Mastering

Palo Alto Networks

Deploy and manage industry-leading PAN-OS 10.x solutions to secure your users and infrastructure

Packt>

www.packt.com

Tom Piens

Mastering Palo Alto Networks

Deploy and manage industry-leading PAN-OS 10.x solutions to secure your users and infrastructure

Tom Piens

Mastering Palo Alto Networks

Contributors

About the author

Tom Piens, PCNSE, CISSP, and founder of PANgurus, has nearly 10 years of experience working with Palo Alto Networks customers. Tom has been at the forefront of engaging with customers, responding to questions, and analyzing unique needs to apply the best possible solutions or workarounds. He has authored a great many articles on the Palo Alto Networks knowledge base and discussion forum solutions, including the popular Getting Started series. Also known as "reaper" on the PANgurus and LIVEcommunity forums, and `@PANWreaper` on Twitter, Tom has been recognized by Palo Alto Networks user groups and community members, and by countless thankful customers.

3

Building Strong Policies

In this chapter, you will get comfortable with configuring security profiles, building rule bases for security, and **Network Address Translation** (**NAT**). We will learn what each setting does, what its expected behavior is, and how it can be leveraged to lead to the desired outcome. Taking full control over all of the features available in the different rule bases will enable you to adopt a strong security stance.

In this chapter, we're going to cover the following main topics:

- Understanding and preparing security profiles
- Understanding and building security rules
- Setting up NAT in all possible directions

Technical requirements

Before you get started, your firewall must have connectivity between at least two networks, with one preferably being your **Internet Service Provider** (**ISP**), to fully benefit from the information provided in this chapter.

Understanding and preparing security profiles

Before you can start building a solid security rule base, you need to create at least one set of security profiles to use in all of your security rules.

> **Important note**
>
> Security profiles are evaluated by the first security rule that a session is matched against. If a six-tuple is matched against a security rule with no or limited security profiles, no scanning can take place until there is an application shift and the security policy is re-evaluated. It is important for *all* security rules to have security profiles.

The Antivirus profile

The Antivirus profile has three sections that depend on different licenses and dynamic update settings. The actions under **ACTION** rely on the threat prevention license and antivirus updates, **WILDFIRE ACTION** relies on the WildFire license and the WildFire updates that are set to periodical updates (1 minute or longer intervals), and **DYNAMIC CLASSIFICATION ACTION** relies on WildFire set to real time. If any of these licenses are missing from your system, the actions listed in their columns will not be applied. **Application Exception** allows you to change the action associated with a decoder for individual applications as needed. The actions that can be set for both threat prevention and WildFire antivirus actions are as follows:

- **allow**: Allows matching signatures *without* logging

- **drop**: Drops matching signatures and writes an entry in the threat log

- **alert**: Allows matching signatures to pass but writes an entry in the threat log

- **reset-client**: Drops matching packets, sends a TCP RST to the client, and writes an entry in the threat log

- **reset-server**: Drops matching packets, sends a TCP RST to the server, and writes an entry in the threat log

- **reset-both**: Drops matching packets, sends a TCP RST to the client and server, and writes an entry in the threat log

Packet captures can be enabled for further analysis by the security team or as forensic evidence. They are attached to the threat log and are limited to packets containing matched signatures.

Create a new Antivirus profile by going to **Objects | Security Profiles | Antivirus**.

As the following screenshot shows, we will use all the default settings:

DECODER ∧	ACTION	WILDFIRE ACTION	DYNAMIC CLASSIFICATION ACTION
http	default (reset-both)	default (reset-both)	default (reset-both)
http2	default (reset-both)	default (reset-both)	default (reset-both)
imap	default (alert)	default (alert)	default (alert)
pop3	default (alert)	default (alert)	default (alert)
smb	default (reset-both)	default (reset-both)	default (reset-both)

Figure 3.1 – Antivirus Profile

We will now have a look at the Anti-Spyware profile.

The Anti-Spyware profile

The Anti-Spyware profile is extremely customizable and is built by a set of rules within the profile. These rules serve to change the default actions associated with each threat; so, if no rules are created at all, the profile will simply apply the default action for a specific signature when it is detected.

Anti-Spyware supports the same actions as Antivirus (**allow**, **drop**, **alert**, **reset-client**, **reset-server**, and **reset-both**), as well as **block-ip**:

- **block-ip** can track by source or source-destination pair and will block the offending IP for a duration of `1-3600` seconds. Tracking by source will block all connections from the client for the duration of the block, while tracking by source-destination will only block connections from the client to the target destination and will not block the same client from connecting to other destinations.

The **Packet capture** options include `none`, `single-packet`, and `extended-capture`. While `single-packet` only captures the packet containing the payload matching a signature, `extended-capture` enables the capture of multiple packets to help analyze a threat. The number of packets captured by `extended-capture` can be configured via **Device | Setup | Content-ID**. The default is `5`.

> **Important note**
> Enabling packet capture on all threats does require some CPU cycles. The impact will not be very large, but if the system is already very taxed, some caution is advised.

Severity indicates the severity level of the threat that applies to this rule.

Create a new Anti-Spyware profile, as in the following screenshot, and add the following rules:

- **POLICY NAME**: `simple-critical`

  --**SEVERITY**: `critical`

  --**ACTION**: `block-ip (source, 120)`

  --**PACKET CAPTURE**: `single-packet`
- **POLICY NAME**: `simple-high`

  --**SEVERITY**: `high`

  --**ACTION**: `reset-both`

  --**PACKET CAPTURE**: `single-packet`

- **POLICY NAME**: `simple-medium`

    --**SEVERITY**: `medium`

    --**ACTION**: `reset-both`

    --**PACKET CAPTURE**: `single-packet`

- **POLICY NAME**: `simple-low-info`

    --**SEVERITY**: `low, informational`

    --**ACTION**: `default`

    --**PACKET CAPTURE**: `disable`

Your profile will now look like this:

Anti-Spyware Profile

Name: ASprofile

Description:

Signature Policies | Signature Exceptions | DNS Policies | DNS Exceptions

POLICY NAME	SEVERITY	ACTION	PACKET CAPTURE
simple-critical	critical	block-ip (source,120)	single-packet
simple-high	high	reset-both	single-packet
simple-medium	medium	reset-both	single-packet
simple-low-info	low informational	default	disable

⊕ Add   ⊖ Delete   ↑ Move Up   ↓ Move Down   ⊚ Clone   🔍 Find Matching Signatures

OK   Cancel

Figure 3.2 – Anti-Spyware Profile

As you can see in the following screenshot, we need to make sure we review **Category** as this allows a fine-grained approach to each specific type of threat if granularity and individualized actions are needed at a later stage:

Figure 3.3 – Anti-Spyware categories

The Anti-Spyware profile also contains DNS signatures, which are split into two databases for the subscription services.

The content DNS signatures are downloaded with the threat prevention dynamic updates. The DNS Security database uses dynamic cloud lookups.

The elements in each database can be set to **Alert**, **Allow**, **Block**, or **Sinkhole**. **Sinkhole** uses a DNS poisoning technique that replaces the IP in the DNS reply packet, so the client does get a valid DNS reply, but with an altered destination IP. This ensures that infected endpoints can easily be found by filtering traffic logs for sessions going to the sinkhole IP. You can keep using the Palo Alto Networks default sinkhole, `sinkhole. paloaltonetworks.com`, or use your preferred IP.

The way that the DNS sinkhole works is illustrated by the following steps and diagram:

1. The client sends a DNS query to resolve a malicious domain to the internal DNS server.

2. The internal DNS relays the DNS lookup to an internet DNS server.

3. The firewall forges a poisoned reply to the DNS query and replies to the internal DNS server with a record pointing to the sinkhole IP.

4. The DNS reply is forwarded to the client.

5. The client makes an outbound connection to the sinkhole IP, instead of the malicious server. The admin immediately knows which host is potentially infected and is trying to set up **Command and Control** (**C2**) connections:
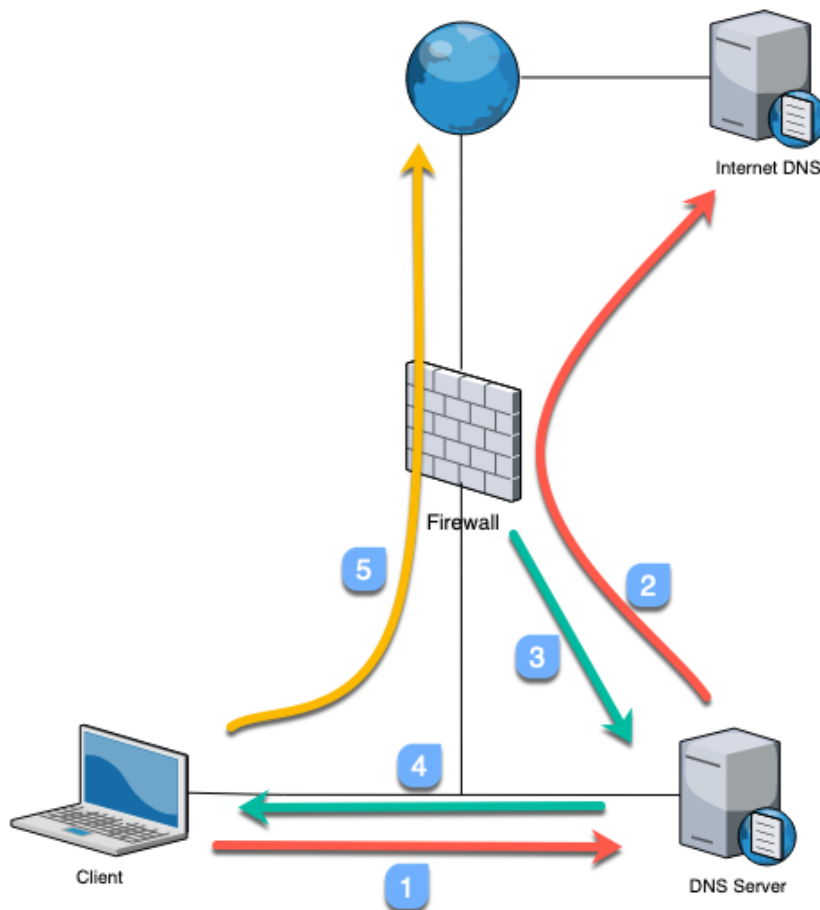

Figure 3.4 – How a DNS sinkhole works

Blocking instead of sinkholing these DNS queries would implicate the internal DNS server as requests are relayed through it. *Make sure you set the DNS Security action to sinkhole if you have the subscription license.*

The default action for the **Command and Control** and **Malware** domains is to block and change them to sinkholes, as shown. For research purposes, you can enable packet capture:

Anti-Spyware Profile   ⑦ ▭

Name	ASprofile	▣
Description		

Signature Policies | **Signature Exceptions** | **DNS Policies** | **DNS Exceptions**

DNS Policies

🔍 Q   6 items → ✕

	SIGNATURE SOURCE	LOG SEVERITY	POLICY ACTION	PACKET CAPTURE
	⌄ : Palo Alto Networks Content			
☐	default-paloalto-dns	high	sinkhole	disable
	⌄ : DNS Security			
☐	Benign Domains	default (none)	default (allow)	disable
☐	Command and Control Domains	default (high)	sinkhole	disable
☐	Dynamic DNS Hosted Domains	default (medium)	default (allow)	disable
☐	Malware Domains	default (high)	sinkhole	disable
☐	Recently Registered Domains	default (medium)	default (block)	disable

DNS Sinkhole Settings

Sinkhole IPv4	Palo Alto Networks Sinkhole IP (sinkhole.paloaltonetworks.com)	⌄
Sinkhole IPv6	IPv6 Loopback IP (::1)	⌄

OK   Cancel

Figure 3.5 – Anti-Spyware DNS signatures

Let's now look at the Vulnerability Protection profile.

The Vulnerability Protection profile

The Vulnerability Protection profile also uses rules to control how certain network-based attacks are handled. **ACTION** contains the same options as Anti-Spyware: **allow**, **drop**, **alert**, **reset-client**, **reset-server**, **reset-both**, and **block-ip**. The reset actions send TCP RST packets. **block-ip** blocks all packets coming from a source and can be set to **monitor source** to block everything, or a source destination, to only block packets to a specific destination for an amount of time.

Host Type helps determine whether the rule applies to a threat originating from a client (upload), server (download), or either.

Make sure you review **Category**, as in the following screenshot, as this allows a fine-grained approach to each specific type of threat if granularity and individualized actions are needed at a later stage:

Figure 3.6 – The Vulnerability Protection profile categories

Create the following rules:

- **Rule Name:** `simple-client-critical`

  --**Host Type:** `client`

  --**Severity:** `critical`

  --**Action:** `block-ip (source, 120)`

  --**Packet Capture:** `single-packet`
- **Rule Name:** `simple-client-high`

  --**Host Type:** `client`

  --**Severity:** `high`

  --**Action:** `reset-both`

  --**Packet Capture:** `single-packet`
- **Rule Name:** `simple-client-medium`

  --**Host Type:** `client`

  --**Severity:** `medium`

  --**Action:** `reset-both`

  --**Packet Capture:** `single-packet`
- **Rule Name:** `simple-server-critical`

  --**Host Type:** `server`

  --**Severity:** `critical`

  --**Action:** `block-ip (source, 120)`

  --**Packet Capture:** `single-packet`
- **Rule Name:** `simple-server-high`

  --**Host Type:** `server`

  --**Severity:** `high`

  --**Action:** `reset-both`

  --**Packet Capture:** `single-packet`
- **Rule Name:** `simple-server-medium`

  --**Host Type:** `server`

  --**Severity:** `medium`

--**Action:** `reset-both`

--**Packet Capture:** `single-packet`

- **Rule Name:** `simple-low-info`

    --**Host Type:** `any`

    --**Severity:** `low, informational`

    --**Action:** `default`

    --**Packet Capture:** `disable`

Your profile should now look like this:

Vulnerability Protection Profile   ⑦ ▭

| Name | VPprofile |
| Description | |

Rules | **Exceptions**

	RULE NAME	THREAT NAME	CVE	HOST TYPE	SEVERITY	ACTION	PACKET CAPTURE
☐	simple-client-critical	any	any	client	critical	block-ip (source,120)	single-packet
☐	simple-client-high	any	any	client	high	reset-both	single-packet
☐	simple-client-medium	any	any	client	medium	reset-both	single-packet
☐	simple-server-critical	any	any	server	critical	block-ip (source,120)	single-packet
☐	simple-server-high	any	any	server	high	reset-both	single-packet
☐	simple-server-medium	any	any	server	medium	reset-both	single-packet
☐	simple-low-info	any	any	any	low informational	default	disable

⊕ Add ⊖ Delete ↑ Move Up ↓ Move Down ⊚ Clone ⌕ Find Matching Signatures

OK   Cancel

Figure: 3.7 – Vulnerability Protection Profile

In the next subsection, we will learn about URL filtering and its categories.

URL filtering

URL filtering leverages URL categories to determine what action to take for each category.

There are two groups of categories: custom URL categories and the dynamic categories provided by the URL filtering license.

Custom URL categories

Custom URL categories do not require a license, so you can create these objects and apply URL filtering even without access to the URL filtering license.

Go to **Objects | Custom Objects | URL Category** to create a new custom category and add websites. It takes a light form of **Regular Expression** (**RegEx**) matched against the address, so neither `http://` nor `https://` are required to match.

The string used in a custom URL category is divided up into substrings, or tokens, by separators. The `./?&=;+` characters are considered separators, so `www.example.com` has three tokens and two separators. Each token can be replaced by a wildcard (`*`) to match subdomains or entire **Top-Level Domains** (**TLDs**). Wildcards cannot be used as part of a token; for example, `www.ex*.com` is an illegal wildcard. Each string can be closed by a forward slash (/) or be left open by not adding an end slash. Not ending a string could have consequences if the string is very short or very common as it could match unintended longer addresses. For example, the `*.com` string could match `www.communicationexample.org`, so adding an ending slash would prevent this.

URL filtering profile

When configuring the URL filtering profile, you need to select which action to apply.

Some possible actions are as follows:

- **Allow**: Allows a category without logging.

- **Alert**: Allows a category and logs the access in the URL filtering log.

- **Block**: Blocks the request, injecting an `HTTP 503` error and a redirect to a page hosted on the firewall explaining to the user their access was declined and the action logged.

- **Continue**: Injects an interactive web page informing the user that they are about to access a restricted website and provides a **Continue** button for them to acknowledge the risk associated with accessing the site.

- **Override**: Injects an interactive web page that allows the user to continue if they are able to provide a password to continue. This password can be set in **Device | Setup | Content-ID | URL Admin Override**. An **Interface Management** profile (**Network | Network Profiles | Interface Mgmt**) needs to be created, with the **Response pages** service enabled and added to the interface where users connect to for this page to work, as follows:

Interface Management Profile

Name: responsepages

Administrative Management Services

- [ ] HTTP
- [ ] HTTPS
- [ ] Telnet
- [ ] SSH

Network Services

- [x] Ping
- [ ] HTTP OCSP
- [ ] SNMP
- [x] Response Pages
- [ ] User-ID
- [ ] User-ID Syslog Listener-SSL
- [ ] User-ID Syslog Listener-UDP

PERMITTED IP ADDRESSES

⊕ Add   ⊖ Delete

Ex. IPv4 192.168.1.1 or 192.168.1.0/24 or IPv6
2001:db8:123:1::1 or 2001:db8:123:1::/64

OK   Cancel

Figure 3.8 – Interface Management Profile

As you can see in the following screenshot, the URL filtering profile requires each **CATEGORY** field to be set to an action individually for site access, and if **USER CREDENTIAL SUBMISSION** is enabled, additional filtering can be applied to decide whether a user is allowed to submit corporate credentials to a certain category. This helps prevent phishing attacks:

Figure 3.9 – URL Filtering Profile

As you can see in the following screenshot, if you want to change a lot (or all) of the actions at once, there's a shortcut to help you. If you hover your mouse over **SITE ACCESS** or **USER CREDENTIAL SUBMISSION**, there will be a little arrow that lets you select **Set All Actions** or **Set Selected Actions**:
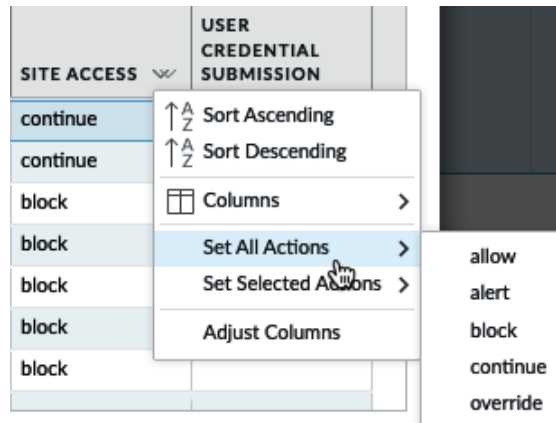
Figure 3.10 – Set All Actions in URL Filtering Profile

A good baseline URL filtering policy can be set up as follows:

1. Set all of the categories to Alert. This will ensure that all of the URL categories are logged.

2. Set **Adult**, **Command and Control**, **Copyright Infringement**, **Extremism**, **Malware**, **Peer-to-Peer**, and **Phishing and Proxy Avoidance and Anonymizers** to Block.

3. Set **Dating**, **Gambling**, **Games**, **Hacking**, **Insufficient Content**, **Not-Resolved**, **Parked**, **Questionable**, **Unknown**, and **Web Advertisements** to Continue.

4. Tweak the settings in accordance with your company policy or local laws and regulations (some URL categories cannot be logged by law, for example).

The **Categories** set to **Continue** are commonly on the fringes of acceptance, but may still need to be accessed for legitimate purposes. The **Continue** action gives the user the opportunity to ensure that they are intending to go to this URL before actually opening the web page.

The URL filtering settings contain several logging options that may come in handy depending on your needs:

*   **Log container page only**: This setting only logs the actual access a user is requesting and will suppress related web links, such as embedded advertisements and content links on the page that the user is visiting, reducing the log volume.

*   **Safe Search Enforcement**: This blocks access to search providers if strict safe search is not enabled on the client side. Currently, Google, Bing, Yahoo, Yandex, and YouTube are supported.

Additional logging can also be enabled:

- **User-Agent**: This is the web browser that the user is using to access a web page.

- **Referer**: This is the web page that links to the resource that is being accessed (for example, Google or CNN linking to a resource page).

- **x-forward-for**: If a downstream proxy is being used by users, this masks their original source. If the downstream proxy supports enabling the **x-forward-for** feature, it will add the client's original IP in the `c` header, allowing the identification of the original user.

The following steps and screenshot show you how to enable these settings in your URL filtering profile:

1. Enable **Log container page only** to provide some privacy to your users and prevent the logging of embedded ad pages.

2. Enable **Safe Search Enforcement**.

3. Enable additional logging for **User-Agent** and **Referer**:

Figure 3.11 – URL filtering settings

The **User Credential Detection** tab allows you to enable credential detection (see *Chapter 6, Identifying Users and Controlling Access* for more details).

HTTP Header Insertion lets you control web application access by inserting HTTP headers into the HTTP/1.x requests to application providers. As you can see in the following example, this can help you control which team IDs can be accessed in Dropbox, which tenants and content can be accessed in Office 365 and Google app-allowed domains. You can create any URL that needs to have a certain header inserted to ensure users are accessing the appropriate instance:

Figure 3.12 – HTTP Header Insertion

Now, let's look at the file blocking profile.

The file blocking profile

The default strict file blocking profile contains all the file types that are commonly blocked and serves as a good template to start from. Select the strict profile and click on the **clone** action, as in the following screenshot, to create a new profile based on this one. If any file types do need to be allowed in your organization, remove them from the block action:
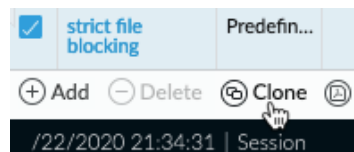

Figure: 3.13 File blocking profile clone

The direction lets you determine whether you want to only block uploads or downloads or both directions for a specific file type, as well as groups of file types. File blocking profiles also use rules so that file types can be grouped with their own directions and actions. The default action is `Allow`, so any file type not included will be allowed to pass through (but will be scanned if an appropriate security profile is attached to the security policy). The available actions are **Alert**, **Block**, and **Continue**, which works similarly to the URL filtering **Continue** option if the file is being downloaded from a web page that supports the HTTP redirect to serve the user a warning page before continuing with the download or upload.

Review all the file types and set the ones you want to block. Any file types that you are not sure about and would like to get a chance to review first can be set to the **Alert** action so that you can keep track of occurrences under **monitor | logs | data filtering**.

As you can see in the following screenshot, we can create sets of file types by clicking on the **Add** button and selecting the file type, and then setting the action:
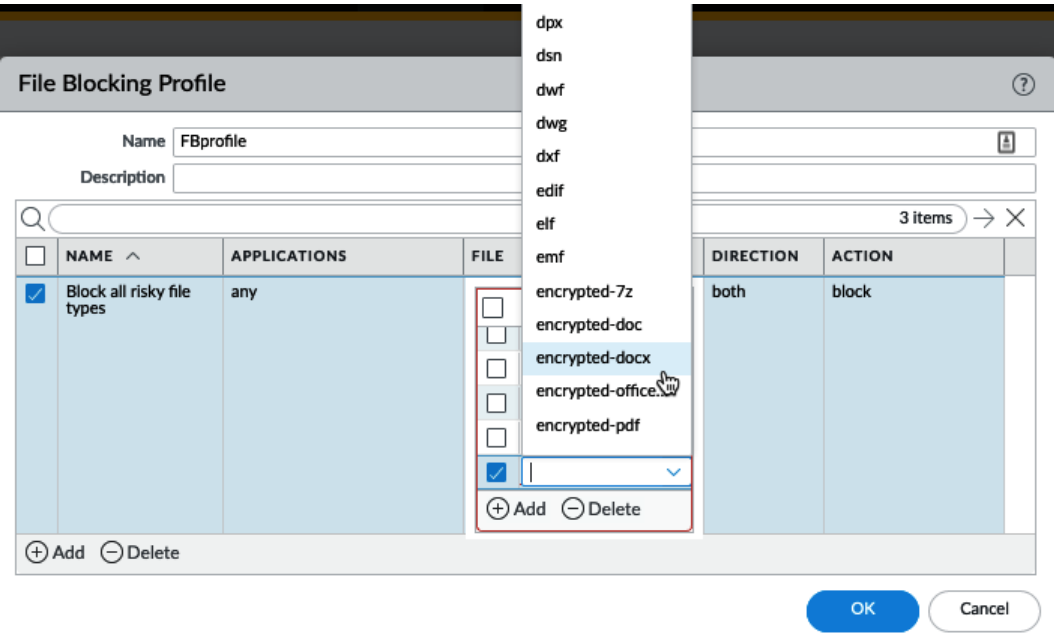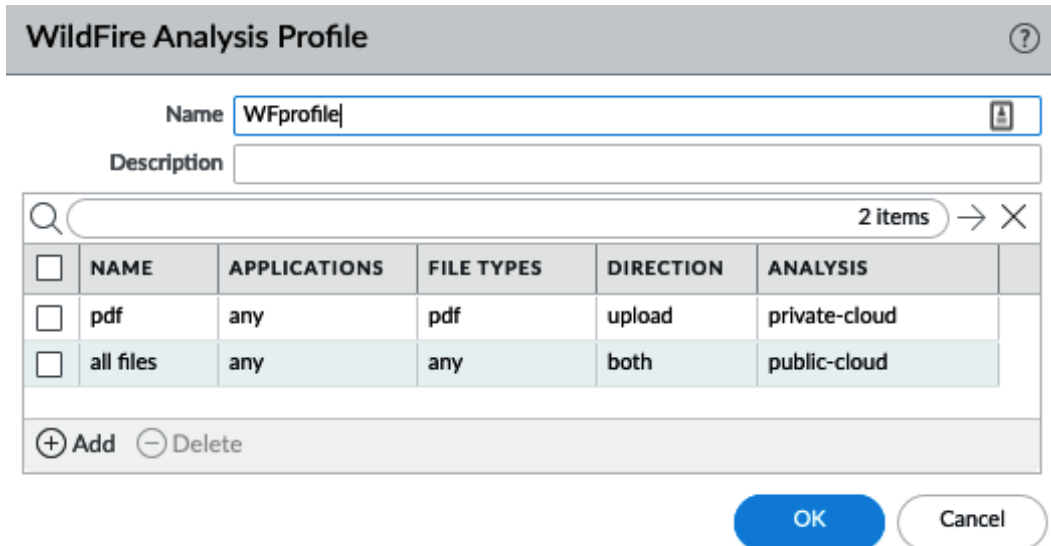

Figure: 3.14 File Blocking Profile

We will now have a look at the WildFire Analysis profile.

The WildFire Analysis profile

The WildFire Analysis profile controls which files are uploaded to WildFire for analysis in a sandbox and which ones are sent to a private instance of WildFire (for example, the WF-500 appliance). Clone the default profile to upload all files to WildFire, or create a new profile if you want to limit which files are forwarded or need to redirect files to a private cloud. If no WildFire license is available, only **Portable Executables** (**PEs**) are forwarded to WildFire.

If all file types can be uploaded for inspection, simply set a rule for any application and any file type. If exceptions exist, either create a rule to divert specific files to a private cloud, if you have a WildFire appliance in your data center, or specify which files *can* be uploaded, as shown:

Figure 3.15 – WildFire Analysis Profile

Next, let's learn about custom objects.

Custom objects

Some security profiles support custom objects. We have already looked at custom URL categories, but the other custom objects are explained in the following sections.

The Custom Spyware/Vulnerability objects

You can create your own signatures using RegEx to detect spyware phone-home/C2 or network vulnerabilities. The **Configuration** page, as shown in the following screenshots, requires basic information, such as an ID number that is between `15.000-18.000` for spyware and `41.000-45.000` for vulnerabilities, a name, a severity value, a direction, and any additional information that may be useful later on. The direction and affected client help the Content-ID engine identify which direction packets that match this signature can be expected:

Figure 3.16 – The Custom Spyware and Vulnerability objects

Under **Signatures**, you have two main modes of adding signatures, as you can see in the following screenshot:

- **Standard**: This adds one or more signatures, combined through logical AND or OR statements.

- **Combination**: This combines predefined (dynamic update) signatures with a timing component requiring *n* number of hits over *x* amount of time, aggregated for source, destination, or source-destination:

Figure 3.17 – The Standard or Combination signatures

Let's focus on standard signatures. From the main screen, you can add sets of signatures, which are all separated by a logical OR statement.

Once you start building a set, you need to decide on the scope. The transaction matches a signature in a single packet and the session spans all the packets in the session. If the signature you are adding to identify a threat always occurs in a single packet's payload, you should set a transaction. This will allow the Content-ID engine to stop scanning at once. If you are adding multiple strings, you can enable **Ordered Condition Match**, which requires the signatures to match from top to bottom in an ordered way. If this option is turned off, the last signature may be detected before the first. If you add multiple strings, you can link them by adding an AND condition.

A signature consists of the following:

- **An operator**, which is either a pattern, or a greater, equal, or smaller operator. Greater, equal, and smaller operators allow you to target a header, payload, payload lengths, and more. A pattern lets you match an exact string found anywhere in a packet or a series of packets.

- **A context**, which is where, in any of the available protocols, the signature may be found (for example, if you look for a string in `http-req-host-header`, that same string will not be matched if it is seen in the payload). Many contexts will be self-explanatory, as you can see in the following screenshot, but for a full list, there's a good online resource describing all the contexts at `https://knowledgebase.paloaltonetworks.com/KCSArticleDetail?id=kA10g000000ClOFCA0`:
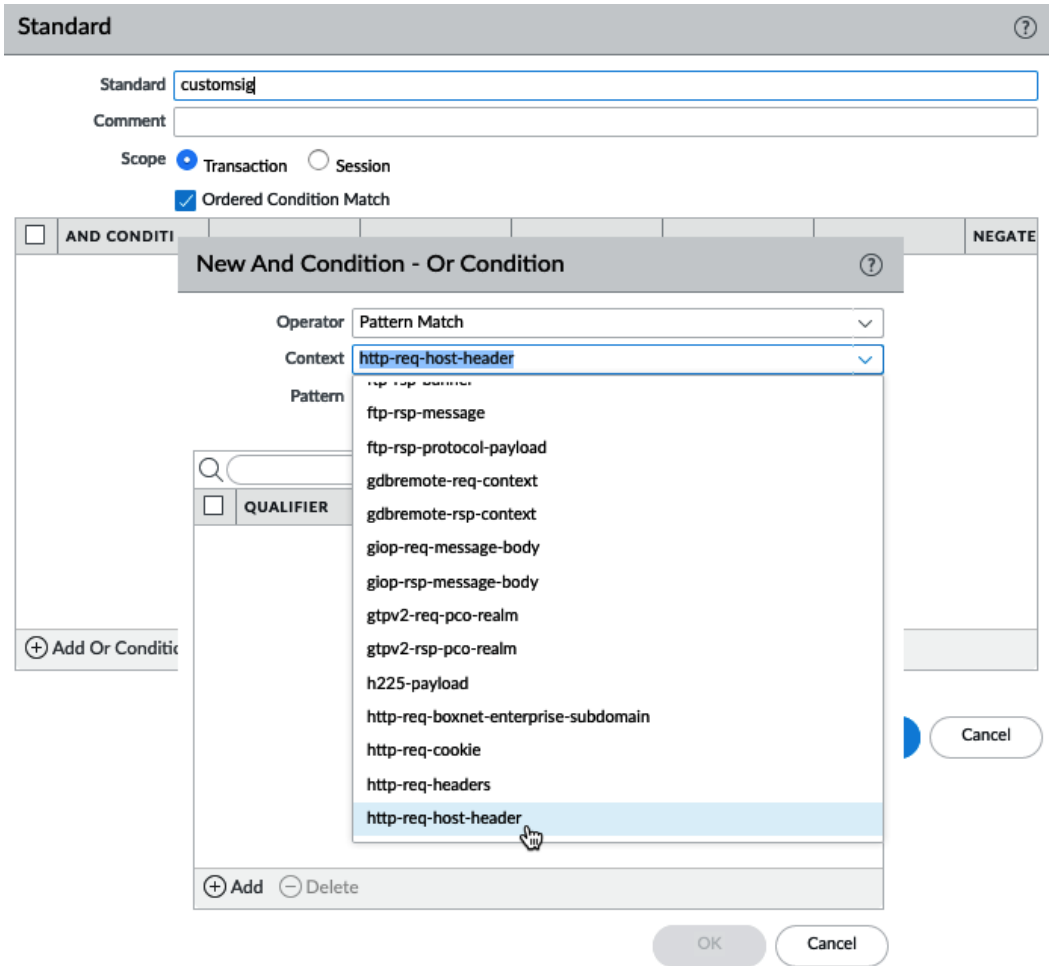

Figure: 3.18 – Creating signatures

- **A pattern or value**: If you want to, for example, match a hostname in an `http` request header, you would use the `domain\.tld` RegEx, where the backslash indicates that the dot following it is an exact match for a dot and not a RegEx wildcard.

  The available RegEx wildcard characters include the following:

.	1.3	matches a single character (e.g. 123, 133)
?	dots?	matches string with or without last character (e.g. dot, dots)
*	dots*	matches string with or without last character, and multiple repeats of last character (e.g. dot, dots, dotssss)
+	dots+	matches single or multiple repetitions of the preceding letter (e.g. dots, dotssss)
|	((exe)|(msi))	OR function to match multiple possible strings (e.g. dot.exe, dot.msi)
[]	x[abc]	matches preceding string followed by any character between squared brackets (e.g. xa, xb, xc)
-	x[a-z]	matches any character in a range (e.g. xa,xm)
^	x[^AB]	matches any character except the ones listed (e.g. xC, x5)
{}	x{1,3}	matches anything after x as long as it is 1 to 3 bytes in length (e.g. x1, x123)
\	x\.y	Escape character to exactly match a special character (e.g. www\.pangurus\.com)
&amp		used to match & in a string

Figure 3.19 – Supported RegEx wildcard characters

- **A qualifier** can further limit in which stage of a transaction a pattern can be matched, either in method or type. Using a qualifier is optional:

Figure 3.20 – Host Header pattern

With the above custom objects you are able to identify sessions behaving in a specific way, but this process can also be applied to identify information and keywords inside a session.

The custom data pattern

In the custom data pattern, you can add strings of sensitive information or indicators of sensitive information being transmitted. There is a set of predefined patterns, including social security numbers, credit card numbers, and several other identification numbers. You can use regular expressions to match exact strings in documents or leverage file properties. Once the appropriate parameters have been chosen, you can add these custom data patterns to a data filtering profile and, as you can see in the following screenshot, assign weights. These weights determine how many times a certain marker can be hit in a session before an alert is generated in the form of a log entry and when a session should be blocked for suspicious behavior (for example, it might be acceptable for an email to go out containing one social security number, but not multiple):

Figure 3.21 – Data filtering

Now that you've had a chance to review and configure all the available security profiles, the easiest way to apply them to security rules is by using **Security Profile Groups**.

Security profile groups

Now that you've prepared all of these security profiles, create a new security profile group, as in the following screenshot, and call it `default`. This will ensure that the group will automatically be added to every security rule you create:

Figure 3.22 – The default security profile group

> **Important note**
>
> It is not harmful to add *all* of the security policies to a security rule as Content-ID will intelligently only apply appropriate signatures and heuristics to applications detected in the session (for example, `http` signatures will not be matched with `ftp` sessions).

Also, create a **Log Forwarding** profile called `default`, but you can leave the actual profile empty for now.

Understanding and building security rules

We now need to build some security rules to allow or deny traffic in and out of the network. The default rules will only allow intrazone traffic and will block everything else, as you can see here:

	NAME	TYPE	Source		Destination		APPLICATION	SERVICE	ACTION	PROFILE	OPTIONS
			ZONE	ADDRESS	ZONE	ADDRESS					
10	intrazone-default	intrazone	any	any	(intrazone)	any	any	any	⊘ Allow	none	none
11	interzone-default	interzone	any	any	any	any	any	any	⊘ Deny	none	none

Figure 3.23 – Default security rules

We will first make sure "bad" traffic is dropped by creating two new rules—one for inbound and one for outbound traffic.

Dropping "bad" traffic

The inbound rule will have the external zone as a source and the three **External Dynamic Lists** (**EDLs**) containing known malicious addresses. These lists are updated via the threat prevention dynamic updates. The **Source** tab should look similar to the following:
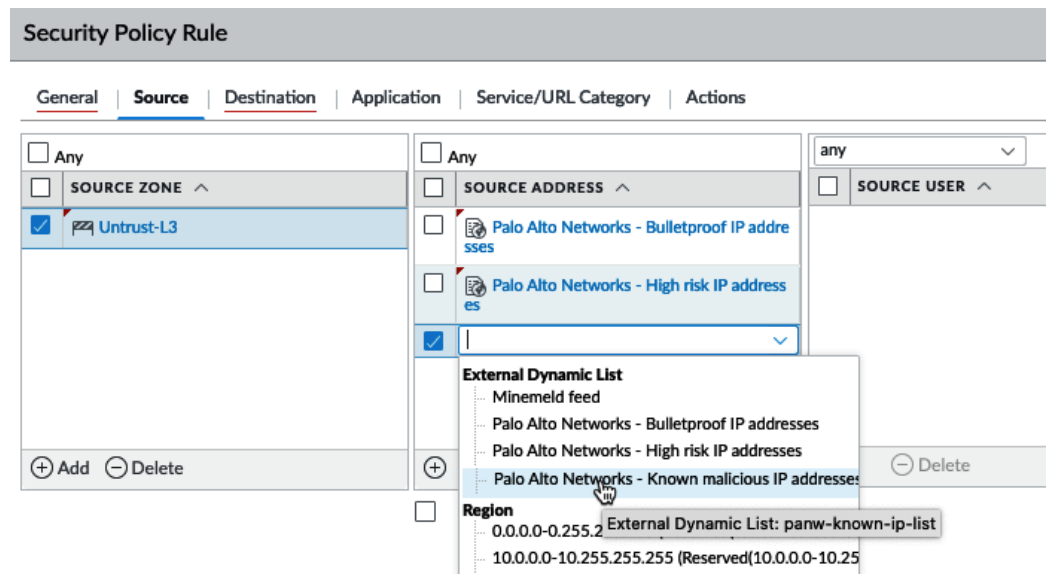

Figure 3.24 – Reconfigured external dynamic lists

In the **Destination** tab, set the destination zones to both the external zone and any zone where you intend to host internal servers that you will allow inbound NAT to (for example, corporate mail or web servers) and set the destination addresses to Any, as in the following screenshot:
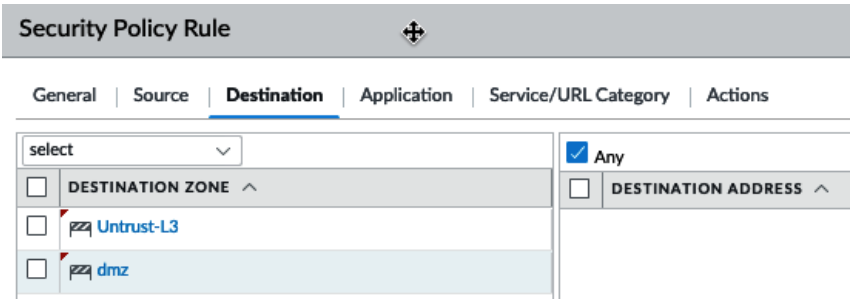
Figure: 3.25 – Security rule destination zones

In the **Actions** tab, set the action to `Drop`. This will silently discard any inbound packets:

Figure 3.26 – Security rule actions

Follow the next steps to create this rule:

1. Create a new security rule and give it a descriptive name.

2. Set the source zone to any zone that is connected to the internet (for example, **Untrust**).

3. Set the source addresses to the three predefined EDLs.

4. Set the destination zones to your internal zones that will accept inbound connections from the internet (for example, DMZ), also including the external zones.

5. Set the action to **Drop**.

> **Important note**
> You may have noticed that the **Profile Setting** fields and **Log Forwarding** are filled out with the **default** profiles that you created in the previous step. In all rules where sessions are blocked, content scanning will not take place, so having these profiles will not cause overhead.

Click **OK**, and then make the reverse rule, as in the following screenshot, setting the source zones to your internal zones, the destination to the external zone, and the predefined EDL as addresses. If you changed the DNS sinkhole IP address to one of your choosing, add this IP here as well:
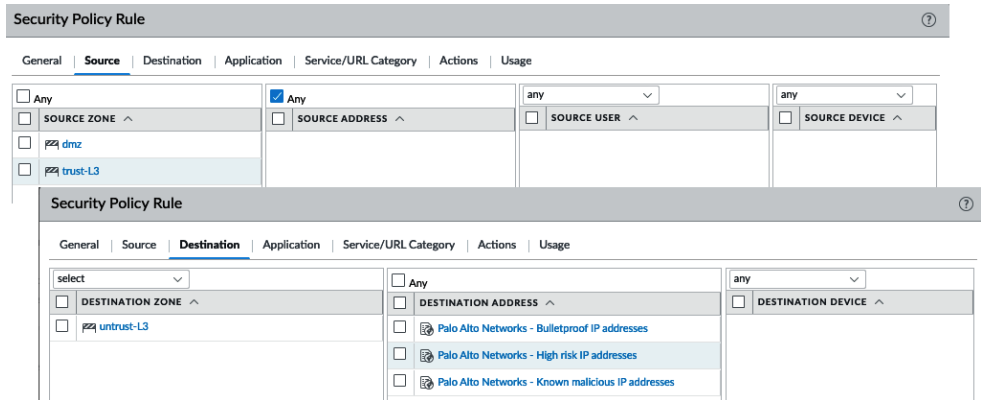
Figure 3.27 – Outbound drop rules

Follow these steps to create the above rule:

1. Create a new security rule and give it a descriptive name.

2. Set the source zone to your internal zones.

3. Set the destination zone to any zone leading out to the internet (for example, **Untrust**).

4. Set three destination addresses and for each one, select one of the predefined EDLs.

5. Set **Action** to Drop.

A good practice is to add some catch all rules to the end of your rule base, as in the following screenshot, once all the required policies have been added to any catch connections that are not allowed. One catch all rule should be set to application-default and one to any; this will help identify standard applications that are not hitting a security policy and (more suspicious) non-standard applications that are not using a normal port (see the *Allowing applications* section to learn about the application-default service):

Figure 3.28 – The catch all rules at the end of the rule base

You now have some rules actively dropping connections you do not want to get past the firewall, but there are more options available than to just silently discard packets. We'll review the other options next.

Action options

There are multiple actions that handle inbound connections, some of which are stealthy and some of which are noisy and informative, depending on your needs:

- **Deny**: This action will drop the session and enforce the default **Deny** action associated with an application. Some applications may silently drop while others send an RST packet.

- **Allow**: This allows the session to go through.

- **Drop**: This silently discards packets.

- **Reset Client**: This sends a TCP RST to the client.

- **Reset Server**: This sends a TCP RST to the server.

- **Reset Both**: This sends a TCP RST to both the client and the server.

If you check the **Send ICMP Unreachable** checkbox and the ingress interface is Layer 3, an **ICMP Unreachable** packet is sent to the client for all of the dropped TCP or UDP sessions.

Allowing applications

There are generally two approaches to determining which applications you want to allow:

- Creating a group of known applications

- Creating an application filter to sort applications by their behavior

From **Objects | Application Groups**, you can create groups of known applications that can be used in security policies, as shown:

> **Important note**
> The security rule base is evaluated from top to bottom and the evaluation is stopped once a match is found, then the matching security rule is enforced. This means blocking rules need to be placed *above* the allowing rule if there could be an overlap.
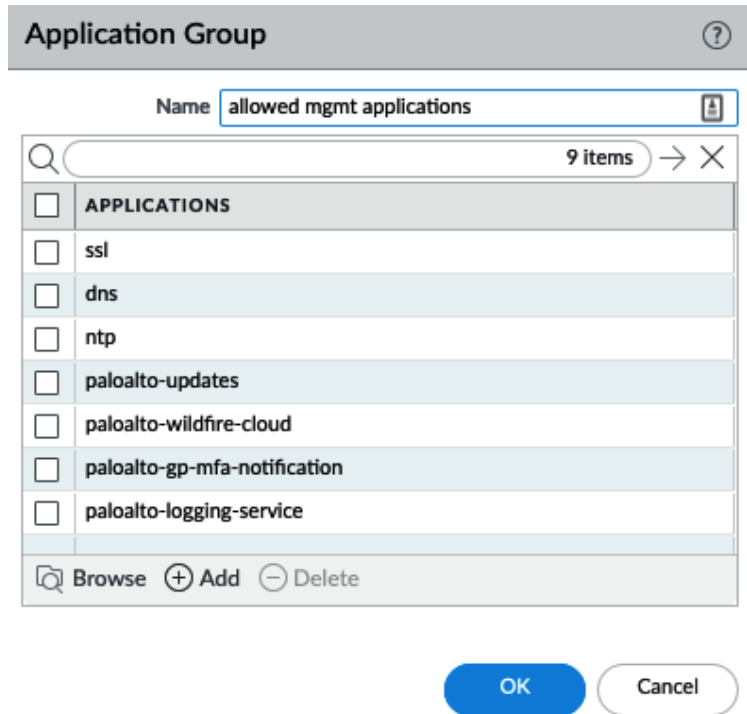
Figure 3.29 – Application Group

With the widespread adoption of cloud-based hosting and cheap SaaS solutions, more traditional programs are turning into web-based applications that are accessible over a web browser. This makes it harder for an administrator to easily determine which applications need to be allowed as the needs of the business change quickly. Application filters created in **Objects | Application Filters** let you create a dynamic application group that adds applications by their attributes, rather than adding them one by one. These attributes can be selected for both "good" properties to be added to allow rules (as you can see in the following screenshot) or "bad" properties to drop rules:

Figure 3.30 – Application Filter with basic attributes

Alternatively, the filter can be based on the predefined and custom tags assigned to applications, as follows:

Figure 3.31 – Application Filter with tags

You can mix and match application groups and filters to build further security rules by adding them to the **APPLICATIONS** tab, as you can see here:
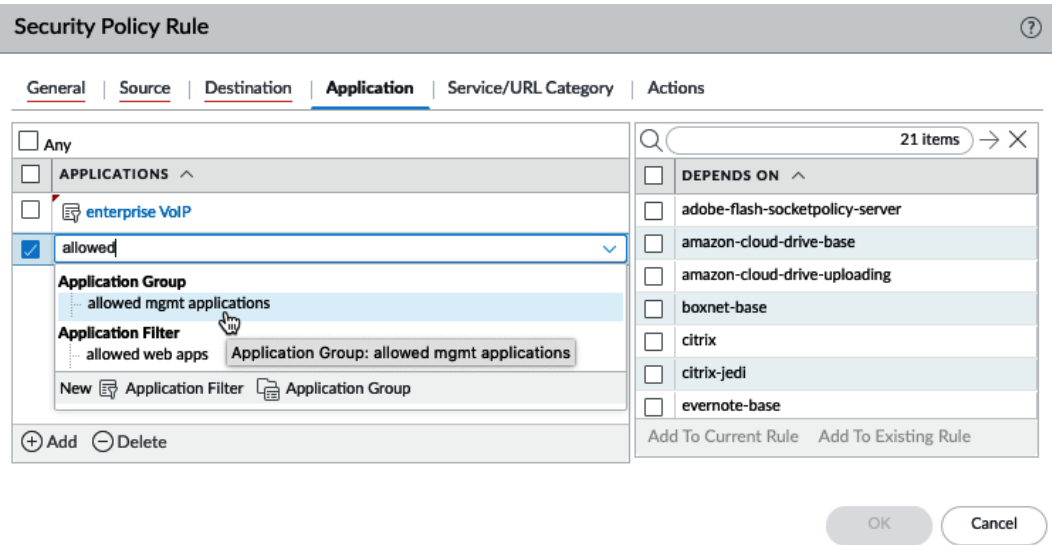
Figure: 3.32 – The APPLICATIONS tab

To create a new **Allow** rule using an application filter, do the following:

1. Create a new security rule and add a descriptive name.

2. Set the source zone to the internal zones that will connect to the internet.

3. Set the destination zone to `external zone`.

4. In **APPLICATIONS**, add a new line and select **Application Filter**.

5. Click on all of the desired attributes and review some of the applications at the bottom. Add a descriptive name and click **OK** on the filter, and again on the security rule.

You now have an **allow** rule based on an application filter!

Application dependencies

As you may have noticed in the previous screenshot, when you start adding applications to a security rule, there may be applications that have dependencies. These applications rely on an underlying protocol or build on an existing more basic application that needs to be added and allowed in the security rule base for this sub-application to work. They do not necessarily need to be added to the same security policy.

Starting from PAN-OS 9.1, these dependencies are displayed in the security rule. As you can see in the following screenshot, they appear when you are adding new applications and can immediately be added to the same security rule or to a different one in the security rule base. In older PAN-OS versions, users will only be warned about these dependencies once the configuration is committed. You can review application dependencies for individual applications via **Objects | APPLICATIONS**, too:
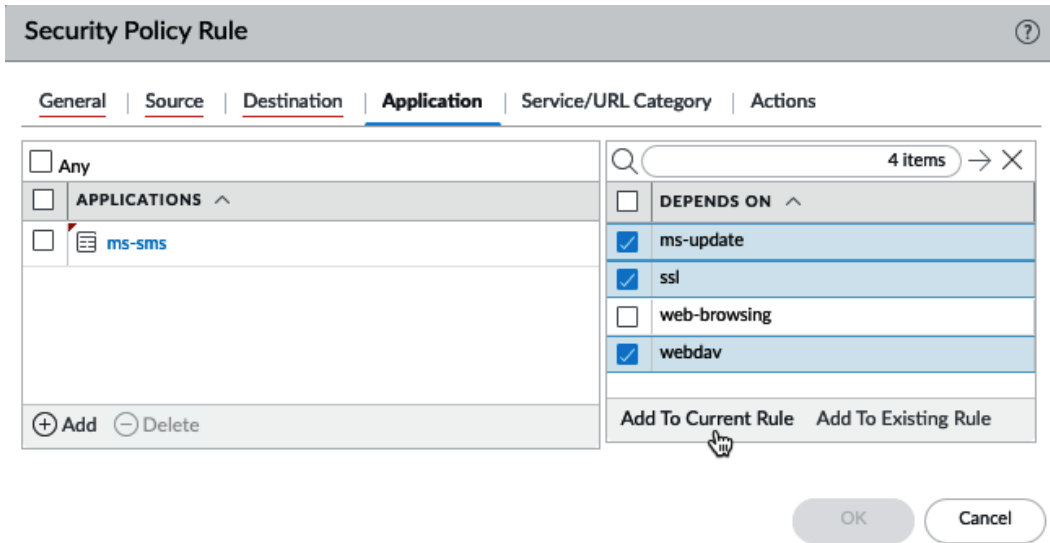
Figure: 3.33 – Application dependencies

Now that the applications have been set, let's look at how service ports are controlled.

Application-default versus manual service ports

Each application will use a certain service port to establish a connection. By default, each service is set to `application-default`, which forces each application to use its default ports (for example, web-browsing uses ports `80` (unsecured) and `443` (SSL) secured, FTP uses ports `21` (unsecured plaintext) and `990` (secured), and so on).

> **Important note**
> Protocols that use pinholing, such as FTP, are automatically taken care of via the **Application Layer Gateway** (**ALG**), which is a part of the content decoder that is specific to this protocol.

If an application needs a custom port, you can add a manual service object, but this would prevent the use of `application-default`. So, any exceptions should preferably be made in individual rules to prevent applications from "escaping" via an unusual port:
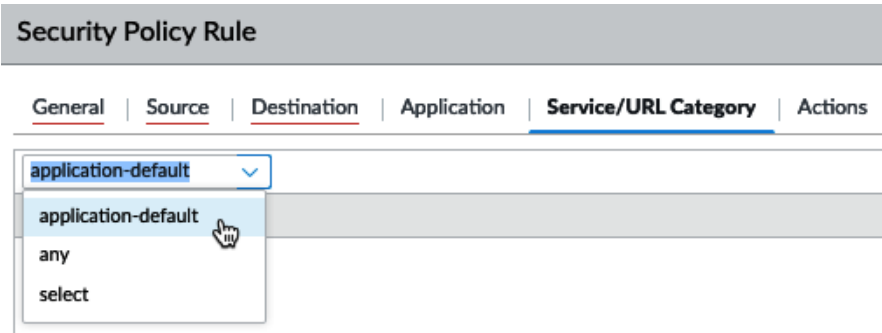
Figure 3.34 – Service ports

Adding a URL category can be used to allow or block URL categories at the TCP layer. For **drop** or **deny** actions, this will mean that the session is dropped, rather than returning a friendly blocked page to the user.

Controlling logging and schedules

By default, each security rule is set to **Log at Session End**. This means that a log is only written to the traffic log once a session is broken down. For some sessions, it may be interesting to log more interactions, and so **Log at Session Start** could be enabled. This does cause quite a lot of overhead, however, as there will be a log for each new stage of a session when the SYN packet is received and for every application switch. So, there could be two to five additional log entries for a single session.

Other applications that are very chatty or less relevant may not need to be logged at all, such as DNS.

> **Important note**
> Even with both start or end log disabled in the security rule action tab, any threats detected in a session will still be logged to the threat log.

Log forwarding can be used to forward logs to Panorama or a syslog server or to send out an email. As you can see in the following screenshot, if you call one of the log forwarding profiles `default`, it will be used in all the new rules, so logs are automatically forwarded:

Figure 3.35 – Log options and schedules

Schedule can be used to create timeframes when this security rule will be active if certain applications are only allowed at specific times of the day (for example, Facebook can be allowed during lunch and after hours):

Figure 3.36 – Schedules

Before you continue putting this new knowledge to work and start creating more rules, let's review how you can prepare address objects.

Address objects

To make managing destinations in your security and NAT policy a little easier, you can create address objects by going to **Objects | Addresses**. When you create a new object here, you can reuse the same object in different rules, and if something changes, you only need to change the address object for all the security and NAT rules to be automatically updated:

1. Click on **Add** and provide a descriptive name for the address. It is good practice to set up a naming convention so that you can repeat this process for all other address objects. A good example is to prefix all server names with `S_` and all networks with `N_` so that they're easily identifiable.

2. Set a description if needed.

3. Select the type of object that this will be.

    --**IP Netmask** lets you set an IP with a subnet mask down to `/32` or `/64` for a single IPv4 or Ipv6 address (no need to add `/32`).

    --**IP Range** lets you define a range that includes all the IP addresses between the first and last IP set in the range, separated with a dash (`-`).

    --**IP Wildcard Mask** lets you set a subnet masking that covers binary matches, where a zero bit requires an exact match in the IP bit, and `1` is a wildcard. So, for example, a wildcard subnet of `0.0.0.254` translates to `000000000.0000000.0 000000.11111110`. the first three bytes are set and in the last byte, all but the first bit are wildcards. This means that if the associated IP address is set to `10.0.0.2` (`00001010.0000000.000000.00000010`), all of the IPs in the subnet that end in 0 will be matched (that is, all of the even IP addresses). If the IP is set to `10.0.0.1`, all of the odd IPs would match. This type of object can only be used in security rules.

    --**FQDN** lets you set a domain name that the firewall will periodically resolve according to the **Time To Live** (**TTL**) and cache. Up to `10` `A` or `AAAA` records are supported for each FQDN object. Use the **Resolve** link to verify that the domain can be resolved.

4. Add a tag to easily identify and filter policies for this object.

5. Click **OK**.

Once you have sets of objects that are similar, you can also create groups by going to **Objects | Address Groups**. These groups can be used to bundle objects for use in security or other policies.

Tags

Tags can be leveraged to group, filter, or easily identify many other objects. Security zones, policy rules, or address objects can all be tagged with up to 64 tags per object. By going to **Objects | Tags**, you can create new tags:

1. Click on **Add** and create a descriptive and preferably short name for the tag (up to 127 characters). You can also use the dropdown to select one of the already-created security zones, which will cause the tags to be automatically assigned to this zone.

2. Select a color or leave it as None.

3. Add a comment.

4. Click **OK**.

As you can see in the following screenshot, tags can then be used to visually enhance your rule base or to filter for specific types of rules:

Figure: 3.37 – Tags in the security policy

> **Important note**
>
> While building security rules, objects (such as addresses, applications, services, and so on) can be clicked and dragged from the object browser on the left into any rule, and from one rule to another. There is no need to open a rule and navigate to the appropriate tab to add objects.

While you're on the **Security policy** tab, there's a tool called **Policy Optimizer** on the bottom left-hand side that can help improve your security rules by keeping track of rule usage.

Policy Optimizer

After a while, you will want to review the security rule base you've built to make sure you haven't missed any applications, left rules too open, or have any duplicates that leave rules unused. Policy Optimizer records statistics relating to your rules and can report the following:

- Rules that have been unused for 30 days, 90 days, or for all time so that you can delete them

- Rules that are set up with no applications defined and the applications that were accepted by those rules

- Rules that have applications that are not being used so that you can remove these excess applications

Now that you are able to build a complete security rule base, there may need to be Network Address Translation for sessions coming in from the internet.

Creating NAT rules

Unless you are one of the lucky few organizations that were able to get their very own `A (/8)` or `B (/16)` class subnets, your internal network segments will most likely be made up of one or several of the well-known RFC1918 private IP address allocations: `10.0.0.0/8`, `172.16.0.0/12`, or `192.168.0.0/16`. NAT is needed for your hosts to be able to reach the internet and your customers and partners to reach publicly available resources hosted in your data center. NAT rules can be configured through **Policies | NAT**.

For this section, keep the following interface setup in mind:

INTERFACE	INTERFACE TYPE	MANAGEMENT PROFILE	LINK STATE	IP ADDRESS	VIRTUAL ROUTER	SECURITY ZONE
ethernet1/1	Layer3		🔲	198.51.100.2/24	default	Untrust-L3
ethernet1/2	Layer3		🔲	192.168.27.1/24	default	Trust-L3
ethernet1/3	Layer3		🔲	10.0.0.1/24	default	DMZ-L3

Figure 3.38 – Interface zone and IP configuration

Address translation comes in different flavors depending on the direction and purpose, each with its own nuances. Let's first review Inbound NAT.

Inbound NAT

For Inbound NAT, it is important to remember that the firewall is zone-based and the source and destination zone are determined *before* the NAT policy is evaluated:
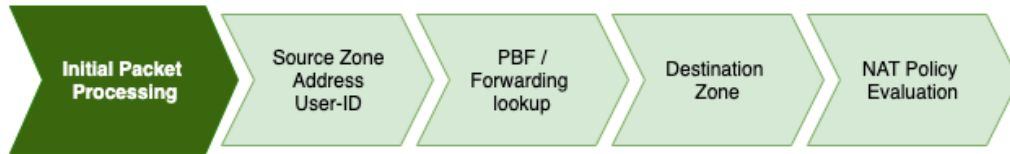
Figure 3.39 – Packet flow stages

This means that for inbound NAT, the source and destination zone will be identical. The routing table will determine the source zone based on the default route and the destination zone based on the connected network, which is configured on the external interface.

For example, if the `203.0.113.1` internet IP is connecting to the `198.51.100.2` firewall IP to reach the `10.0.0.5` server, the firewall will look up `203.0.113.5` in its routing table and find that it only matches the default route, `0.0.0.0/0`, which points out of the `ethernet1/1` interface, which is in the `Untrust-L3` zone. It will then look up `198.51.100.2` (the original destination IP in the packet header) and find it in the `198.51.100.0/24` connected network on the `ethernet1/1` interface, which is in the `Untrust-L3` zone.

The **Original Packet** tab needs to have the following:

- The same source and destination zones.

- **Source Address** can be **Any** for generic internet sources, specific IP addresses, or subnets if the source is known.

- **Destination Interface** indicates which interface the packet is headed to. This can be important in cases where there are multiple interfaces with overlapping routes.

- **Service** can be used to restrict which destination port is allowed in the received packets. This will help in cases where the IP space is restricted and **Port Address Translation** (**PAT**) is required to host different services on the same external IP and will prevent over-exposing an internal host.

- **DESTINATION ADDRESS** needs to be a single IP for a one-to-one destination NAT (don't add a subnet). Having a subnet-based destination NAT is possible, but only for **Session Distribution**:

Figure 3.40 – Original Packet NAT translation

In the **Translated Packet** tab, you can set what needs to be changed for the external client to be able to reach the internal server:

- **Source Address Translation** will usually be set to None, but it can be set to match an internal interface subnet or loopback interface if required. This would let the server receive a packet sourced from an internal IP, rather than the original internet IP.

- Destination translation to a static IP, also known as one-to-one NAT, changes the destination IP to a single internal server.

- **Translated Port** can be used if the internal service runs on a different port than the externally advertised one. For example, externally, a web server could be reachable on default SSL port 443, while on the server itself, the service is enabled on 8443:

NAT Policy Rule   ⑦

General | Original Packet | **Translated Packet**

┌─ **Source Address Translation** ──────────────┐   ┌─ **Destination Address Translation** ──────────────┐
│   │   │   **Translation Type** │ Static IP   ⌄ │ │
│ **Translation Type** │ None   ⌄ │   │   **Translated Address** │ 10.0.0.5   ⌄ │ │
│   │   │   **Translated Port** │ [1 - 65535]   │ │
│   │   │   ┌─ ☐ **Enable DNS Rewrite** ──────────────┐   │
│   │   │   │   Direction │ reverse   ⌄ │   │
│   │   │   └──┘   │
└───┘   └──┘

                                                              (OK)   (Cancel)

Figure 3.41 – Translated Packet NAT translation

Next, let's take a look at address translation in the opposite direction.

Outbound NAT

Outbound NAT rewrites the source IP addresses of internal clients to the interface associated with a different zone. This could be an internet-facing zone or one connecting to a partner, VPN, or WAN, as in the following screenshot:

- The source zone will reflect the interface that the clients are connected to.

- The destination zone and destination interface will reflect the egress interface that a routing lookup determines based on the original packet:

NAT Policy Rule   ⑦

General | **Original Packet** | Translated Packet

☐ Any	Destination Zone	☐ Any	☑ Any
☐ SOURCE ZONE ⌃	Untrust-L3 ⌄	☐ SOURCE ADDRESS ⌃	☐ DESTINATION ADDRESS ⌃
☐ 🖾 Trust-L3	**Destination Interface**	☐ 🖵 dhcpspace	
	ethernet1/1 ⌄		
	Service		
	any ⌄		
⊕ Add ⊖ Delete		⊕ Add ⊖ Delete	⊕ Add ⊖ Delete

                                                              (OK)   (Cancel)

Figure 3.42 – Outbound NAT Original Packet

> **Important note**
>
> When using an IP pool for source translation, the firewall will use proxy ARP to gain ownership of IP addresses. This means that you don't need to physically configure all of the IP addresses on an interface, but it is recommended that you have at least the subnet configured on an interface so that the firewall knows which interface is used to broadcast the proxy ARP packets. If the subnet does not exist on an interface, proxy ARP will be broadcast out of all the interfaces.

Hide NAT or one-to-many NAT

The most common implementation of outbound NAT is the infamous *hide NAT*, or many-to-one, which changes the source IP addresses of all internal clients to the external IP(s) of the firewall. It is best to place this rule near the bottom of the rule base as it will catch any non-specific sessions and rewrite the source IP to that of the firewall.

The best option for this type of NAT is **Dynamic IP and Port** (**DIPP**). DIPP rewrites the source IP to that of a selected interface or a manually entered IP, IP-range, or subnet, and assigns a random source port to the session on egress, as you can see here:

Figure 3.43 – DIPP to an interface IP or manual selection

DIPP supports around `64.000` concurrent sessions per available source IP, multiplied by the oversubscription factor supported by the platform you are deploying these rules on. As a rule of thumb, smaller platforms commonly support 2x oversubscription, larger platforms support 4x, and extra-large platforms up to 8x. When multiple IPs are available, DIPP assigns a rewrite IP based on a hash of the source IP so that the same source always gets the same translation address. Once the concurrent allowance for a given translation address is depleted, new sessions will be blocked until existing sessions are freed up.

You can check the current oversubscription ratio by using the following command:

```
admin@PA-220> show running nat-rule-ippool rule <rule name>


VSYS 1 Rule <rule name>:
Rule: <rule name>, Pool index: 1, memory usage: 20344
-----------------------------------------
Oversubscription Ratio:              2
Number of Allocates:                 0
Last Allocated Index:                0
```

If more than `64.000 x oversubscription ratio` concurrent sessions per source are needed, or source ports need to be maintained, you can opt to use Dynamic IP instead of DIPP. Dynamic IP will simply "hop" to the next available IP in its assigned translation addresses for a given source IP while maintaining the source port. As a fallback, if the available IP pool does get depleted because Dynamic IP does not support oversubscription, you can enable DIPP. The IP used in the fallback should not overlap with any of the main IP pools:

Figure 3.44 – Dynamic IP with two subnets and DIPP fallback

In some cases a server or host on the network will need to "own" its own IP address, which can be achieved with one-to-one NAT rules.

One-to-one NAT

Static IP will always translate a source into the same translation IP and maintain the source port. An IP range can be set, in which case the source IPs will be sequentially matched to the translated IPs, but it is important that the source range and translation range are identical in size; for example, `10.0.0.5-10.0.0.15` translates to `203.0.113.5-203.0.113.115`.

The bi-directional option creates an *implied* inbound NAT rule to allow inbound translation for the same source/translated-source pairs. This implied rule reuses the destination zone set in the rule and `any` as the new source zone. The 'Translated source' address of the configured rule will be used as the 'Original destination' address, and the 'Original Source' of the configured rule will be used as the 'Translated destination' of the implied rule.

For the outbound rule, as you can see in the following screenshot, you have the following:

- **Source**: `Trust-L3`
- **Destination**: `Untrust-L3`
- **Original source**: `serverfarm`
- **Translated source**: `serverfarm-public`

For rules that have bi-directional set, the following implied NAT rule will be created:

- **Source**: `any`
- **Destination**: `Untrust-L3`
- **Original destination**: `serverfarm-public`
- **Translated destination**: `serverfarm`

NAT Policy Rule   ⑦

General | **Original Packet** | Translated Packet

☐ Any	Destination Zone	☐ Any	☑ Any
☐ SOURCE ZONE ⌃	Untrust-L3 ⌄	☐ SOURCE ADDRESS ⌃	☐ DESTINATION ADDRESS ⌃
☐ 🖧 Trust-L3		☐ 🖥 serverfarm	
	Destination Interface		
	ethernet1/1 ⌄		
	Service		
	any ⌄		
⊕ Add ⊖ Delete		⊕ Add ⊖ Delete	⊕ Add ⊖ Delete

                                                   (OK) (Cancel)

NAT Policy Rule   ⑦

General | Original Packet | **Translated Packet**

Source Address Translation

Translation Type | Static IP ⌄

Translated Address | serverfarm-public ⌄

☑ Bi-directional

Destination Address Translation

Translation Type | None ⌄

                                                   (OK) (Cancel)

Figure 3.45 – Static IP NAT with the Bi-directional option

In some cases "double NAT" needs to be applied to sessions that need to take an unusual route due to NAT. These types of NAT rules are called U-turn or hairpin NAT rules.

U-turn or hairpin NAT

If an internal host needs to connect to another internal host by using its public IP address, a unique problem presents itself.

For each session, only one NAT rule can be matched. When the client connects to the public IP, the routing table will want to send the packet out to the internet, which will trigger the hide NAT rule, which translates the source IP. The packet should then go back inside as the destination IP is also owned by the firewall, but a second NAT action can't be triggered, so the packet is discarded.

> **Important note**
>
> If the hide NAT IP is identical to the destination IP, which is common in environments with few public IP addresses, the packet will be registered as a land attack:

```
admin@PA-220> show counter global | match land


Global counters:
Elapsed time since last sampling: 26.05   seconds


name      value  rate severity category  aspect   description
-------------------------------------------------------------
Flow_parse_land 1 1  drop      flow       parse    Packets
dropped: land attack
```

A workaround to this problem, if changing the internal DNS record or adding an entry to the host file of the client is not possible, is to configure a U-turn or hairpin NAT.

> **Important note**
>
> If you are using PAN-OS 9.0.2 or later, refer to the following *Enable DNS Rewrite* section.

This type of NAT combines the destination and source NAT and must be placed at the top of the rule base to prevent the hide NAT rule from catching these outbound sessions. The reason the source NAT is required is to make the session stick to the firewall so that no asymmetric routes are created.

If you were to configure the destination NAT to rewrite the public IP for the internal IP without translating the source, the server would receive a packet with the original source IP intact and reply directly to the client, bypassing the firewall. The next packet from the client would be sent to the firewall, which would try to perform TCP session sanity checks and determine whether the TCP session was broken, discarding the client packet. Adding source translation would force the server to reply to the firewall, which would then forward the translated packet back to the client:

Figure 3.46 – U-turn NAT

This type of complication can also be addressed by changing the DNS query to the internal IP of the final destination.

Enable DNS Rewrite

Enable DNS Rewrite was introduced in PAN-OS 9.0.2 and later and enables the NAT policy to be applied inside DNS response packets:

- It reverse translates the DNS response that matches the *translated* destination address in the rule. If the NAT rule rewrites `198.51.100.2` to `10.0.0.5`, the reverse rewrite will change the DNS response of `10.0.0.5` to `198.51.100.2`.

- It forward translates the DNS response that matches the *original* destination address in the rule. The forward DNS rewrite changes the DNS response of `198.51.100.2` to `10.0.0.5`.

This could be useful in a scenario where internal hosts need to query a DNS server in the DMZ for an FQDN of a server also hosted in a DMZ where they receive the external IP in the DNS response. This could lead to odd routing issues (see the *U-turn or hairpin NAT* section) as the destination IP will match the external zone, but both the client and server are on internal zones:
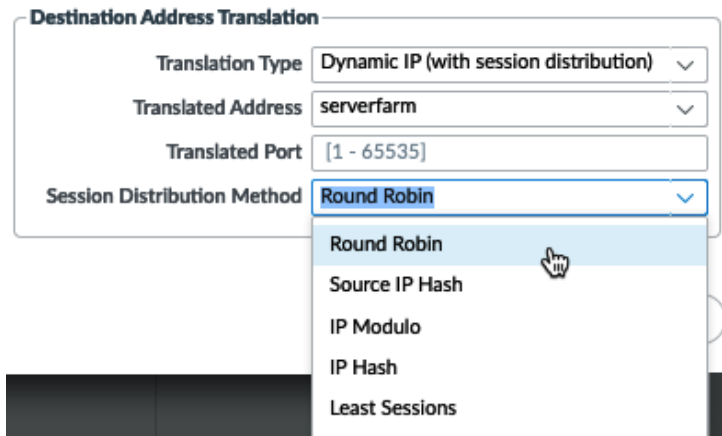

Figure 3.47 – Enable DNS Rewrite

If a service is hosted on several physical servers (the original destination is an FQDN that returns several IP addresses), the destination translation settings can be set to **Dynamic IP (with session distribution)**. The firewall will rewrite the destination IP according to the chosen method:

Figure 3.48 – Dynamic IP (with session distribution)

With this information you will now be able to resolve any Network Address Translation challenges you may face.

Summary

In this chapter, you learned how to create security profiles and how to build a set of profiles that influence how your firewall processes threats. You learned how to create a default security profile group so that your security rule base starts off with a strong baseline of protection, as well as how to create solid security rules. You can now make complex NAT policies that cater to the needs of your inbound and outbound connections.

In the next chapter, we will see how to take even more control of your sessions by leveraging policy-based routing to segregate business-critical sessions from the general internet, limit bandwidth-hogging applications with quality of service, and look inside encrypted sessions with SSL decryption.