Each is based on the live system and with the exception of the **listening** report, is in raw iptables format:

raw
builtins
before−rules
user−rules
after−rules
logging−rules
listening
added

The **raw** report shows the complete firewall, while the others show a subset of what is in the **raw** report.

The **listening** report will display the ports on the live system in the listening state for tcp and the open state for udp, along with the address of the interface and the executable listening on the port. An '*' is used in place of the address of the interface when the executable is bound to all interfaces on that port. Following this information is a list of rules which may affect connections on this port. The rules are listed in the order they are evaluated by the kernel, and the first match wins. Please note that the default policy is not listed and tcp6 and udp6 are shown only if IPV6 is enabled.

The **added** report displays the list of rules as they were added on the command−line. This report does not show the status of the running firewall (use '**ufw status**' instead). Because rules are normalized by **ufw**, rules may look different than the originally added rule. Also, **ufw** does not record command ordering, so an equivalent ordering is used which lists IPv6−only rules after other rules.

NOTES

On installation, **ufw** is disabled with a default incoming policy of deny, a default forward policy of deny, and a default outgoing policy of allow, with stateful tracking for NEW connections for incoming and forwarded connections. In addition to the above, a default ruleset is put in place that does the following:

- DROP packets with RH0 headers

- DROP INVALID packets

- ACCEPT certain icmp packets (INPUT and FORWARD): destination-unreachable, source-quench, time-exceeded, parameter-problem, and echo-request for IPv4. destination-unreachable, packet-too-big, time-exceeded, parameter-problem, and echo-request for IPv6.

- ACCEPT icmpv6 packets for stateless autoconfiguration (INPUT)

- ACCEPT ping replies from IPv6 link-local (ffe8::/10) addresses (INPUT)

- ACCEPT DHCP client traffic (INPUT)

- DROP non-local traffic (INPUT)

- ACCEPT mDNS (zeroconf/bonjour/avahi 224.0.0.251 for IPv4 and ff02::fb for IPv6) for service discovery (INPUT)

- ACCEPT UPnP (239.255.255.250 for IPv4 and ff02::f for IPv6) for service discovery (INPUT)

Rule ordering is important and the first match wins. Therefore when adding rules, add the more specific rules first with more general rules later.

ufw is not intended to provide complete firewall functionality via its command interface, but instead provides an easy way to add or remove simple rules.

The status command shows basic information about the state of the firewall, as well as rules managed via the **ufw** command. It does not show rules from the rules files in /etc/ufw. To see the complete state of the

firewall, users can **ufw show raw**. This displays the filter, nat, mangle and raw tables using:

```
iptables −n −L −v −x −t <table>
ip6tables −n −L −v −x −t <table>
```

See the **iptables** and **ip6tables** documentation for more details.

If the default policy is set to REJECT, **ufw** may interfere with rules added outside of the ufw framework. See README for details.

IPV6 is allowed by default. To change this behavior to only accept IPv6 traffic on the loopback interface, set IPV6 to 'no' in /etc/default/ufw and reload **ufw**. When IPv6 is enabled, you may specify rules in the same way as for IPv4 rules, and they will be displayed with **ufw status**. Rules that match both IPv4 and IPv6 addresses apply to both IP versions. For example, when IPv6 is enabled, the following rule will allow access to port 22 for both IPv4 and IPv6 traffic:

```
ufw allow 22
```

IPv6 over IPv4 tunnels and 6to4 are supported by using the 'ipv6' protocol ('41'). This protocol can only be used with the full syntax. For example:

```
ufw allow to 10.0.0.1 proto ipv6
ufw allow to 10.0.0.1 from 10.4.0.0/16 proto ipv6
```

IPSec is supported by using the 'esp' ('50') and 'ah' ('51') protocols. These protocols can only be used with the full syntax. For example:

```
ufw allow to 10.0.0.1 proto esp
ufw allow to 10.0.0.1 from 10.4.0.0/16 proto esp
ufw allow to 10.0.0.1 proto ah
ufw allow to 10.0.0.1 from 10.4.0.0/16 proto ah
```

In addition to the command−line interface, **ufw** also provides a framework which allows administrators to modify default behavior as well as take full advantage of netfilter. See the **ufw−framework** manual page for more information.

SEE ALSO

ufw−framework(8), **iptables**(8), **ip6tables**(8), **iptables−restore**(8), **ip6tables−restore**(8), **sysctl**(8), **sysctl.conf**(5)

AUTHOR