Firewall Best Practices

Christian Winter
Advisor: Cornelius Diekmann
Seminar Future Internet SS2016
Lehrstuhl Netzarchitekturen und Netzdienste
Fakultät für Informatik, Technische Universität München
Email: winterch@in.tum.de

ABSTRACT

With growing Internet usage in both private and commercial fields, filtering network traffic to prevent harmful use is gaining importance. But correctly filtering packets without restricting any wished functionality is complicated. New protocols are introduced and existing ones like ICMP and IP are updated. For each protocol, filtering rules have to be created separately. We summarize the most important best practices in firewall filtering for common protocols and give insights on why these filtering rules have to be applied. Additionally, we explain where information on creating individual policies can be found online and issues that need consideration are listed. The best practices summarized here are generic and can therefore be implemented for most firewall models. To give a concrete example, a filtering policy for a gateway firewall according to these generic best practices is given throughout this paper.

Keywords

Firewall, Best Practice, IP Filtering, ICMP Filtering, Packet Filtering

1 Introduction

Besides encryption and authentication, filtering network traffic in firewalls is an important and widely used measure to prevent attacks against online services, like online banking and shops. Therefore, creating and maintaining firewalls is a task network administrators have to face. A main part in firewall management is creating and maintaining a filtering table. Filtering tables can easily grow to several thousand lines of rules, as is evident from different rule sets collected for research [10]. Therefore, checking whether a firewall filters potentially harmful messages can be challenging. In addition, the IETF recommendations and best practices are spread over several different RFCs and often implicit in protocol standards. This makes it difficult for firewall administrators to ensure their firewall acts according to best practices. Network protocols and corresponding filtering policies are still a changing and widely researched field, evident from recently published articles and RFCs covering these topics [11, 14, 18]. This paper gives an overview over best practices for widely used protocols and summarizes them. In addition it lists where additional information on these best practices can be found and what attacks filtering accordingly can counter.

Section 2 gives recommendations on how to deal with certain address registries and port ranges and how spoofing of address registries can endanger the network. Furthermore it covers several aspects of the IP protocol family, like IPv6 extension headers and fragmentation, as well as attacks on those. Section 3 covers the dangers in the Internet Control Message Protocol (ICMP) in versions 4 and 6, lists possible attacks and gives recommendations which messages and types should be dropped in a firewall. Finally, Section 4 gives an outlook what else firewall administrators should consider, including IP tunnels, multicast messages, and other protocols. To give detailed filtering recommendations on some policy dependent best practices, this paper explains how a example firewall could be configured. We assume this firewall resides as a gateway between a local network and the Internet. Inside the protected network a web server should be reachable for HTTP requests from the Internet. This example will be used throughout the paper to show how filtering recommendations based on policies or services in the network could be implemented.

2 IP and Port Considerations

An often used method in packet filtering is deciding whether the packet may pass through the firewall or is dropped based on the source or destination addresses and the transport layer ports of a packet. Many of these rules are based on the policy of the network and dependent on what services should be available to and from the outside world. But there are best practices which should be implemented in every rule set, depending on the location of the firewall in a network. This section deals with these filtering best practices for different firewall locations.

2.1 Special Purpose IP Address Registries

Several IP address registries are reserved by the Internet Assigned Numbers Authority (IANA) for special purposes and therefore need additional consideration when implementing firewall rule sets. In this section we want to discuss the registries that should be filtered for different firewall use cases. A full list of all registries can be found in RFC 6890 [6]. There are some addresses that are not valid for use in any packet leaving a host. RFC 6890 [6] defines the address registries that are not valid as source or destination address for both IPv4 and IPv6. These registries have to be dropped in every firewall. In addition to those addresses, there are other restrictions for firewalls that are not limited to local-only traffic. When a firewall monitors traffic involving at least one interface connected to a network with a public address range, some other address registries are not valid for use in packets transiting this firewall. These address registries mainly consist of those reserved for private use and local

doi: 10.2313/NET-2016-09-1_01

Firewall type	DROP range	
All firewalls	127.0.0.0/8	192.0.2.0/24
	198.51.100.0/24	203.0.113.0/24
	240.0.0.0/4	(192.0.0.0/24)
Gateway firewalls	0.0.0.0/8	10.0.0.0/8
	100.64.0.0/10	169.254.0.0/16
	172.16.0.0/12	192.0.0.0/29
	192.168.0.0/16	198.18.0.0/15
	255.255.255.255/32	
Example firewall	192.88.99.0/24	224.0.0.0/4
	& gateway	

Table 1: IPv4 Filter Considerations

Firewall type	DROP range	
All firewalls	::1/128	2001:db8::/32
	2001:10::/28	(2001::/23)
Gateway firewalls	::/128	100::/64
	2001::/32	2001:2::/48
	fc00::/7	::ffff:0:0/96
	fe80::/10	without fe80::/64
Example firewall	64:ff9b::/96	2002::/16
	& gateway	

Table 2: IPv6 Filter Considerations

protocols. Table 1 and Table 2 list the address registries that should be blocked in these firewalls. The addresses in parentheses have to be dropped unless RFC 6890 [6] defines a more specific registry as valid. The registry of *fe80::/10* is not valid outside a local network, with exception of link local addresses *fe80::/64* used for the Neighbour Discovery Protocol (NDP) [29]. Additionally the tables show which registries will be dropped in our example firewall. For our firewall we don't want to allow non-global address registries. To keep the tables compact, the registries that have to be dropped for all gateway firewalls are not repeated for our example firewall, but have to be dropped as well. Furthermore we do not want to support protocols not essential to network communication, like addresses reserved for benchmarking. As discussed in Subsection 4.2 we are not depending on any multicast messages, so we drop the corresponding address registry in our firewall. Finally we do not want to support IPv4 to IPv6 address translation, so we also drop the registries reserved for translation.
To prevent attackers from inserting invalid packets into the network, e.g. packets with addresses that are reserved for local protocols, and thereby bypassing other security measures, all packets containing such an address as a source or a destination have to be dropped by the firewall.

2.2 Spoofing Considerations

Some Denial of Service (DoS) attacks use spoofed IP source addresses, addresses that were not assigned to the sender, making it hard to trace back this kind of attack [12].

2.2.1 Attack Example

An example for a spoofing attack is a variant of TCP SYN flooding described in RFC 2827 [12]. Thereby an attacker sends many TCP SYN packets to the victim using random spoofed source addresses. This results in a high load of TCP SYN/ACK packets to be sent out by the victim and additional a lot of TCP connections to store and track on the victim. This high load of packets might lead to a system crash or at least to a drop in overall performance for other connections.

2.2.2 Ingress Filtering

To counter this kind of attacks with spoofed packets to be started from an attacker inside a network to a remote victim, all routers of this network should implement ingress filters [12]. Spoofed packet attacks can only be fully countered if all networks are filtered accordingly to RFC 2827 [12], else an attacker in an unfiltered network could still carry out this attack. In our firewall, for example, it would be nearly impossible to tell if a packet arriving from the internet was really sent by the host in the source address because all remote hosts are connected to the same interface. On the other hand it is easily detectable if a packet trying to leave our network has a source address that was not assigned to a host inside the network. Even if all firewalls would implement ingress filters, RFC 2827 [12] states that spoofing an address from a host residing in the same network as the attacker, and therefore being a valid address at the router, will still be possible. An ingress filter should ensure that a packet entering the firewall has an IP address that resides behind the interface it entered through, thereby "prohibiting an attack[er] from using 'invalid' source addresses which reside outside of this prefix range."[12] A general filtering rule for ingress filtering can't be given because it is highly dependent on the interfaces of a firewall and the subnets residing behind these, which makes ingress filtering a complex task mainly for large firewall rule sets. A good way of ensuring that a firewall prohibits spoofed packets from passing through is the algorithm described by Diekmann et al. [11] which checks spoofing protection for a finished rule set, or to use a reverse path forwarding mechanism described in RFC 3704 [3]. Such a mechanism ensures that packets arriving at a firewall will only be forwarded if the firewall is on the route from the source specified in the packet to the destination. This is done by checking if a new packet to the source of the received packet would be sent via the interface the received packet arrived at. If this is not the case, the packet is dropped at the firewall [3].

2.3 IP Header Options

The Internet Protocol specifies header options to extend the protocol functionality. Those options mainly fulfil an important task in the routing of IP packets, but some are the basis for attacks. For IPv4, RFC 7126 [18] lists possible threats related to header options and what impact dropping packets containing them would have. We will only deal with some of the options listed there, but it is recommended that all of the options are reviewed.

2.3.1 Unknown IPv6 Extension Headers

Especially the extension headers of IPv6 pose a threat to network security as the IPv6 specification RFC 2460 [9] does not specify how packets with unknown options should be dealt with at firewalls along the path of the packet. But it specifies that "extension headers are not examined or processed by any node along a packet's delivery path, until the packet reaches the node [...] identified in the Destination Address field of the IPv6 header." [9] Because this prohibits middleboxes from checking for unknown header options, RFC 4942

doi: 10.2313/NET-2016-09-1_01

[7] states that this rule may be ignored by firewall administrators. Even if a middlebox firewall ignores the processing rule named above by checking extension headers and detects an unknown header option, it may be problematic to drop this packet because the firewall can not know if this unknown option is implemented in the target. The best trade-off is selecting firewalls which allow filtering based on IP extension header types used in a packet. This allows the firewall to drop unknown options and easily whitelist new header types when they are introduced [7].

2.3.2 Source Routing

IPv6 source routing, header type 43, can be used to specify a list of intermediate hosts along the path of a packet. At each of these intermediate hosts the destination address is set to the address of the next intermediate host or, at the last intermediate host, to the destination [9]. This header option, especially routing type 0, can be used for several different attacks, e.g. for bypassing filtering devices and bandwidth exhaustion [20]. Therefore RFC 5095 [1] deprecates the routing type 0 routing header (RH0) for IPv6. Packets containing a RH0 can be dropped in the firewall, but firewalls must not drop all packets containing a routing header, and forwarding packets with a type 43 header of other routing types must be permitted [1]. In our example firewall we want to drop IPv6 packets containing a RH0. Similar to the IPv6 RH0, IPv4 options 131 and 137 implement source routing. These types could be used to bypass firewall rules, to learn about the networks topology and to exhaust the bandwidth of a network and dropping them only has a small impact on troubleshooting [18]. Because of the small benefit of source routing compared to the dangers we want to drop all IPv4 packets containing a option type 131 or 137 in our example firewall, and it is suggested to do this in all firewalls.

2.3.3 Fragmentation

Another threat posed by IP header options are fragmentation attacks. This kind of attacks utilizes the vague regulations on filtering in the definitions of IPv4 and IPv6.

2.3.3.1 Overlapping Fragments

A common attack scenario is bypassing firewall checks by overriding protocol headers, e.g. the TCP header, with overlapping fragments. A detailed attack on IPv4 can be found in RFC 1858 [34] and on IPv6 in RFC 5722 [24]. Although the attack is very similar for both IP versions, the prevention measures differ. To prevent overriding transport-layer headers in IPv4, it should always be checked that the fragment offset field in the IP header is larger or equal to the length of the transport-layer header [34]. In IPv6, routers are not allowed to fragment packets and therefore fragmentation is managed by the fragment extension header, defined in Section 4.5 of RFC 2460 [9]. Because of this different fragmentation approach, a more general filtering method has to be applied. To counter an overlapping fragment attack in IPv6, when two fragments overlap, the whole packet, including all fragment packets arriving later on, must be silently discarded [24]. To this rule there is one exception. Atomic fragments are packets containing a fragment header without being split into multiple packets. Such packets are sent when an IPv6 host received a ICMPv6 Code 2 messages stating that along the packet path a section has a MTU smaller then the guaranteed IPv6 minimum MTU of 1280 [17]. To allow IPv6 hosts to use this option, RFC 6946 [17] updates RFC 5722 [24] to allow certain packets through the firewall that would have been filtered by the original filtering policy stated above. Packets with no fragment offset and the "more flag" set to 0, atomic fragments, have to be processed independently of any other packets. This means packets with the same source and destination address and fragment identification are not to be seen as overlapping when they are atomic fragments and therefore must not be dropped [17].

2.3.3.2 Tiny Fragments

Another attack against IPv6 using fragmentation is a DoS attack using unnecessary tiny fragments without a terminating packet, a packet where the "More Fragments" flag is set to 0. This could lead to an overload of fragment buffers and drain resources of the target machine. To prevent such an attack a firewall should always check that packets containing non-final fragments are at least half the size of the protocols guaranteed minimum MTU, in the case of IPv6 640 octets [7]. This way no correctly fragmented packet is dropped while the number of packets an attacker can send through the firewall is minimized, thereby relieving the fragmentation buffer. Another attack utilizes the small minimum MTU in IPv4. This allows the fragment size to be chosen small enough to force some TCP header fields into the second fragment. This prohibits TCP filter rules from matching the header parts in the second fragment and thereby might allow the packet through the firewall [34]. In addition there is an attack combining both IPv4 attacks described here. This attack combines tiny and overlapping fragments to bypass security measures preventing the overlapping fragment attack or the tiny fragment attack alone [28]. The countermeasure for this attack can be used to eliminate all other fragmentation attacks against IPv4 listed here. It must be ensured that a firewall checks that all the header fields relevant for filtering must be in the first fragment and that, after the first packet, no fragment offset value small enough to override relevant header fields is allowed [28]. In the example firewall we want to have every TCP header field up to the "Window Size" (Bytes 14 and 15) for filtering. Therefore we request the first fragment to carry at least 16 Bytes of payload and that there is no packet after the first with a fragment offset smaller than 2.

2.4 Application Ports

In addition to the special IP address registries described in Subsection 2.1, there are some well-known application ports that need special consideration. Although all of these ports serve a well intended purpose, the applications associated with these ports have been the target of successful attacks and therefore should be blocked in the firewall. The ports and applications listed are the most common examples of such vulnerable ports but additional vulnerabilities might come up in the future and may be added to the block list.

2.4.1 Network Management Protocols

Some protocols for managing hosts over the network have vulnerabilities that attackers could utilize in attacks against victims running such a protocol. The most prominent network management protocols with weaknesses are listed in

doi: 10.2313/NET-2016-09-1_01

this section and it is explained how and why to block them. Microsoft's implementation for Remote Procedure Calls (RPC), described by Microsoft's TechNet [27], is an interprocess communication technique for client/server architectures. The implementation includes various vulnerabilities, some exploited by the MSBlaster worm. To prevent such attacks, the corresponding TCP and UDP port 135 should be blocked by a firewall.

The Simple Network Management Protocol (SNMP), as described in RFC 1157 [5], is an IETF standard protocol used to gather and manage information on network devices running on UDP ports 161 and 162 as defined in RFC 3417 [32]. For SNMP multiple vulnerabilities at least in the versions 1, 2 and 3 are known, some of which can lead to DoS attacks and are open to format string and buffer overflow attacks [22]. To prevent attacks on SNMP, it is suggested by Jiang [22] to block all ports related to SNMP in the firewall to prevent attacks from outside the network. When it is required to access SNMP from outside the network, the access to these ports should be restricted to a limited IP address range, e.g. by using a VPN and creating an exception only for VPN traffic.

The Intelligent Platform Management Interface (IPMI) is a widely used standard for extensible, scalable interoperable server hardware management architecture independent of the server's power status. But the architectures protocol has vulnerabilities that could be utilized in attacks against the network, e.g. probing for other devices in the network, as pointed out by Gasser et al. [14]. To counter these flaws, Gasser [14] suggests to drop UDP port 623 used in IPMI in a firewall or to restrict the access to IPMI devices to a VPN and drop all other packets in the firewall.

2.4.2 Filtering Recommendation

For our own firewall we also want to drop additional TCP/IP application layer protocols. For example Microsoft NetBIOS and SMB do not fulfil functionality we need to access from outside our firewall, so we want to drop the corresponding ports. In addition to these ports, we also want to block the telnet protocol used for remotely controlling hosts over the Internet. This protocol runs on TCP port 23 and allows to take full control over a host while transmitting everything unencrypted [31]. Therefore it would allow attackers to intercept communication and take over hosts in our network. To prevent these attacks we drop the corresponding port. Most of the vulnerabilities attributed to applications mentioned can be prevented by blocking the corresponding ports in the firewall. To simplify firewall configuration we want to whitelist only port 80 in our firewall. This way all other ports are dropped by the firewall and we don't have to deal with the potentially dangerous ports separately. If a firewall configuration using blacklisting is wished, Table 3 lists the ports discussed in this section. But because there might be other security issues in other applications not listed in this paper it is advised to use a whitelisting strategy, or at least to update the rule set accordingly when another flaw is detected.

Port	Protocol	Application	Recommendation
23	TCP	Telnet	DROP
135	TCP	Microsoft RPC	DROP
135	UDP	Microsoft RPC	DROP
137	UDP	NetBIOS	DROP
137	TCP	NetBIOS	DROP
138	UDP	NetBIOS	DROP
139	UDP	NetBIOS	DROP
161	UDP	SNMP	DROP
162	UDP	SNMP	DROP
445	TCP	SMB	DROP
623	UDP	IPMI	DROP

Table 3: Port filtering recommendations

3 ICMP and ICMPv6

The Internet Control Message Protocol (ICMP) plays an important part in upholding Internet communication. With the introduction of IPv6, the corresponding ICMPv6, gained even more importance in communication. It handles error notification in the Internet Protocol and therefore must be implemented on every IP module [30]. But some of these control messages can be used for malicious intent and therefore several ICMP messages should be filtered in a firewall while others must not be blocked. This section deals with ICMPv4 and ICMPv6 message filtering along with possible attacks against ICMP and suggests a concrete filtering policy.

3.1 ICMPv4

During the lifespan of IPv4, different control messages have been introduced and others have been deprecated again already. This section deals with summarizing the essential ICMP messages and which should be dropped because they do not fulfil a valid purpose any more. A blogpost by John Albin [2] summarizes all common ICMPv4 messages and gives recommendations in filtering them. This, of course, is no scientifically accurate source, but it does not violate any RFC that was released until the publishing of this post in 2005. Later on, one of the messages marked essential, ICMP source quench, was deprecated and will be dealt with later on, all other recommendations are still valid today. Table 4 summarizes the filtering recommendations for our example firewall. In the following part we want to take a look at a possible classification of some ICMPv4 message types and how they should be dealt with.

3.1.1 Essential Messages

There are messages which have to be passed and accepted by all IPv4 hosts to enable communication. Albin [2] lists message type 3, destination unreachable, as essential to communication. ICMPv4 type 3 messages should be allowed through the firewall, because without these ICMP messages, communication is impossible under certain circumstances. When a datagram must be fragmented to be forwarded, because the network MTU is too small and the packet is not allowed to be fragmented, the packet has to be discarded and a "destination unreachable" error message may be sent [30]. A host dropping all ICMPv4 type 3 messages will not notice unfragmentable packets sent are too big and therefore will not be able to reach the destination.

doi: 10.2313/NET-2016-09-1_01

3.1.2 Deprecated Messages

Some ICMPv4 messages are not relevant in today's network communication and therefore have been deprecated. The ICMPv4 source quench mentioned in Subsection 3.1 is one of these messages. It is not to be used for congestion control because it is known to be unfair and ineffective [16]. In addition to this message, RFC 6918 [19] deprecates other message types and explains why they are not relevant any longer. These messages should not only be dropped because they are deprecated, but also because they might be used in attacks. The source quench message, for example, can be used to permanently slow down IP connections in both IPv4 and IPv6 and therefore slowing down hosted services [15]. Table 4 summarizes all the deprecated messages in the drop recommendations.

3.1.3 Potentially Dangerous Messages

In addition to the types deprecated, there are other types that firewalls might want to filter. The ICMPv4 types 0 and 8 used for ping and type 11 used for traceroute are not essential to communication as stated by Albin [2] but are used widely for host location and troubleshooting. If one doesn't wish the hosts in his network to be located by others it is suggested to block those message types. The message types 13 to 16 give additional information about the addressed host that attackers could use and Albin [2] suggests to drop these messages, therefore our example firewall will drop them, as well as all other ICMPv4 types not listed. We will however allow ping and traceroute for troubleshooting purposes.

3.2 ICMPv6

As stated in the introduction to this section, ICMPv6 gained importance in the IP communication compared to the ICMPv4 protocol. It is essential for IPv6, taking over many functions in error handling and information distribution, and therefore includes many different message types and options. Because it is a lot more extensive than ICMPv4, filtering is more difficult and requires a more detailed approach. To enable more detailed filtering depending on the state of connections, RFC 4890 [8] suggests to enable stateful packet filtering where possible. This allows the firewall to determine whether an ICMP error message arriving at the firewall corresponds to a sent packet and therefore is legit. For this to be possible, RFC 4890 [8] states that the firewall must be able to perform deep packet inspection on these error messages. This section will discuss ICMPv6 traffic passing through the firewall, but packets addressed to the firewall itself should be filtered as well. Table 6 suggests filtering for this traffic, detailed information on why it should be filtered can be found in RFC 4890 [8]. In the Tables 5 and 6 the terms "consider dropping" and "policy dependent" refer to the corresponding lines in the table. The types listed there will be dropped in our example firewall when referred, but are not repeated for reasons of clarity. Furthermore RFC 4890 [8] includes an example script for configuring an iptables firewall according to most of the best practices in Appendix B of the RFC, useful for quick configuration according to these guidelines.

3.2.1 Essential Messages

There are message types essential to communication, like messages for error detection and notification when a host is unreachable. Dropping these messages will prevent or severely impact communication establishment and maintenance, therefore these messages must not be dropped [8]. RFC 4890 [8] states the types 1, 2 as essentials, aswell as the ICMP code 0 of type 3 and codes 1 and 2 for type 4. Furthermore the types 128 and 129 are essential to Teredo, discussed in Subsubsection 4.1.1, and should be allowed because they pose no imminent threat in IPv6 port scanning. In our firewall we therefore allow all these message types to pass.

Action	Types
ALLOW	3, 0, 8, 11
DROP	4, 6, 13-18, 30-37
DROP when not needed	other types

Table 4: ICMPv4 filtering recommendations

3.2.2 Important Messages

In addition to the essential messages there are ICMPv6 messages that fulfil an important task in IP connections. These messages should not be dropped unless there is a severe reason. The type 3 code 1, type 4 code 0 and types 144 - 147 for mobile IPv6 should be allowed to pass the firewall [8]. Because we do not want to support mobile IPv6 in our example network, we will not allow the latter through the firewall.

3.2.3 Policy Dependent Messages

The broad variety of ICMPv6 messages includes many that are only essential to certain protocols as well as unallocated types. Therefore a policy has to be defined on how to deal with these messages. Messages from unallocated areas that only transit the firewall and do not end on a host inside the protected network should be able to pass the firewall. These types might be implemented on other hosts and already in use and dropping them could cause damage to this connection. For the site the firewall is protecting, administrators might want to choose dropping these messages until they are allocated to prevent covert channels. A part of these messages might become essential in the future therefore the firewall policy should be updated regularly to include such allocations [8]. In addition to those solely dependent on policy, RFC 4890 [8] lists other messages that should be dropped in most cases. ICMPv6 includes messages meant only for local information exchange that should not be able to leave the local network and therefore should not occur outside a local scope. Table 5 includes all such messages in the drop section, a more detailed classification can be found in RFC 4890 [8]. Our firewall will drop all these policy dependent messages, as it is located on the outside of the local scope and we do not want to support any experimental protocols.

3.2.4 Possible Attacks

With the variety of ICMPv6 messages come several possible attacks. These include man in the middle attacks on protocols, probing, and different Denial of Service attacks. In addition ICMP error messages can be used to establish a covert channel through error type payloads [8]. To counter these attacks, very detailed filtering of ICMP traffic in the firewall is essential.

Action	Types (Codes)
ALLOW	1, 2, 3 (0), 4 (1,2), 128, 129
Consider allowing	3 (1), 4 (0), 144-147
Policy dependent	15, 5-99, 102-126 154-199, 202-254
Consider dropping	100, 101, 127 138-140, 200, 201, 255
DROP addressed to example network	5-99, 102-126, 144-147 150, 154-199, 202-254 & consider dropping

Table 5: ICMPv6 filtering recommendations passing the firewall

Action	Types (Codes)
ALLOW	1, 2, 3 (0), 4 (1,2), 128, 129 130-136, 141-143, 148, 149 151-153
Consider allowing	3 (1), 4 (0), 144-147, 150
Policy dependent	4-99, 102-126, 137, 139, 140
Consider dropping	100, 101, 127 154-199, 200-255
DROP in example	144-147, 150 policy dependent & consider dropping

Table 6: ICMPv6 filtering recommendations addressed to the firewall

4 Other Considerations

This document focuses on the most common and important firewall best practices, but considerations should be made in other fields of network traffic that are left out in this paper. Some of those network issues are introduced briefly in this section.

4.1 IP Tunnels

IP tunnels are often used to encrypt IP payload or to support protocols not transmittable over IPv4. These tunnels pose a potential threat for networks as firewalls are not always able to check all of the contained data [25]. Therefore firewall administrators should consider if they want to accept packets using tunnel mechanisms in their network.

4.1.1 IPSec and Teredo

The most prominent implementations of IP tunnels are IPSec [23] and Teredo [21]. IPSec enables encryption and integrity protection of IP packets, providing confidentiality of the payload during transport. But for firewalls, this means they can not know what is transported in the payload, therefore anything could be transported through the firewall in an Encapsulating Security Payload (ESP) packet. To ensure that IPSec connections can take place, middlebox firewalls should always allow IPSec traffic, for end point firewalls a more differentiated handling is required. Teredo tackles problems that occur when trying to access IPv6 networks residing behind NAT devices by tunnelling these packets via UDP. It is possible that packet filters do not recognise that there is another IP datagram contained in the packet and therefore might not check this inner packet [25]. More detailed considerations to these and other problems of IP Tunnelling can be found in RFC 6169 [25].

4.1.2 6to4

A special case of IP Tunnels are 6to4 tunnels that are used as a IPv6 interim mechanism to connect IPv6 networks over IPv4 clouds described in RFC 3056 [4]. The transmission protocol has weaknesses that can lead to DoS attacks and service theft, where an attacker uses a service he is not authorized to use [33]. RFC 3964 [33] discusses a variety of attacks against 6to4 communication and provides suggestions on fixing some of these flaws.

4.2 Multicast

A network administrator should always consider if and how they want their network to be reachable with multicast messages. A general recommendation whether to filter multicasts or not cannot be given because it highly depends on the policy of the local network. Depending on this policy, it might be feasible to drop some or all multicast packets at the firewall or allow communication with the inner network. Most intranets will not need external multicast at all, and others might only need some of these messages. RFC 2588 [13] gives recommendations how firewalls should and can control the traversal of multicast packets and what else firewall administrators should consider when dealing with multicast messages for a gateway firewall. Our example firewall will drop all multicast packets as they do not fulfil a purpose we need in our local network.

4.3 Other Protocols

Although this paper mainly focuses on TCP and UDP in the transport layer and IP and ICMP in the network layer, firewall administrators should be aware that there are other network protocols that a firewall can stumble upon. An example for a protocol not discussed here is the Routing Information Protocol (RIP) used for distance vector routing [26]. A whitelisting firewall would normally drop packets using unknown protocols because it does not have a rule allowing theses datagrams. But rule sets accepting all packets from a certain source address will allow these datagrams through the firewall, which might have impact on the networks security. Even though these protocols are not widely used and might not pose an actual threat, administrators should be aware that other protocols exist.

5 Conclusion

To give a detailed overview over the most important best practices, we reviewed dozens of RFCs and other sources and summarized the filtering recommendations they give for several aspects of different transport and network layer protocols. In addition we show what attacks can be prevented by implementing these best practices and what liabilities might come with these implementations. We learned that some aspects of these best practices are dependent on local policies and the type of firewall used. Furthermore we showed how these policy dependent implementations could be done for a example firewall we defined.
This paper lists the most important best practices available at the moment, but other aspects of firewall calibration have to be considered as well. The best practices listed here should be updated and extended accordingly if new vulnerabilities are found or a new protocol is introduced.

6 References

[1] J. Abley, P. Savola, and G. Neville-Neil. Deprecation of Type 0 Routing Headers in IPv6. RFC 5095 (Proposed Standard), Dec. 2007.

[2] J. Albin. Everything you ever wanted to know about ICMP, (but were afraid to ask rfc792), 2005 (accessed March 17, 2016). http://john.albin.net/essential-icmp.

[3] F. Baker and P. Savola. Ingress Filtering for Multihomed Networks. RFC 3704 (Best Current Practice), Mar. 2004.

[4] B. Carpenter and K. Moore. Connection of IPv6 Domains via IPv4 Clouds. RFC 3056 (Proposed Standard), Feb. 2001.

[5] J. Case, M. Fedor, M. Schoffstall, and J. Davin. Simple Network Management Protocol (SNMP). RFC 1157 (Historic), May 1990.

[6] M. Cotton, L. Vegoda, R. Bonica, and B. Haberman. Special-Purpose IP Address Registries. RFC 6890 (Best Current Practice), Apr. 2013.

[7] E. Davies, S. Krishnan, and P. Savola. IPv6 Transition/Co-existence Security Considerations. RFC 4942 (Informational), Sept. 2007.

[8] E. Davies and J. Mohacsi. Recommendations for Filtering ICMPv6 Messages in Firewalls. RFC 4890 (Informational), May 2007.

[9] S. Deering and R. Hinden. Internet Protocol, Version 6 (IPv6) Specification. RFC 2460 (Draft Standard), Dec. 1998. Updated by RFCs 5095, 5722, 5871, 6437, 6564, 6935, 6946, 7045, 7112.

[10] C. Diekmann. net-network. GitHub repository https://github.com/diekmann/net-network, 2014 (accessed March 20, 2016; last commit March 18, 2016).

[11] C. Diekmann, L. Schwaighofer, and G. Carle. Certifying spoofing-protection of firewalls. In *Network and Service Management (CNSM), 2015 11th International Conference on*, pages 168–172. IEEE, 2015.

[12] P. Ferguson and D. Senie. Network Ingress Filtering: Defeating Denial of Service Attacks which employ IP Source Address Spoofing. RFC 2827 (Best Current Practice), May 2000. Updated by RFC 3704.

[13] R. Finlayson. IP Multicast and Firewalls. RFC 2588 (Informational), May 1999.

[14] O. Gasser, F. Emmert, and G. Carle. Digging for dark IPMI devices: Advancing BMC detection and evaluating operational security. *Traffic Monitoring and Analysis*, 2016.

[15] F. Gont. ICMP Attacks against TCP. RFC 5927 (Informational), July 2010.

[16] F. Gont. Deprecation of ICMP Source Quench Messages. RFC 6633 (Proposed Standard), May 2012.

[17] F. Gont. Processing of IPv6 "Atomic" Fragments. RFC 6946 (Proposed Standard), May 2013.

[18] F. Gont, R. Atkinson, and C. Pignataro. Recommendations on Filtering of IPv4 Packets Containing IPv4 Options. RFC 7126 (Best Current Practice), Feb. 2014.

[19] F. Gont and C. Pignataro. Formally Deprecating Some ICMPv4 Message Types. RFC 6918 (Proposed Standard), Apr. 2013.

[20] J. Hui, J. Vasseur, D. Culler, and V. Manral. An IPv6 Routing Header for Source Routes with the Routing Protocol for Low-Power and Lossy Networks (RPL). RFC 6554 (Proposed Standard), Mar. 2012.

[21] C. Huitema. Teredo: Tunneling IPv6 over UDP through Network Address Translations (NATs). RFC 4380 (Proposed Standard), Feb. 2006. Updated by RFCs 5991, 6081.

[22] G. Jiang. Multiple vulnerabilities in SNMP. In *Computer*, volume 35, pages 2–4. IEEE, Apr 2002.

[23] S. Kent and K. Seo. Security Architecture for the Internet Protocol. RFC 4301 (Proposed Standard), Dec. 2005. Updated by RFCs 6040, 7619.

[24] S. Krishnan. Handling of Overlapping IPv6 Fragments. RFC 5722 (Proposed Standard), Dec. 2009. Updated by RFC 6946.

[25] S. Krishnan, D. Thaler, and J. Hoagland. Security Concerns with IP Tunneling. RFC 6169 (Informational), Apr. 2011.

[26] G. Malkin. RIP Version 2. RFC 2453 (Internet Standard), Nov. 1998. Updated by RFC 4822.

[27] Microsoft TechNet. What Is RPC?: Remote Procedure Call (RPC), Last updated 2016 (accessed March 17, 2016). https://technet.microsoft.com/en-us/library/cc787851(v=ws.10).aspx.

[28] I. Miller. Protection Against a Variant of the Tiny Fragment Attack (RFC 1858). RFC 3128 (Informational), June 2001.

[29] T. Narten, E. Nordmark, W. Simpson, and H. Soliman. Neighbor Discovery for IP version 6 (IPv6). RFC 4861 (Draft Standard), Sept. 2007. Updated by RFCs 5942, 6980, 7048, 7527, 7559.

[30] J. Postel. Internet Control Message Protocol. RFC 792 (Internet Standard), Sept. 1981. Updated by RFCs 950, 4884, 6633, 6918.

[31] J. Postel and J. Reynolds. Telnet Protocol Specification. RFC 854 (Internet Standard), May 1983. Updated by RFC 5198.

[32] R. Presuhn. Transport Mappings for the Simple Network Management Protocol (SNMP). RFC 3417 (Internet Standard), Dec. 2002. Updated by RFCs 4789, 5590.

[33] P. Savola and C. Patel. Security Considerations for 6to4. RFC 3964 (Informational), Dec. 2004.

[34] G. Ziemba, D. Reed, and P. Traina. Security Considerations for IP Fragment Filtering. RFC 1858 (Informational), Oct. 1995. Updated by RFC 3128.