Understand Snort3 Rules

Contents

[Introduction](#)
[Prerequisites](#)
[Requirements](#)
[Licensing](#)
[Components Used](#)
[Background Information](#)
[Snort3 rules](#)
[Rule actions](#)
[Rule anatomy](#)
[Rule features](#)
[Examples](#)
[Example with http service header and sticky buffer http_uri](#)
[Example with file service header](#)
[Related Links](#)

Introduction

This document describes rules for the **Snort3** engine in the Cisco **Secure Firewall Threat Defense (FTD).**

Prerequisites

Requirements

Cisco recommends that you have knowledge of these topics:

- Cisco **Secure Firewall Threat Defense (FTD)**
- **Intrusion Prevention System (IPS)**
- **Snort2** syntax

Licensing

No specific license requirement, the base license is sufficient and the features mentioned are included in the **Snort** engine within the FTD and in the **Snort3** open-source versions.

Components Used

The information in this document is based on these software and hardware versions:

- Cisco **Secure Firewall Threat Defense (FTD)**, Cisco **Secure Firewall Management Center (FMC)** version 7.0+ with **Snort3.**

The information in this document was created from the devices in a specific lab environment. All of

the devices used in this document started with a cleared (default) configuration. If your network is live, ensure that you understand the potential impact of any command.

Background Information

Snort is the Cisco IPS engine capable of real-time traffic analysis and packet logging.

Snort can perform protocol analysis, content searching, and detect attacks.

Snort3 is an updated version of the Snort2 IPS with a new software architecture that improves performance, detection, scalability, and usability.

Snort3 rules

They use that LUA format to make the **Snort3** rules easier to read, write and verify.

Rule actions

This new version changes the rule actions, the new definitions are:

- **Pass**: Stop evaluation of subsequent rules against packet
- **Alert**: Generate event only
- **Block**: Drop packet, block remainder session
- **Drop**: Drop packet only
- **Rewrite**: Required if the replaces option is used
- **React**: Send HTML block response page
- **Reject**: Inject TCP RST or ICMP unreachable

Rule anatomy

The anatomy is:

The rule header contains the action, protocol, source and destination network(s), and port(s).

In **Snort3**, the rule header can be one of the next options:

- Service rule header

```
<iline" lang="lua">alert http ( msg:"Alert HTTP rule"; flow:to_client,established;
content:"evil", nocase; sid:1000001; )
```

- File rule header

```
alert file ( msg: “Alert File example”; file_data; content:”malicious_stuff”; sid:1000006; )
```

- Conventional rule header

```
alert tcp $EXTERNAL_NET any -> $HOME_NET $HTTP_PORTS ( msg:"Alert HTTP rule";
flow:to_client,established; content:"evil", nocase; sid:1000001; )
```

Rule features

Some of the new features are:

- Arbitrary whitespace (each option on its own line)

```
alert tcp $EXTERNAL_NET any -> $HOME_NET $HTTP_PORTS ( msg:"Alert TCP rule";
flow:to_client,established; content:"evil", nocase; sid:1000000; )
```

- Consistent use of , and ;

```
content:"evil", offset 5, depth 4, nocase;
```

- Networks and ports are optional

```
alert http ( Rule body )
```

- Adds more sticky buffers (This is not the complete list)

```
http_uri http_raw_uri http_header http_raw_header http_trailer http_raw_trailer http_cookie
http_raw_cookie http_true_ip http_client_body http_raw_body http_method http_stat_code
http_stat_msg http_version http2_frama_header script_data raw_data
```

- C Style comments

```
alert http ( msg:"Alert HTTP rule"; /* I can write a comment here */ ... )
```

- Remark (rem) keyword

```
alert http ( msg:"Alert HTTP rule"; flow:to_client,established; rem:"Put comments in the rule
anywhere"; content:"evil", nocase; sid:1000001; )
```

- appids keywords

```
alert tcp $HOME_NET any -> $EXTERNAL_NET any ( msg:"Alert on apps"; appids:"Google, Google
Drive"; content:"evil", nocase; sid:1000000; )
```

- sd_pattern for sensitive data filtering
- Regex keyword with the usage of hyperflex technology
- Service keyword replaces metadata

Examples

Example with http service header and sticky buffer http_uri

Task: Write a rule that detects the word **malicious** in the HTTP URI.

Solution:

```
alert http ( msg:"Snort 3 http_uri sticky buffer"; flow:to_server,established; http_uri;
content:"malicious", within 20; sid:1000010; )
```

Example with file service header

Task: Write a rule that detects PDF files.

Solution:

```
alert file ( msg:"PDF File Detected"; file_type: "PDF"; sid:1000008; )
```

Related Links

Snort Rules and IDS Software Download

Github