POLITECNICO DI TORINO

Fundamentals of Information Systems Security

Student:

Gianmarco Michelini

Academic Year 2024/2025

All notes are derived from oral presentations and written papers.

Contents

List of Figures

List of Tables

Chapter 1

Firewall and IDS/IPS

[1]

What is a Firewall?

Literally, a firewall is a 'wall to protect against fire propagation' (a safety feature designed to compartmentalize fire and limit damage). In the context of computer networks, it guarantees a controlled connection between networks at different security levels, serving as boundary protection and a network filter.

1.1 Ingress vs. Egress Firewall

> **Beware**
>
> Bidirectional protection is essential. This concept involves protecting both incoming (ingress) and outgoing (egress) network traffic to ensure comprehensive security.

The Ingress Firewall:

- Is intended for **incoming** connections.

- Typically, **controls access** to the (public) services offered by your network or system.

The Egress Firewall:

- Is intended for **outgoing** connections.

- Typically, used to **monitor and control** the activity of internal personnel or devices (to prevent unauthorized traffic, and also for privacy and data protection).

> **Classification of Traffic**
>
> It's straightforward to classify traffic for **channel-based services** (e.g., TCP applications), but more challenging for **message-based stateless services** (e.g., ICMP, UDP applications), due to their lack of a consistent connection state.

1.2 Three Commandments of Firewall

1. The firewall (FW) must be the only contact point between the internal network and the external network.

2. Only "authorized" traffic should be allowed to pass through the firewall.

3. The firewall must be a highly secure system.

   – *D. Cheswick and S. Bellovin*

Referring to each rule:

1. The behavior of employees poses a risk.

2. The technician must understand what they are configuring, especially which rules are necessary.

3. Dedicated security elements should be used to avoid cross-vulnerabilities.

1.3 Authorization Policies

We have two possible choices:

- **Permitlist** (AKA allowlist): "All that is not explicitly permitted, is forbidden."

  - Higher security (gatekeeper).
  - More complex to manage.

- **Blocklist** (AKA denylist): "All that is not explicitly forbidden, is permitted."

  - Lower security (open gates).
  - Easier to manage.

FW: Basic Components

> **Beware**
>
> The Firewall is a system! With several components.

- **Packet filter** / **screening router** / **choke**: A component that filters traffic at the network level.

- **Bastion host**: A secure system with auditing.

- **Application gateway (proxy)**: A service that works on behalf of an application, with access control.

- **Dual-homed gateway**: A system with two network cards and routing disabled (ip-forwarding off).

> **What is a Proxy**
>
> A proxy is a system or service that sits between the client and the application server. It intercepts and controls the traffic between the two, often for purposes such as filtering, caching, security, or access control.

1.4 Control Mechanisms for each Level

To provide a clear and structured explanation of the different controls at various network levels, here's an overview of each control type along with a comparison of how they differ in terms of:

- Controls to be performed (i.e., threats detected).

- Performance.

- Protection of the firewall OS.

- Keeping or breaking the client-server model (where breaking means no direct communication between client and server).

Different controls at various network levels:

- (Static) packet filter.

- Stateful/stateless (dynamic) packet filter.

- Cutoff proxy.

- Circuit-level gateway / proxy.

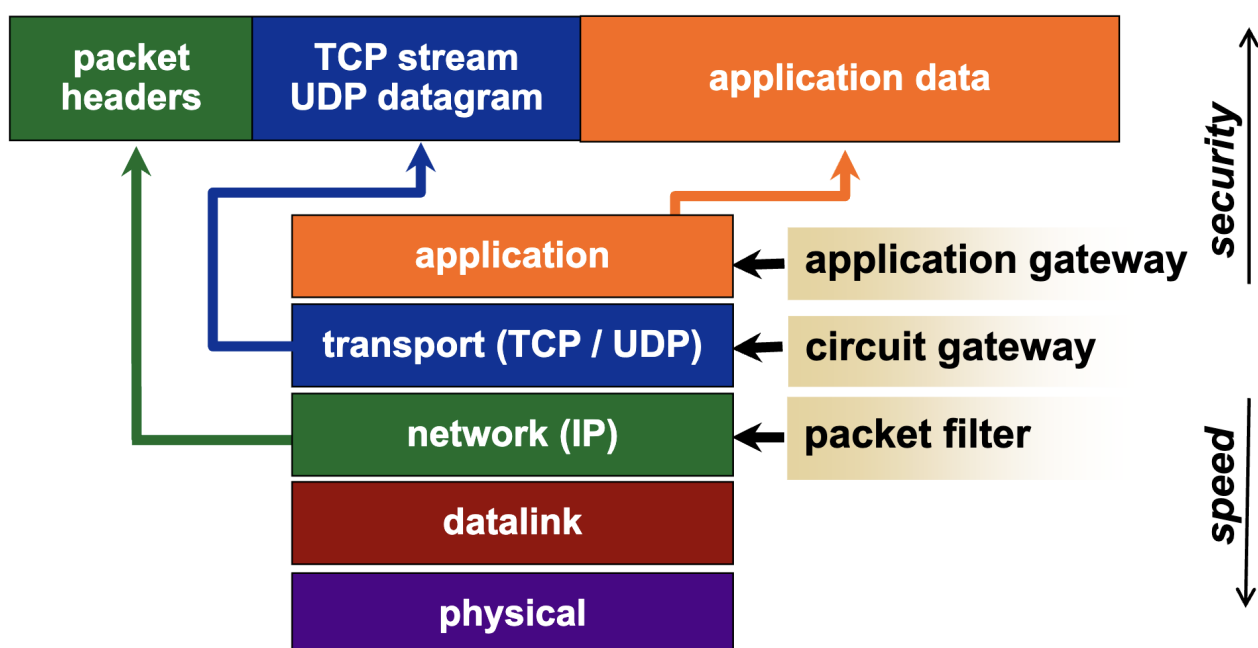- Application-level gateway / proxy.

- Stateful inspection.

Figure 1.1: Controls for each level.

1.4.1 Packet Filter

> **Beware**
>
> Order is important (first match principle).

- Historically available on routers, nowadays found in almost every OS.

- Performs packet inspection at the network level.

- Inspects the IP header.

- Inspects the transport header.

- Rule examples:

    - Permit incoming connections to our web server:

        `src any dst 10.1.2.3/0.0.0.0 tcp 80 allow`

    - Only our internal DNS server can query external DNS servers:

        `src 10.1.2.1/0.0.0.0 dst any udp 53 allow`

- **Pros:**

    - Independent of applications.
    - Good scalability.
    - Good performance.
    - Low cost (available on routers and in many OS).

- **Cons:**

    - Approximate controls: easy to "fool" (e.g., IP spoofing, fragmented packets).
    - Difficult to support services with dynamically allocated ports (e.g., FTP).
    - Complex to configure (and understand the configuration sometimes).
    - Difficult to perform user authentication.

1.4.2 Application-level Gateway

Composed of a set of proxies (collection of elements) inspecting the packet payload at the application level:

- Often requires modifications to the client application.

- May optionally mask or renumber the internal IP addresses.

- When used as part of a firewall, usually performs peer authentication.

- Provides top security (e.g., protects against buffer overflow vulnerabilities in the target application).

- Difference between forward proxy (egress) and reverse proxy (ingress).

- Rule example:

  ```
  deny dangerous HTTP methods "PUT, DELETE deny"
  ```

- SMP (Symmetric Multiprocessing) may improve performance.

- **Pros:**

  - Rules are more fine-grained and simpler compared to those of a packet filter.
  - Provides more protection for the server.
  - May authenticate the client.

- **Cons:**

  - Every application requires a specific proxy.
  - Delay in supporting new applications.
  - Heavy on resources (many processes).
  - Low performance (due to user-mode processes).
  - Completely breaks the client/server model.
  - Not transparent to the client.
  - The proxy's OS may be vulnerable to attacks.
  - Problems with Application-Level Security Techniques that Do Not Permit Traffic Inspection (e.g., TLS)

Variants of Application-level Gateway:

- **Transparent Proxy:**

  - Less intrusive for the client.
  - Requires additional work (packet rerouting and destination extraction).

- **Strong Application Proxy:**

  - Checks semantics, not just syntax.
  - Only some commands/data are forwarded, based on deeper inspection.
  - This is the only correct configuration for a proxy in cases requiring high security.

1.4.3 Circuit-level Gateway

Generic Proxy (i.e., not "Application-Aware")

- Creates a transport-level circuit between the client and server.

- Does not understand or manipulate the payload data in any way.

- Simply copies TCP segments or UDP datagrams between its two interfaces, provided they match the access control rules.

- Re-assembles the IP packets, which helps provide protection against some Layer 3 (L3) and Layer 4 (L4) attacks.

- Breaks the TCP/UDP-level client/server model during the connection.

- Provides more protection for the server.

  - Isolated from attacks related to the TCP handshake.
  - Isolated from attacks related to IP fragmentation.

- May authenticate the client, but this requires modification to the application.

- Exhibits many limitations of a packet filter.

- **SOCKS** is one of the most well-known examples of a generic proxy.

1.4.4 HTTP Proxy

Forward Proxy

- HTTP Server Acting as a Front-End:

- Acts as an egress control, passing requests to the real (external) server.

- **Benefits** (in addition to network ACLs):

  - Shared cache of external pages for all internal users.
  - Authentication and authorization of internal users.
  - Various controls, such as allowed sites, transfer direction, data types, etc.
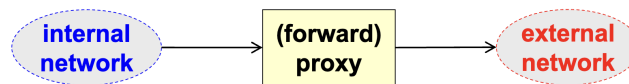

Figure 1.2: Forward proxy example.

Reverse Proxy

- Acts as a front-end for the real server(s), forwarding requests to them.

- Implements network ACL and content inspection.

- **Additional Benefits:**

  - Obfuscation: Hides information about the real server(s).
  - TLS Accelerator: Handles TLS encryption, leaving unprotected connections between the proxy and the backend servers.
  - Load Balancer.
  - Web Accelerator: Caches static content.
  - Compression.
  - Spoon Feeding: Retrieves a full dynamic page from the backend server and delivers it to the client based on its speed, offloading the application server.
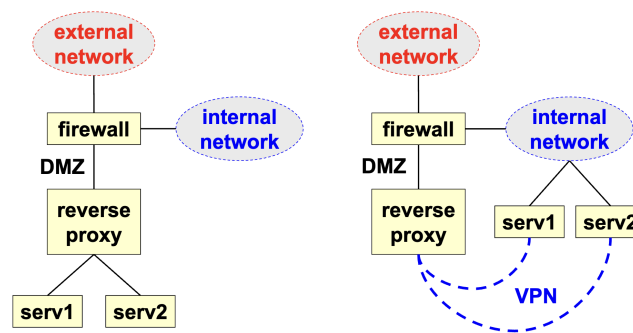
Figure 1.3: Reverse proxy example.

1.4.5 WAF

(Web Application Firewall)

The large use of web applications leads to an increase in threats targeting them.

- A WAF is a module installed at a proxy (either forward and/or reverse) to filter application traffic.

- Filters the following types of traffic:
  - HTTP commands.
  - HTTP request/response headers.
  - HTTP request/response content.

- **ModSecurity:**
  - A popular plugin for Apache and NGINX, which power about 50% and 30% of worldwide HTTP servers, respectively.
  - Includes the **OWASP ModSecurity Core Rule Set (CRS)** to protect against a wide range of attacks.

1.5 Firewall Architectures

1.5.1 Packet Filter

> **Beware**
>
> Simple, cost-effective, but... insecure!

- Exploits the packet filter to screen traffic at both the IP and upper layers.

- The Packet Filter element represents a single point of failure.

- If implemented with a router, it becomes a "screening router," eliminating the need for additional dedicated hardware.

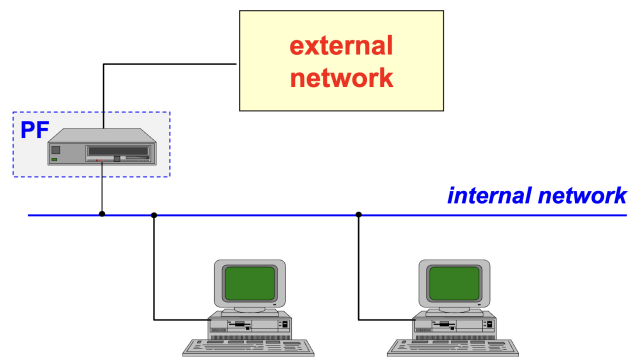- No need for a proxy, thus no modification of applications is required.

Figure 1.4: Packet filter architecture.

1.5.2 Dual-homed Gateway

- Easy to implement.

- Small additional hardware requirements.

- The internal network can be masqueraded.

- Inflexible: The packet filter cannot easily adapt to changing network requirements or policies, and it does not provide much flexibility in managing traffic.

- High work overhead.

- The packet filter element performs an initial screening of the traffic.

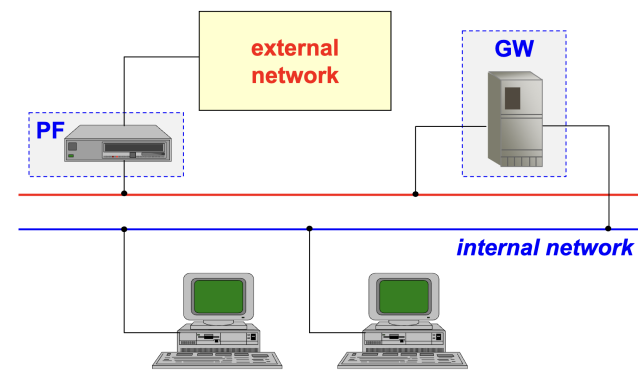- The gateway could become a bottleneck, reducing the overall performance of the system.

Figure 1.5: Dual-homed gateway architecture.
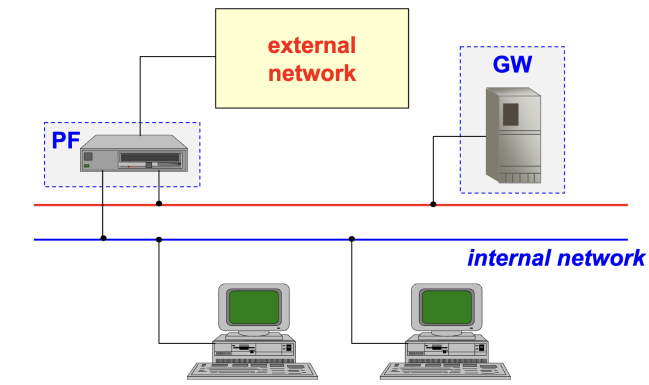
1.5.3 Screened Host

Figure 1.6: Screened host architecture.

Bibliography

[1] Antonio Lioy. Firewall and ids/ips, 2024.