Ingress filtering

From Wikipedia, the free encyclopedia

[Jump to navigation](https://en.wikipedia.org/wiki/Ingress_filtering#mw-head)[Jump to search](https://en.wikipedia.org/wiki/Ingress_filtering#searchInput)

|  |  |
| --- | --- |
|  | This article **needs additional citations for**[**verification**](https://en.wikipedia.org/wiki/Wikipedia:Verifiability). Please help [improve this article](https://en.wikipedia.org/w/index.php?title=Ingress_filtering&action=edit) by [adding citations to reliable sources](https://en.wikipedia.org/wiki/Help:Referencing_for_beginners). Unsourced material may be challenged and removed. *Find sources:* ["Ingress filtering"](https://www.google.com/search?as_eq=wikipedia&q=%22Ingress+filtering%22) – [news](https://www.google.com/search?tbm=nws&q=%22Ingress+filtering%22+-wikipedia) **·** [newspapers](https://www.google.com/search?&q=%22Ingress+filtering%22+site:news.google.com/newspapers&source=newspapers) **·** [books](https://www.google.com/search?tbs=bks:1&q=%22Ingress+filtering%22+-wikipedia) **·** [scholar](https://scholar.google.com/scholar?q=%22Ingress+filtering%22) **·** [JSTOR](https://www.jstor.org/action/doBasicSearch?Query=%22Ingress+filtering%22&acc=on&wc=on) *(February 2014) (*[*Learn how and when to remove this template message*](https://en.wikipedia.org/wiki/Help:Maintenance_template_removal)*)* |

In [computer networking](https://en.wikipedia.org/wiki/Computer_networking), **ingress filtering** is a technique used to ensure that incoming [packets](https://en.wikipedia.org/wiki/Packet_(information_technology)) are actually from the networks from which they claim to originate. This can be used as a countermeasure against various [spoofing attacks](https://en.wikipedia.org/wiki/Spoofing_attack) where the attacker's packets contain fake [IP addresses](https://en.wikipedia.org/wiki/IP_address) to make it difficult to find the source of the attack. This technique is often used in the [denial-of-service attack](https://en.wikipedia.org/wiki/Denial-of-service_attack), and this is a primary target of ingress filtering.



**Contents**

* [1Problem](https://en.wikipedia.org/wiki/Ingress_filtering#Problem)
* [2Potential solutions](https://en.wikipedia.org/wiki/Ingress_filtering#Potential_solutions)
* [3Networks](https://en.wikipedia.org/wiki/Ingress_filtering#Networks)
* [4Deployment](https://en.wikipedia.org/wiki/Ingress_filtering#Deployment)
* [5See also](https://en.wikipedia.org/wiki/Ingress_filtering#See_also)
* [6References](https://en.wikipedia.org/wiki/Ingress_filtering#References)
* [7External links](https://en.wikipedia.org/wiki/Ingress_filtering#External_links)

Problem[[edit](https://en.wikipedia.org/w/index.php?title=Ingress_filtering&action=edit&section=1)]

Networks receive packets from other networks. Normally a packet will contain the [IP address](https://en.wikipedia.org/wiki/IP_address) of the computer that originally sent it. This allows devices in the receiving network to know where it came from, allowing a reply to be routed back (amongst other things), except when IP addresses are used through a proxy or a spoofed IP address, which does not pinpoint a specific user within that pool of users.

A sender IP address can be faked ('[spoofed](https://en.wikipedia.org/wiki/IP_address_spoofing)'), characterising a [spoofing attack](https://en.wikipedia.org/wiki/Spoofing_attack). This disguises the origin of packets sent, for example in a [denial-of-service attack](https://en.wikipedia.org/wiki/Denial-of-service_attack). The same holds true for proxies, although in a different manner than "IP spoofing."

Potential solutions[[edit](https://en.wikipedia.org/w/index.php?title=Ingress_filtering&action=edit&section=2)]

One potential solution involves implementing the use of intermediate Internet gateways (i.e., those servers connecting disparate networks along the path followed by any given packet) filtering or denying any packet deemed to be illegitimate. The gateway processing the packet might simply ignore the packet completely, or where possible, it might send a packet back to the sender relaying a message that the illegitimate packet has been denied. [Host intrusion prevention systems (HIPS)](https://en.wikipedia.org/wiki/Intrusion_prevention_system) are one example of technical engineering applications that help to identify, prevent and/or deter unwanted, unsuspected and/or suspicious events and intrusions.

Any router that implements ingress filtering checks the source IP field of IP packets it receives, and drops packets if the packets don't have an IP address in the IP address block, to which the interface is connected. This may not be possible if the end host is [multi-homed](https://en.wikipedia.org/wiki/Multi-homed) and also sends transit network traffic.

In ingress filtering, packets coming into the network are filtered if the network sending it should not send packets from the originating IP address(es). If the end host is a stub network or host, the router needs to filter all IP packets that have, as the source IP, [private addresses](https://en.wikipedia.org/wiki/Private_network) ([RFC 1918](https://tools.ietf.org/html/rfc1918)), [bogon addresses](https://en.wikipedia.org/wiki/Bogon_(address)) or addresses that do not have the same network address as the interface.[[1]](https://en.wikipedia.org/wiki/Ingress_filtering#cite_note-1)

Networks[[edit](https://en.wikipedia.org/w/index.php?title=Ingress_filtering&action=edit&section=3)]

*Network ingress filtering* is a [packet filtering](https://en.wikipedia.org/wiki/Packet_filtering) technique used by many [Internet service providers](https://en.wikipedia.org/wiki/Internet_service_provider) to try to prevent [source address spoofing](https://en.wikipedia.org/wiki/Source_address_spoofing) of Internet traffic, and thus indirectly combat various types of [net abuse](https://en.wikipedia.org/wiki/Computer_crime) by making Internet traffic traceable to its source.

Network ingress filtering is a "good neighbor" policy which relies on cooperation between ISPs for their mutual benefit.

The [best current practices](https://en.wikipedia.org/wiki/Best_current_practice) for network ingress filtering are documented by the [Internet Engineering Task Force](https://en.wikipedia.org/wiki/Internet_Engineering_Task_Force) in **BCP 38** and **BCP 84**, which are defined by [RFC 2827](https://tools.ietf.org/html/rfc2827) and [RFC 3704](https://tools.ietf.org/html/rfc3704), respectively.[[2]](https://en.wikipedia.org/wiki/Ingress_filtering#cite_note-2)[[3]](https://en.wikipedia.org/wiki/Ingress_filtering#cite_note-3)

BCP 84 recommends that [upstream providers](https://en.wikipedia.org/w/index.php?title=Upstream_provider&action=edit&redlink=1) of IP connectivity filter packets entering their networks from downstream customers, and discard any packets which have a source address which is not allocated to that customer.

There are many possible ways of implementing this policy; one common mechanism is to enable [reverse path forwarding](https://en.wikipedia.org/wiki/Reverse_path_forwarding) on links to customers, which will indirectly apply this policy based on the provider's [route filtering](https://en.wikipedia.org/wiki/Route_filtering) of their customers' [route announcements](https://en.wikipedia.org/w/index.php?title=Route_announcement&action=edit&redlink=1).

Deployment[[edit](https://en.wikipedia.org/w/index.php?title=Ingress_filtering&action=edit&section=4)]

As of 2012, one report suggests that, contrary to general opinion about the lack of BCP 38 deployment, some 80% of the Internet (by various measures) were already applying anti-spoofing packet filtering in their networks.[[4]](https://en.wikipedia.org/wiki/Ingress_filtering#cite_note-4)

See also[[edit](https://en.wikipedia.org/w/index.php?title=Ingress_filtering&action=edit&section=5)]

* [Egress filtering](https://en.wikipedia.org/wiki/Egress_filtering)
* [IP address spoofing](https://en.wikipedia.org/wiki/IP_address_spoofing)
* [Prefix hijacking](https://en.wikipedia.org/wiki/Prefix_hijacking)

References[[edit](https://en.wikipedia.org/w/index.php?title=Ingress_filtering&action=edit&section=6)]

* 1. [**^**](https://en.wikipedia.org/wiki/Ingress_filtering#cite_ref-1) Robert Gezelter (1995) *Security on the Internet* Chapter 23 in Hutt, Bosworth, and Hoytt (1995) "Computer Security Handbook, Third Edition", Wiley, section 23.6(b), pp 23-12, et seq.
  2. [**^**](https://en.wikipedia.org/wiki/Ingress_filtering#cite_ref-2) [*Ferguson, P.*](https://en.wikipedia.org/w/index.php?title=P._Ferguson_(Cisco_Systems)&action=edit&redlink=1)*; [Senie, D.](https://en.wikipedia.org/w/index.php?title=D._Senie_(Amaranth_Networks)&action=edit&redlink=1" \o "D. Senie (Amaranth Networks) (page does not exist)) (May 2000).*[*Network Ingress Filtering: Defeating Denial of Service Attacks which employ IP Source Address Spoofing*](https://tools.ietf.org/html/rfc2827)*.*[*IETF*](https://en.wikipedia.org/wiki/Internet_Engineering_Task_Force)*.*[*doi*](https://en.wikipedia.org/wiki/Doi_(identifier))*:*[*10.17487/RFC2827*](https://doi.org/10.17487%2FRFC2827)*. BCP 38.*[*RFC*](https://en.wikipedia.org/wiki/RFC_(identifier))[*2827*](https://tools.ietf.org/html/rfc2827)*. Retrieved 18 Feb 2014.*
  3. [**^**](https://en.wikipedia.org/wiki/Ingress_filtering#cite_ref-3) [*Baker, F.*](https://en.wikipedia.org/w/index.php?title=F._Baker_(Cisco_Systems)&action=edit&redlink=1)*; [Savola, P.](https://en.wikipedia.org/w/index.php?title=P._Savola_(CSC/FUNET)&action=edit&redlink=1" \o "P. Savola (CSC/FUNET) (page does not exist)) (March 2004).*[*Ingress Filtering for Multihomed Networks*](https://tools.ietf.org/html/rfc3704)*.*[*IETF*](https://en.wikipedia.org/wiki/Internet_Engineering_Task_Force)*.*[*doi*](https://en.wikipedia.org/wiki/Doi_(identifier))*:*[*10.17487/RFC3704*](https://doi.org/10.17487%2FRFC3704)*. BCP 84.*[*RFC*](https://en.wikipedia.org/wiki/RFC_(identifier))[*3704*](https://tools.ietf.org/html/rfc3704)*. Retrieved 18 Feb 2014.*
  4. [**^**](https://en.wikipedia.org/wiki/Ingress_filtering#cite_ref-4) *Barry Greene (June 11, 2012).*[*"Everyone should be deploying BCP 38! Wait, they are …"*](http://www.senki.org/everyone-should-be-deploying-bcp-38-wait-they-are/)*. senki.org.*

External links[[edit](https://en.wikipedia.org/w/index.php?title=Ingress_filtering&action=edit&section=7)]

* [RFC 2827](https://tools.ietf.org/html/rfc2827) - Network Ingress Filtering: Defeating Denial of Service Attacks which employ IP Source Address Spoofing (BCP 38)
* [RFC 3704](https://tools.ietf.org/html/rfc3704) Ingress Filtering for Multihomed Networks (BCP 84)
* *Jay R. Ashworth.*[*"BCP38.info"*](http://www.bcp38.info/)*.*
* [IETF's BCP Index](http://www.rfc-editor.org/bcp-index.html)

**Ingress Filtering**

Ingress filtering refers to the concept of firewalling traffic entering a network from an external source such as the Internet. In deployments with multi-WAN, the firewall has multiple ingress points. The default ingress policy on pfSense® software is to **block all traffic** as there are no allow rules on WAN in the default ruleset. Replies to traffic initiated from inside the local network are automatically allowed to return through the firewall by the state table.

**Egress Filtering**

Egress filtering refers to the concept of firewalling traffic initiated inside the local network, destined for a remote network such as the Internet. pfSense, like nearly all similar commercial and open source solutions, comes with a LAN rule allowing everything from the LAN out to the Internet. This isn’t the best way to operate, however. It has become the de facto default in most firewall solutions because it is what most people expect. The common misperception is “Anything on the internal network is ‘trustworthy’, so why bother filtering”?

**Why employ egress filtering?**

From our experience in working with countless firewalls from numerous vendors across many different organizations, most small companies and home networks do not employ egress filtering. It can increase the administrative burden as each new application or service may require opening additional ports or protocols in the firewall. In some environments it is difficult because the administrators do not completely know what is happening on the network, and they are hesitant to break things. In other environments it is impossible for reasons of workplace politics. The best practice is for administrators to configure the firewall to allow only the minimum required traffic to leave a network where possible. Tight egress filtering is important for several reasons:

**Limit the Impact of a Compromised System**

Egress filtering limits the impact of a compromised system. Malware commonly uses ports and protocols that are not required on most business networks. Some bots rely on IRC connections to phone home and receive instructions. Some will use more common ports such as TCP port 80 (normally HTTP) to evade egress filtering, but many do not. If access to TCP port 6667, the usual IRC port, is not permitted by the firewall, bots that rely on IRC to function may be crippled by the filtering.

Another example is a case we were involved in where the inside interface of a pfSense installation was seeing 50-60 Mbps of traffic while the WAN had less than 1 Mbps of throughput. There were no other interfaces on the firewall. Some investigation showed the cause as a compromised system on the LAN running a bot participating in a distributed denial of service (DDoS) attack against a Chinese gambling web site. The attack used UDP port 80, and in this network UDP port 80 was not permitted by the egress ruleset so all the DDoS was accomplishing was stressing the inside interface of the firewall with traffic that was being dropped. In this situation, the firewall was happily chugging along with no performance degradation and the network’s administrator did not know it was happening until it was discovered by accident.

The attack described in the above paragraph likely used UDP port 80 for two main reasons:

* UDP allows large packets to be sent by the client without completing a TCP handshake. With stateful firewalls being the norm, large TCP packets will not pass until the handshake is successfully completed, and this limits the effectiveness of the DDoS.
* Those who do employ egress filtering are commonly too permissive, allowing TCP and UDP where only TCP is required, as in the case of HTTP.

These types of attacks are commonly launched from compromised web servers. With a wide open egress ruleset, the traffic will go out to the Internet, and has the potential to overflow the state table on the firewall, cost money in bandwidth usage, and/or degrade performance for everything on the Internet connection.

Outbound SMTP is another example. Only allow SMTP (TCP port 25) to leave any network from a mail server. Or if a mail server is externally hosted, only allow internal systems to talk to that specific outside system on TCP port 25. This prevents every other system in the local network from being used as a spam bot, since their SMTP traffic will be dropped. Many mail providers have moved to using only authentication submission from clients using TCP port 587, so clients should not need access to port 25. This has the obvious benefit of limiting spam, and also prevents the network from being added to numerous black lists across the Internet that will prevent that site from sending legitimate e-mail to many mail servers. This may also prevent the ISP for that site from shutting off its Internet connection due to abuse.

The ideal solution is to prevent these types of things from happening in the first place, but egress filtering provides another layer that can help limit the impact if other measures fail.

**Prevent a Compromise**

Egress filtering can prevent a compromise in some circumstances. Some exploits and worms require outbound access to succeed. An older but good example of this is the Code Red worm from 2001. The exploit caused affected systems to pull an executable file via TFTP (Trivial File Transfer Protocol) and then execute it. A web server almost certainly does not need to use the TFTP protocol, and blocking TFTP via egress filtering prevented infection with Code Red even on unpatched servers. This is largely only useful for stopping completely automated attacks and worms as a real human attacker will find any holes that exist in egress filtering and use them to their advantage. Again, the correct solution to prevent such a compromise is to fix the network vulnerabilities used as an attack vector, however egress filtering can help.

**Limit Unauthorized Application Usage**

Many applications such as VPN clients, peer-to-peer software, instant messengers, and more rely on atypical ports or protocols to function. While a growing number of peer-to-peer and instant messenger applications will port hop until finding a port which is allowed out of the local network, many will be prevented from functioning by a restrictive egress ruleset, and this is an effective means of limiting many types of VPN connectivity.

**Prevent IP Spoofing**

This is a commonly cited reason for employing egress filtering, but pfSense automatically blocks spoofed traffic via pf’s *antispoof* functionality, so it isn’t applicable here. Preventing IP Spoofing means that malicious clients cannot send traffic with obviously falsified source addresses.

**Prevent Information Leaks**

Certain protocols should never be allowed to leave a local network. Specific examples of such protocols vary from one environment to another, but a few common examples are:

* Microsoft RPC (Remote Procedure Call) on TCP port 135
* NetBIOS on TCP and UDP ports 137 through 139
* SMB/CIFS (Server Message Block/Common Internet File System) on TCP and UDP port 445.

Stopping these protocols can prevent information about the internal network from leaking onto the Internet, and will prevent local systems from initiating authentication attempts with Internet hosts. These protocols also fall under [Limit the Impact of a Compromised System](https://docs.netgate.com/pfsense/en/latest/firewall/ingress-egress.html#limit-the-impact-of-a-compromised-system) as discussed previously since many worms have relied upon these protocols to function. Other protocols that may be relevant are syslog, SNMP, and SNMP traps. Restricting this traffic will prevent misconfigured network devices from sending logging and other potentially sensitive information out to the Internet. Rather than worry about what protocols can leak information out of a local network and need to be blocked, the best practice is to only allow the traffic that is required.

**Approaches for implementing egress filtering**

On a network that has historically not employed egress filtering, it can be difficult to know what traffic is absolutely necessary. This section describes some approaches for identifying traffic and implementing egress filtering.

**Allow what is known, block the rest, and work through the fallout**

One approach is to add firewall rules for known required traffic to be permitted. Start with making a list of things known to be required such as in Table [Egress Traffic Required](https://docs.netgate.com/pfsense/en/latest/firewall/ingress-egress.html#table-egress-traffic-required).

| *Egress Traffic Required* | | | |
| --- | --- | --- | --- |
| **Description** | **Source** | **Destination** | **Destination port** |
| HTTP and HTTPS from all hosts | LAN Network | Any | TCP 80 and 443 |
| SMTP from mail server | Mail Server | Any | TCP 25 |
| DNS queries from internal DNS servers | DNS Servers | Any | TCP and UDP 53 |

After making the list, configure firewall rules to pass only that traffic and let everything else hit the default deny rule.

**Log Traffic and Analyze Logs**

Another alternative is to enable logging on all pass rules and send the logs to a syslog server. The logs can be analyzed by the syslog server to see what traffic is leaving the network. pfSense uses a custom log format, so the logs typically need be parsed by a custom script unless the server has some knowledge of the pfSense filter log format. Analysis of the logs will help build the required ruleset with less fallout as it will yield a better idea of what traffic is necessary on the local network.