

A Pattern Language for Firewalls

Eduardo B. Fernandez, Maria M. Larrondo-Petrie, Naeem Seliya, Nelly Delessy-Gassant,
and Markus Schumacher*

Dept. of Computer Science and Eng.
Florida Atlantic University
Boca Raton, FL 33431

* SAP Germany

Introduction

Information security is a growing need for organizations and individuals; therefore, computer systems must be protected against attacks [Fer01]. In the case of computers connected in a local network, attacks may come from hosts in external networks or in other local sub-networks. Network traffic has a layered structure and we need to protect against attacks that may come through any layer; this means that we need different types of defensive structures. Accessing an untrusted site is a risk and we also need to protect the traffic going out of a local network. A common solution to protect local networks is to incorporate a firewall to filter unwanted traffic [Zwi00].

Firewalls have been shown to be very effective in providing security by basically creating a choke point of entry (and exit) into a local network [Bar99]. A firewall therefore restricts unauthorized clients from access to the local network and local networks from accessing external sites that are considered untrustworthy. A firewall can be used as a mechanism to enforce security policies and also allows a limited exposure of the protected network to outsiders.

Several types of firewall exist that represent tradeoffs between complexity, speed, and security and which are tailored to control attacks in specific layers of the network. We present a pattern language to describe different types of firewalls. This language can be used as a guide to select a suitable type for a system or to help designers build new firewalls. Figure 1 shows the patterns in our language and their relationships and dependencies. The Packet Filter Firewall defines a basic filtering function at the IP layer based on packet inspection, typically of network addresses. The Proxy-Based Firewall is used at the network application layer to control access to application services and may be combined with the Packet Filter. Both the Packet Filter and the Proxy-Based firewalls can be complemented with Stateful Firewalls, where the state of the connection is also used to decide access. There are also Application firewalls, intended for filtering user application inputs; these are discussed elsewhere [De104].

A firewall is implemented as a piece of software or a combination of software and hardware that enforces an access control policy between networks. The basic underlying architectures of the various types of firewalls are similar and we try to capture here their generic structures, leaving out implementation details.

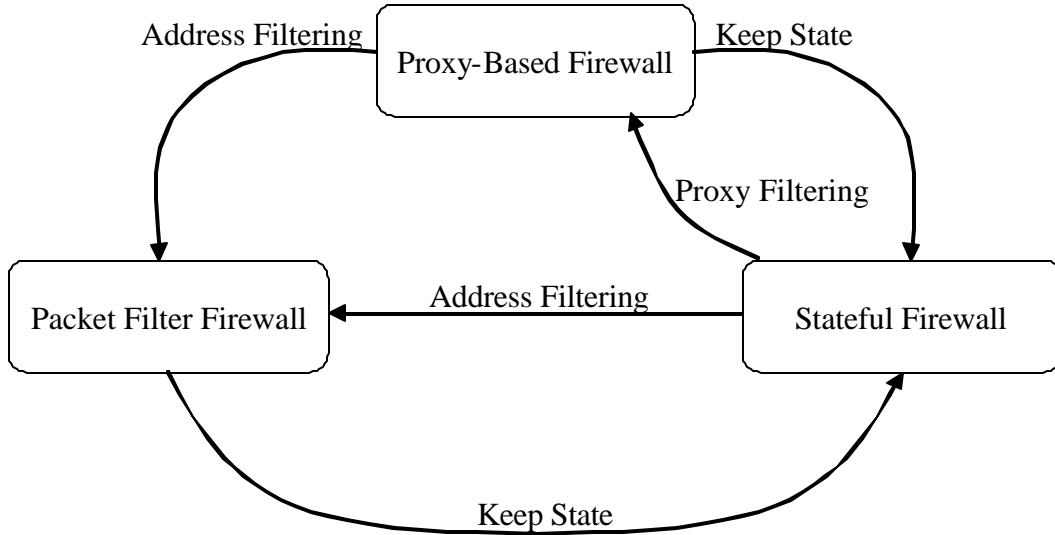


Figure 1: Firewall Pattern Language

Packet Filter Firewall

The Packet Filter Firewall filters incoming and outgoing network traffic in a computer system based on packet inspection at the IP level.

Example

Our system has been attacked recently by a variety of hackers, including somebody who penetrated our operating system and stole our clients' credit card numbers. Our employees are wasting time at work by looking at inappropriate sites in the Internet. If we continue like this we will be out of business quite soon.

Context

Computer systems on a local network connected to the Internet and to other networks with different levels of trust. A host in a local network receives and sends traffic to other networks. This traffic has several layers or levels. The most basic level is the IP level, made up of packets consisting of headers and bodies (payloads). The headers include the source and destination addresses as well as other routing information, the bodies include the message payloads.

Problem

Some of the hosts in other networks may try to attack the local network through their IP-level payloads. These payloads may include for example viruses or application-specific attacks. We need to identify and block those hosts.

The possible solution is constrained by the following forces:

- We need to communicate with other networks so isolating our network is not an option. However, we do not want to take a high risk for doing so.
- The protection mechanism should be able to reflect precisely the security policies of the institution. A too coarse defense may not be useful.
- Any protection mechanism should be transparent to the users. Users should not need to perform special actions to be secure.
- The cost and overhead of the protection mechanism should be relatively low or the system may become too expensive to run.
- Network administrators deploy and configure a variety of protection mechanisms; hence it is important to have a clear model of what is being protected.
- The attacks are constantly changing; hence it should be easy to make changes to the configuration of the protection mechanism.
- It may be necessary to log input and/or output requests for auditing and defense purposes.

Solution

A Packet Filter Firewall intercepts all traffic at a single point and inspects packets. Those coming from or going to untrusted addresses are rejected. The untrusted addresses are determined from a set of rules that implement the security policies of the institution. A client from another network can only access the Local Network if a rule exists authorizing traffic from its address. Specific rules may indicate an address or a range of addresses. Most commercial products order these rules for efficiency in checking. Additionally, if a request is not satisfied by any of the Explicit Rules, then a Default Rule is applied.

Structure

Figure 2 shows an **External Host** requesting access to a **Local Host** (a server), through a **Packet Filter Firewall**. The institution policies are embodied in the objects of class **RuleBase**. The rules in this set are ordered and can be **Explicit** or **Default**.

Dynamics

We describe the dynamic aspects of the Packet Filter Firewall using sequence diagrams that correspond to two of its two basic use cases. There is a use case symmetric to Filtering a client's request, this is Filtering an outgoing request, which we omit for brevity.

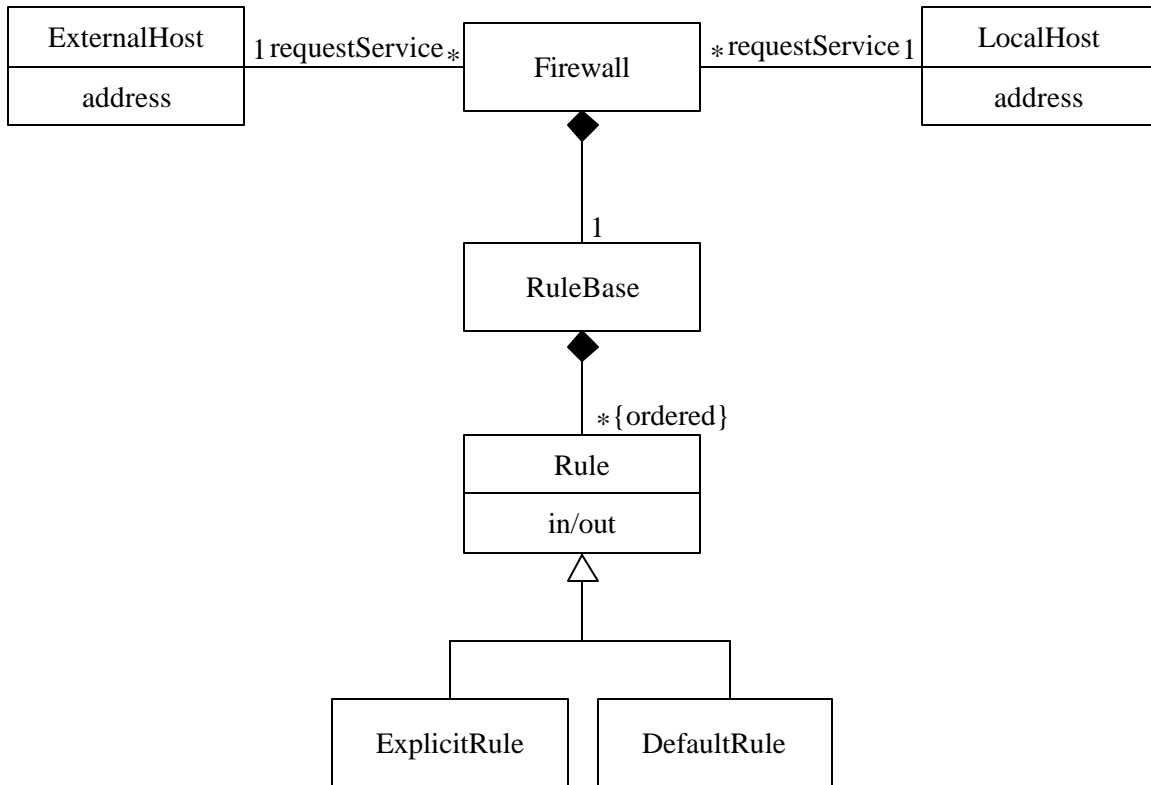


Figure 2: Class diagram for Packet Filter Firewall pattern

Filtering a Client's Request

- **Summary:** A host in a remote network wants access to a local host to either transfer or retrieve information. The access request is made through the firewall, which according to its set of rules determines whether to accept or deny the request, i.e., it filters the access request. Figure 3 illustrates this use case.
- **Actors:** A host in an external network (client).
- **Precondition:** An existing set of rules to filter the request must be in place in the firewall.
- **Description:**
 - a. An external host requests access to the local host.
 - b. A firewall filters the request according to a set of ordered rules. If none of the rules in the rule set are satisfied then a default rule is used for filtering the request.

- c. If the request is accepted, the firewall allows access to the local host.
- Alternate Flow: The request is denied.
- Postcondition: The firewall has accepted the access of a trustworthy client to the local host.

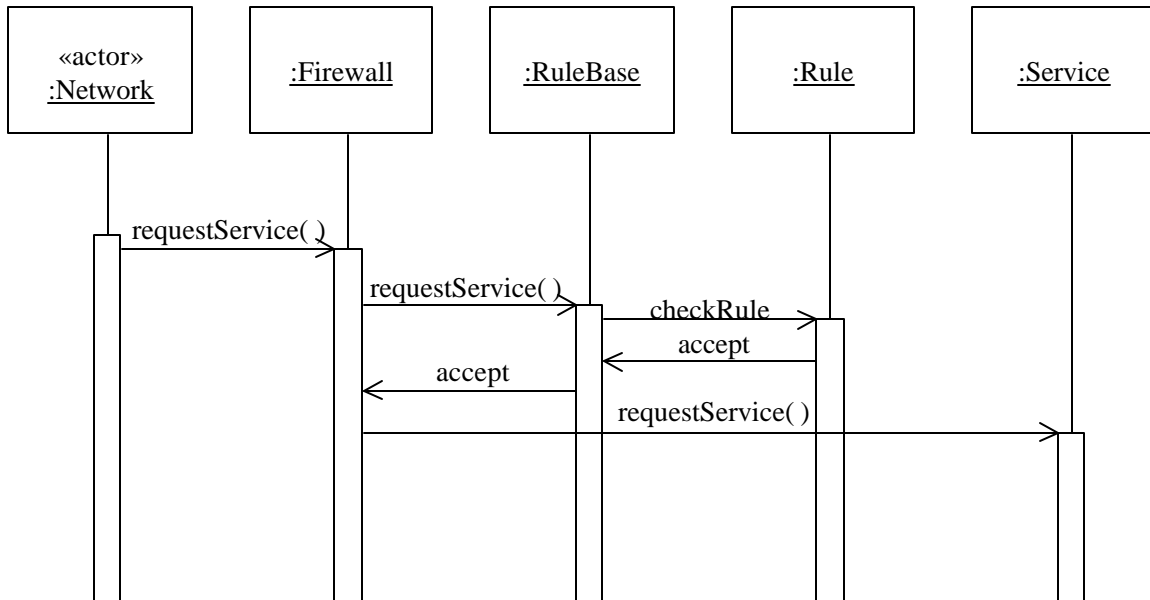


Figure 3: Sequence diagram for filtering a client's request

Defining a new rule

- Summary: The administrator of the local network adds a new rule to the set of rules. The firewall checks whether the new rule to be added does not already exist in the rule set. Figure 4 illustrates this use case.
- Actors: Administrator .
- Precondition: The administrator must have authorization to add rules.
- Description:
 - a. The administrator initiates the adding of a new rule.
 - b. If the rule does not already exist in the rule set then it is added.
 - c. The firewall acknowledges the addition of the new rule.
- Alternate Flow: The rule is not added because it already exists in the rule set.

- Postcondition: A new rule has been added to the rule set of the firewall.

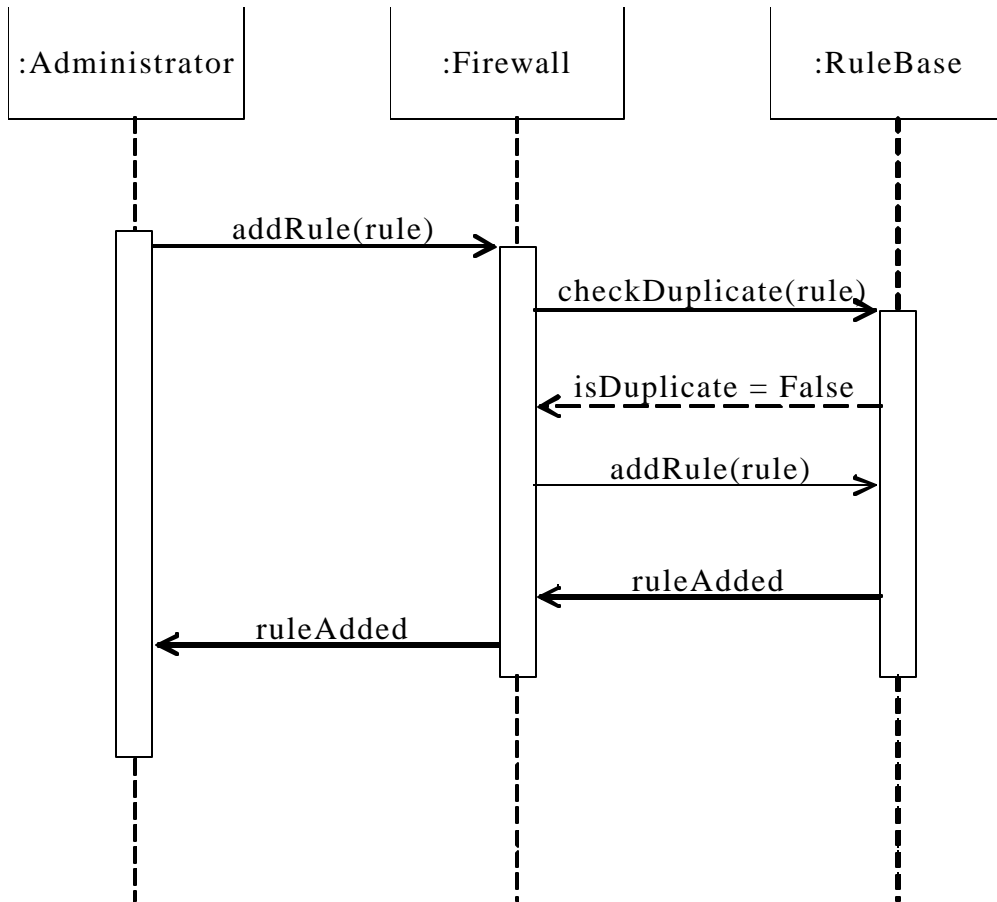


Figure 4: Sequence diagram for defining a new rule

Implementation

1. Define an institution policy about network access, classifying sites according to their trust.
2. Convert this policy into a set of access rules. This can be done manually, which may be complex for large systems. An alternative is using an appropriate commercial product, such as Solsoft [Sol].
3. Note that the idea of a single point of access is virtual, there may be several physical firewalls deployed at different places. This means it is necessary to install firewalls at all external boundaries (routers or gateways).
4. Write the rules in each firewall. Again, products such as Solsoft automatically propagate the rules to each registered firewall.

5. Configure the corresponding firewalls according to standard architectures. A common deployment architecture is the Demilitarized Zone (DMZ) [Dys04, Lon04].

Example resolved

We were able to trace the addresses of our attackers and we got a firewall to block requests from those addresses from reaching our system. We also made a list of addresses of inappropriate sites and blocked access to them from the hosts in our network.

Consequences

The Packet Filter Firewall Pattern has the following advantages:

- A firewall transparently filters all the traffic that passes through it, thus lowering the risk of communicating with potentially hostile networks.
- It is possible to express the institution filtering policies through its filtering rules, with different levels of protection for different parts of the network.
- It is easy to update the rule set to counter new threats.
- Because it intercepts all requests, a firewall allows systematic logging of incoming and outgoing messages. Because of this, a firewall facilitates the detection of possible attacks and helps to hold local users responsible of their actions when interacting with external networks.
- Low cost, it is included as part of many operating systems and simple network devices such as routers.
- Good performance. It only needs to look at the headers of IP packets, not at the complete packet.
- Can be combined with Intrusion Detection Systems (IDS) for greater effectiveness. In this case, the IDS can tell the firewall to block suspicious traffic.

The Packet Filter Firewall Pattern has the following (possible) liabilities:

- The firewall's effectiveness may be limited due to its rule set (order of precedence). Addition of new rules may interfere with existing rules in the rule set; hence, a careful approach should be taken in adding and updating access rules.
- The firewall can only enforce security policies on traffic that goes through the firewall. This means that one must make changes to the network to ensure that there are no other paths into its hosts.
- An IP-level firewall cannot stop attacks coming through the higher levels of the network. For example, a hacker could put malicious commands or data in header data not used for routing and in the payload.
- Each packet is analyzed independently, which means that it is necessary to analyze every packet. This may reduce performance.

- A packet filter cannot recognize forged addresses (IP spoofing) because it only examines the header of the IP packet. This can be corrected (at some extra cost) using Link Layer filtering, where each IP address is correlated to its hardware address [Fra01].

Known Uses

This model corresponds to an architecture that is seen in commercial firewall products, such as: ARGuE (Advanced Research Guard for Experimentation), which is based on Network Associates' Gauntlet Firewall [Eps99]; OpenBSD Packet Filtering Firewall [Rus02], which is the basic firewall architecture for the Berkeley Software Distribution system; and, the Linux Firewall [Zie02], which is the basic firewall architecture used with the Linux operating system. The Packet filter firewall is used as an underlying architecture for other types of firewalls that include more advanced features.

Related Patterns

The Authorization pattern [Fer04] defines the standard security model for the Packet Filter Firewall Pattern. This pattern is also a special case of the Single-Point-of-Access [Yod04] and it is the basis for other, more complex, types of firewalls (described in the other patterns in this language). The DMZ pattern [Dys04] defines a way to configure this pattern in a network. This pattern can also be combined with the Stateful Inspection Firewall.

Proxy-Based Firewall

The Proxy-Based Firewall inspects and filters incoming and outgoing network traffic based on the type of application service they want to access. This pattern controls threats at the Application level of the network architecture. This is in addition to the normal filtering based on addresses.

Example

After we started using Packet Filter firewalls most of the problems went away. However, some of the messages sent from sites we don't consider suspicious contain malicious payloads because hackers are spoofing trusted addresses. These payloads may contain incorrect commands or wrong type and length of parameters. Our Packet Filter firewall cannot stop these attacks because it doesn't look into the message payload and as a result we are having new problems.

Context

Computer systems on a local network connected to the Internet and to other networks, where a higher level of security than the one provided by packet filters is needed. Specifically, we want to control attacks at the Application layer of the network protocol. As mentioned in the example, wrong commands or parameters can produce buffer overflow and other conditions that can be exploited for attacks. In some cases we might also want to authenticate the client to avoid spoofing.

Problem

The Address Filtering Firewall only inspects the network addresses when deciding access for a request. We can only block supposedly malicious sites. How do we protect our network of potential attacks that may be embedded within the data segment of the packets?

Possible solutions are affected by the following forces:

- We need to let external networks access our services and local users access external sites. We don't want isolation as a way to be safe.
- There are a variety of application services in a system, e.g., mail, file transfer, and others. Hackers can plan specific attacks against them and we need to be prepared for a variety of attacks.
- Network administrators deploy and configure a variety of protection mechanisms; hence it is important to have a clear model of what is being protected and what types of attacks are possible.
- The protection mechanism should be able to reflect precisely the security policies of the institution.
- The types of attacks are constantly changing and it should be easy to make changes to the configuration of the protection mechanisms.
- It may be necessary to log requests for auditing and defense purposes.

Solution

Make the client interact only with a proxy of the service requested, which in turn communicates with the protected service. The client can only receive service from the server if an application proxy exists for the requested service. Each application proxy has its own pre-defined (by the administrator) access rules that may be used to authenticate, inspect, change, and filter the incoming (or outgoing) messages.

Structure

Figure 5 shows the class diagram for this pattern. We show here only the Proxy aspects, the classes of Figure 2 can be part of this firewall or can be provided separately. This firewall contains **Proxies**, which in turn contain **Rules**, collected in a **Rule Base**. All the hosts of a local network share the firewall. Each local host provides a set of services. The rules may now specify specific constraints for the use the available services.

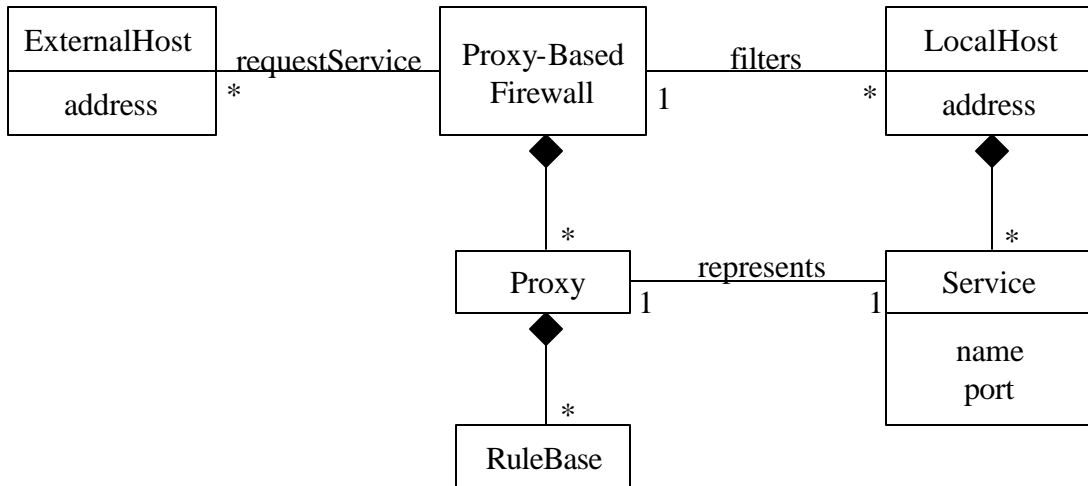


Figure 5: Class diagram for the Proxy-Based Firewall pattern

Dynamics

We show a use case for filtering requests for services.

Use Case for Providing Service to a Client

- Summary: An external client wants access to a service from a local host. The access request is made through the firewall, which according to its application proxies and their rules determines whether to deny or accept the request. Figure 6 illustrates this use case.
- Actors: External client.
- Precondition: None
- Description:
 - An external network requests a service to the Proxy Firewall.
 - The firewall filters the request according to its application proxies and their access rules. If none of the rules in the rule set are satisfied then a default rule is used to filter the request.
 - If the request is accepted, the client is allowed to access the service through the proxy.
- Alternate Flow: If the service request is not supported by the Proxy Firewall, or the firewall considers the client untrustworthy, the firewall will block the access.
- Postcondition: The firewall has accepted the service request from a trustworthy client to the local host.

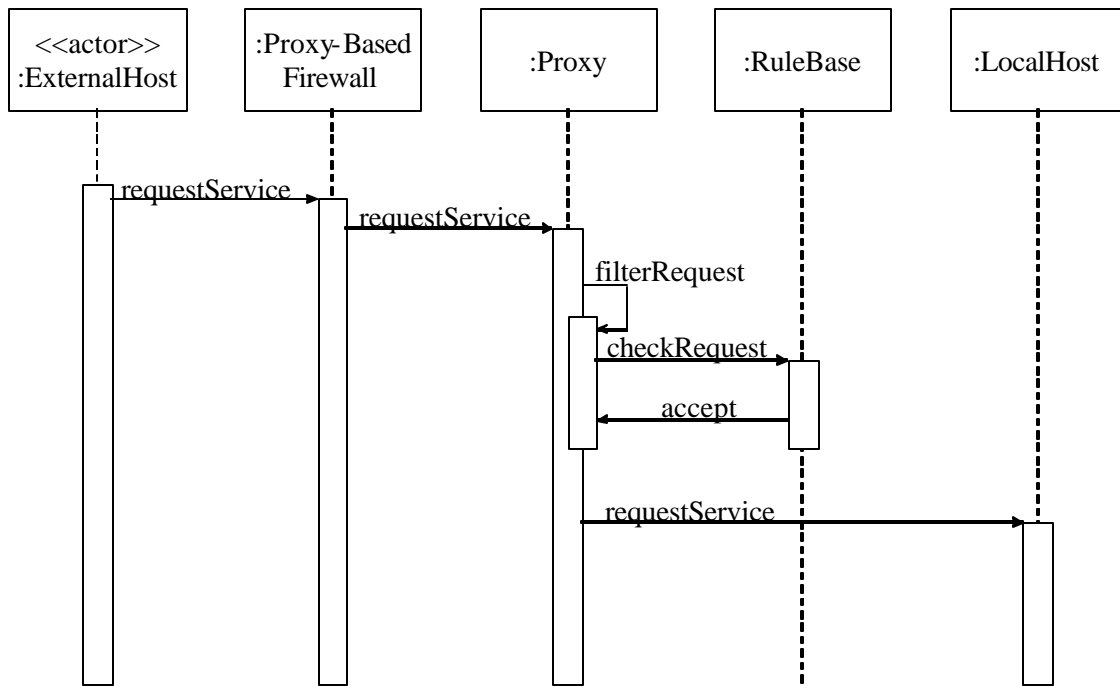


Figure 6. Sequence diagram for filtering service requests.

Implementation

1. According to institution policies, define which services will be made available to clients of the network.
2. Write, reuse, or buy a proxy for each service and assign a port to it.
3. Define who can have what type of access to which service and other restrictions on their use.
4. Implement these constraints in the rule base.
5. Consider configurations such as Reverse Proxy [Som04] or a combination with a Packet Filter firewall in a distributed configuration [Cyb03].

Example resolved

We bought a Proxy-Based firewall and now every request for a service is authenticated and checked. We can verify that the requests are authentic and filter out some payload attacks, e.g., a wrong command for a service.

Consequences

The Application Proxy Firewall Pattern has the following advantages:

- The firewall inspects and filters all access requests based on predefined application proxies that are transparent to the users of the services. In some cases, it may even modify a request.
- It is possible to express the institution's filtering policies through its application proxies and their rules.
- It protects against possible implementation faults in the protocol stacks of the internal systems. The IP (Internet protocol) address of the internal network can be hidden from the external networks.
- A firewall permits systematic logging and tracking of all service requests going through it. This facilitates the detection of possible attacks and helps hold local users responsible of their actions.
- Provides a high level of security because it inspects the complete packet including the headers and data segments.

The Application Proxy Firewall Pattern has the following liabilities:

- Possible implementation costs due to the need for specialized proxies. The proxies also need to be configured correctly. On the other hand, proxies already exist for common services.
- Performance overhead due to the need for inspection of the data segment of packets and maybe additional checking.
- Increased complexity of the firewall. Proxy-based firewalls may require change in applications and/or the user's interaction with the system. This is not necessary, however, in a well-designed system.

Known Uses

Some specific firewall products that use application proxies are Pipex Security Firewalls [Pip03] and InterGate Firewall. The SOCKS Protocol from IETF, although not intended as a firewall, uses a similar principle [Soc].

Related Patterns:

This pattern uses the Proxy pattern from [GOF95]. It can be combined with the Packet Filter Firewall and the Stateful Firewall.

Stateful Firewall

The Stateful firewall filters incoming and outgoing network traffic in a computer system based on state information derived from past communications. State information generally describes whether the incoming packet is part of a new connection or a continuing communication whose connection was approved previously. In other words, states describe a context for each packet.

Example

With Packet Filter and Proxy-Based firewalls we have been able to contain many attacks. However, we are still plagued with distributed denial of service attacks that prevent our customers from reaching our site. We also have performance problems for high-speed streams. In addition, a more sophisticated group of hackers is attacking us sending us viruses whose bodies are assembled from parts included in message data and commands.

Context

Computer systems on a local network connected to the Internet and to other external networks. A higher level of network security is needed than static packet or proxy filtering. A packet filter firewall only inspects the address of the packet, without the knowledge of previous communications of the same network. Similarly, a Proxy-Based firewall filters based on proxy restrictions for each packet. The knowledge of whether a connection is a new connection or an established connection is important for improved security; in particular, denial of service attacks could be identified more conveniently if we knew the relation between packets [Nou00].

Problem

How can we correlate incoming packets? This correlation may be useful to see if they include portions of commands or data needed for attacks or to avoid redundant checks and improve performance.

The solution is affected by the following forces:

- Network administrators deploy and configure a variety of firewalls; hence it is important to have a clear model of what packets correlations are needed being inspected and filtered, and what level of stateful inspection is desired. Otherwise, configuration errors and extra overhead may result.
- The configuration of the firewalls must reflect the institution's security policies; otherwise, it would be difficult to decide on what to filter and what stateful features to include.
- What is being inspected and filtered is constantly changing; hence it should be easy to make changes to the configuration of the firewall.
- It may be necessary to log client requests for auditing and defense purposes.

Solution

Keep a list or table (dynamic rule set) with the connections that have been opened and correlate the type of messages received or sent. This gives the option of not inspecting the packets of a well-established connection.

Structure

Figure 7 shows the Firewall class as including a **State Table** class that describes the existing network connections. The new Client (an external host) can only access our Local Network if a rule exists for authorizing traffic from its address. In addition, if it is a continuing communication from the same client, access is allowed based on whether a corresponding entry is in the State Table. Therefore, each association link between the Client and Local Network is controlled by a Rule and/or an entry in the State Table. The Firewall includes a set of access rules defined for the institution (or local network) according to its policies. A Local Network can have one or more Firewalls. If a particular request is not satisfied by any of the Explicit Rules, then the Default Rule is applied. For every new connection, an entry is made into to State Table.

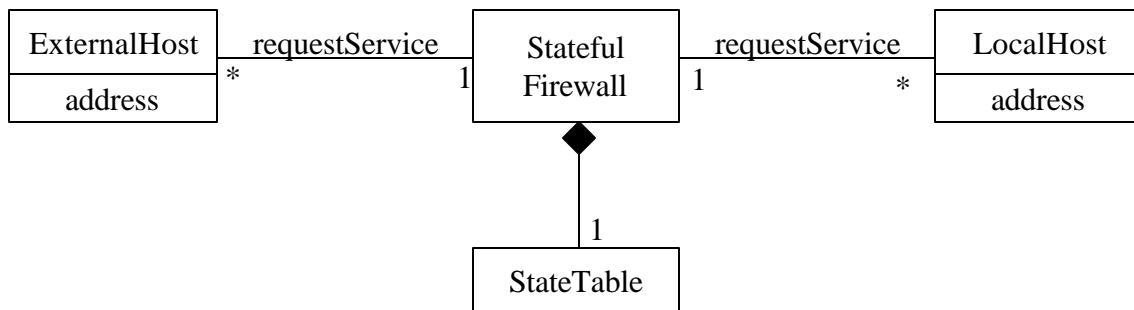


Figure 7: Class diagram for Stateful Packet Filter Firewall

Dynamics

The dynamic aspects of the Stateful Packet Filter Firewall is described by a sequence diagram (Figure 8) that corresponds to the basic use case of filtering a client's request using states.

Filtering a Client's Request (Figure 8):

- **Summary:** A remote network requests access to the local network to either transfer or retrieve information. The access request is made through the firewall,

which according to a state table, and if necessary a set of rules, determines whether to accept or deny the request.

- Actors: External Client.
- Precondition: The state table contains the list of previous established connections or connection attempts. If the state table does not allow a request, rules must be consulted as in the Packet Filter.
- Description:
 - a. An external network requests access to the local network
 - b. A firewall filters the request according to a state table. If the connection exists in the state table, the request is accepted without further inspection.
 - c. If the connection does not exist in the state table, the request is filtered based on a set of rules. If none of the rules are satisfied, then the default rule is used to filter the request, as in the Packet Filter.
 - d. If the request is accepted, the firewall allows access to the local network.

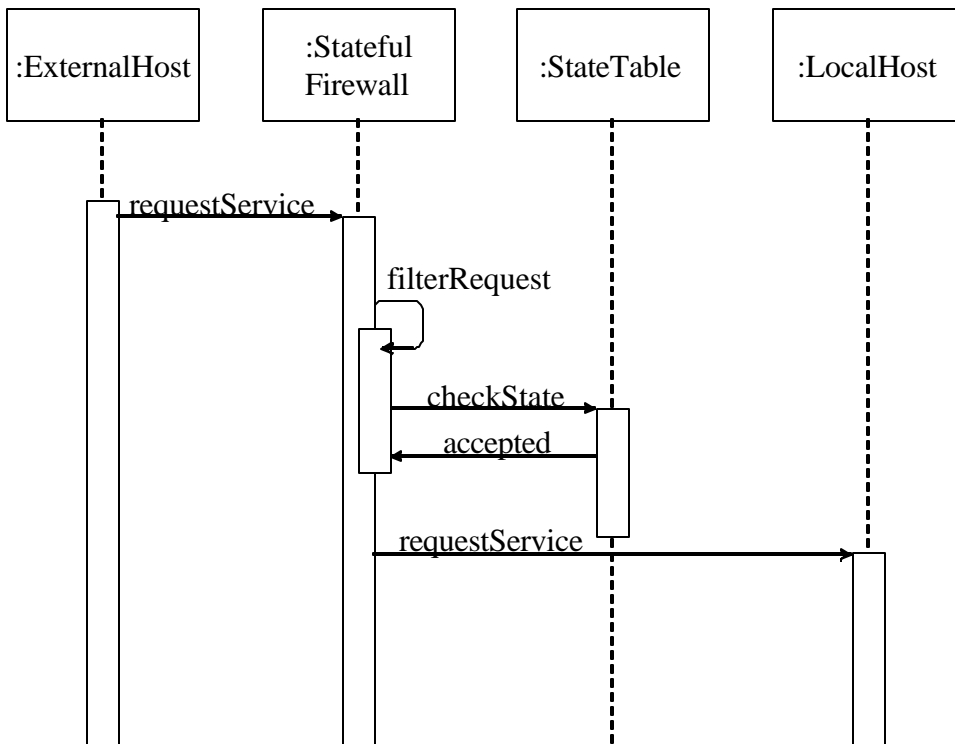


Figure 8: Sequence Diagram for Providing Service to a Client via State Table

- Alternate Flow: If the request is denied, the firewall rejects the access request by the external network to the local network.
- Postcondition: The firewall has filtered the access of a client to a local network.

Implementation

1. Make a list of the types of attacks we want to prevent.
2. Set the state tables to correlate packets according to these attacks.

Example resolved

Typical Denial of Service attacks start by sending a connection request not followed by establishing the connection after its acknowledgement. Our state table keeps a list of all the open connections and if the connections are not established within a given deadline, we just cancel them. We also have a catalog of virus patterns and we can make our firewall inspect sequences of messages to detect these attacks.

Variants

Stateful Packet Filter Firewall

Combines address filtering with state information, i.e., it filters based on the address of the packet and the information in the state table.

Stateful Proxy-Based Firewall

The Stateful Proxy-Based firewall inspects and filters incoming and outgoing network traffic based on the type of network application they are accessing, and the state of the communication between the networks.

Consequences

The Stateful Firewall Pattern has the following advantages:

- Clear objectives, it is relatively easy to set up the state table once we know what attacks we are expecting.
- Low implementation cost, as it requires only a state table. .
- Good performance. It only needs to look at packet headers for new connections. If it is an existing connection, then it looks only at the state table.
- It can enhance the security of the other types of firewalls by adding information about correlated packets.
- New attacks require only more ways to correlate packets.
- It allows connection-based logging of traffic. This may be useful for detecting patterns of attack that can be used by Intrusion Detection Systems.

The Stateful Firewall pattern has the following (possible) liabilities:

- The state table may fill and allow some attacks that take advantage of this fact [Fra01].

Known Uses

This pattern can be found in commercial firewall products from organizations such as Software Technologies [Sof03], Check Point Technologies, and CyberGuard [Hen01]. Some specific firewall products that use stateful application proxies are Pipex Security Firewalls [Pip03] and InterGate Firewall [Vddd].

Related Patterns

This firewall is usually combined with one or both of the previous types of firewalls.

Credits

The first two patterns appeared in [Fer03]. Angela Herzberg was a coauthor of that paper. The participants in the workshop at PLoP'03 and Peter Sommerlad provided valuable comments that improved those patterns. We incorporated later some aspects from [Sch03]. Andy Longshaw provided excellent comments for all the patterns presented here.

References

- [Bar99] Y. Bartal, A. Mayer, K. Nissim, and A. Wool. FIRMATO: A Novel Firewall Management Toolkit. In *Proceedings of the 20th IEEE Symposium on Security and Privacy*, pp 17-31, Oakland, California, April 1999.
- [Che03] W. Cheswick, S.M. Bellovin, A. D. Rubin, *Firewalls and Internet Security*, 2nd Ed., Addison-Wesley, 2003.
- [Cyb03] CyberGuard Corp. , “Defense in depth: Applying a fortified firewall strategy”, White paper, January 2003.
- [Del04] N. Delessy-Gassant, E.B.Fernandez, S. Rajput, and M.M.Larrondo-Petrie, “Patterns for application firewalls”, *Procs. of PLoP 2004*.
- [Dys04] “DMZ”, P. Dyson and A. Longshaw, in *Architecting enterprise solutions*, Wiley 2004. Also, in this chapter.
- [Eps99] J. Epstein. Architecture and Concepts of the ARGuE Guard. In *Proceedings of the 15th Annual Computer Security Applications Conference*, pages 45-54, Phoenix, Arizona, December 1999.
- [Fer01] E. B. Fernandez. An overview of Internet security. In *Proceedings of the World’s Internet and Electronic Cities Conference (WIECC 2001)*, Kish Island, Iran, May 2001.
- [Fer03] E. B. Fernandez, M. L. Petrie, N. Seliya, and A. Herzberg. A Pattern language for firewalls. In *Proceedings of the Pattern Languages of Programs (PLoP) Conference*, 2003.

- [Fer04] E. B. Fernandez, "A pattern language for security models". In this chapter.
- [Fra01] M. Frantzen, F. Kerschbaum, E.E.Schultz, and S. Fahmy, "A framework for understanding vulnerabilities in firewalls using a dataflow model of firewall internals", *Computers & Security*, vol. 20, No 3, 2001, 263-270.
- [Gat00] B. Gatliff. Web by Proxy. *Embedded Systems Programming Magazine*. May 2000. <http://www.embedded.com/internet/0005/0005ia1.htm>
- [GOF95] E. Gamma, R. Helm, R. Johnson, and J. Vlissides, *Design patterns – Elements of reusable object-oriented software*, Addison-Wesley 1995.
- [Hay00] V. Hays, M. Loutrel, and E.B.Fernandez, "The Object Filter and Access Control framework", *Procs. Pattern Languages of Programs (PLoP2000) Conference*, <http://jerry.cs.uiuc.edu/~plop/plop2k>
- [Hen01] P. Henry. An Examination of Firewall Architectures. CyberGuard Corporation White Paper, April 2001. <http://www.cyberguard.com>.
- [Nou00] N. A. Noureldien and I. M. Osman. A Stateful Inspection Module Architecture. In *Proceedings of TENCON 2000*, vol. 2, pp 259-265, Kuala Lumpur, Malaysia, September 2000.
- [Pip03] Pipex Firewall Solutions. <http://www.security.pipex.net/about.shtml>.
- [Rus02] R. E. Rustad, Jr. Guide to OpenBSD Packet Filtering Firewalls. Kuro5hin Article, Nov. 2002, <http://www.kuro5hin.org/story/2002/11/23/14927/477>
- [Sch03] M. Schumacher. Firewall Patterns. In *Proceedings of the Euro PLoP Conference*, 2003.
- [Soc] IETF SOCKS Protocol, <http://www.socks.permeo.com/>
- [Sof03] Software Technologies. www.sofaware.com/downloads/Technical/.
- [Sol] Solsoft, Inc., <http://www.solsoft.com>
- [Som04] P. Sommerlad, "Reverse Proxy". In this chapter.
- [Vddd] InterGate Firewalls from Vicomsoft. <http://www.vicomsoft.com/>
- [Wal01] B. Walder. Firewalls. *Computer Weekly Online*, 2001. <http://www.nss.co.uk/Articles/firewalls01.htm>
- [Yod97] J. Yoder and J. Barcalow, "Architectural patterns for enabling application security". *Procs. PLOP'97*,

<http://jerry.cs.uiuc.edu/~plop/plop97> Also Chapter 15 in *Pattern Languages of Program Design*, vol. 4 (N. Harrison, B. Foote, and H. Rohnert, Eds.), Addison-Wesley, 2000.

- [Yod04] J. Yoder, J. Barcalow, and P. Sommerlad, “Single Access Point”. In this chapter.
- [Zie02] R. Ziegler and C. Constantine. Linux Firewalls: Packet Filtering. *News Riders*, March 2002. <http://informit.com>.
- [Zwi00] E. D. Zwicky, S. Cooper, and B. Chapman. *Building Internet Firewalls*, O’Reilly and Associates, 2nd edition, 2000.