FortiGate – Troubleshooting Guide
Quick Reference

Hubert Wisniewski

About the Autor

Hubert Wisniewski is a Principal Network Engineer at AT&T. He has been working in IT industry for 20 years and last 10 years with computer networks and security. He holds the following certifications: Fortinet NSE4 NSE5 NSE7, Cisco Certified Network Professional (CCNP) R&S Sec, Cisco Certified Design Professional (CCDP), CompTIA Network+ Security+. Hubert, as Fortinet Certified Trainer, delivers official Fortinet courses. He also developed FortiGate Bootcamp. Hubert is a member of Cisco Security Exam Advisory Group as Subject Matter Expert.

About the Technical Reviewers (alphabetical order)

Eduard Dulharu is a Senior Network Architect at AT&T for more than 6 years. He has more than 10 years of experience in designing, implementing and troubleshooting large scale networks for different industries and customers.

Efren Teruel Dominguez is a Senior Network Engineer at AT&T. He has been working with all life cycles of computer networks, like deployment, developing or troubleshooting. In past he worked for Amazon in Dublin and Seattle focusing on automation. He is a member of Cisco Service Provider Advisory Group as SME.

Giovanni Pagano Dritto is a Freelancer Network Consultant and programmer. He works in the IT industry for over 10 years and he has extensive work experience with a wide variety of network technologies and vendors. He is also an experienced Python Developer with a demonstrated history of working in different sectors of IT industry.

Lucian Lisov is a Senior Network Engineer at AT&T with over 10 years of experience. Specialized in enterprise data center infrastructures he devotes most of his time to private/public clouds and automation.

Warning, disclaimer, copyright, trademark, acknowledgments and errata

The information in the book is provided on an “as is” basis. The author disclaims all responsibility for errors or omissions, including without limitation responsibility for damages resulting from the use of the book.

The opinions expressed in this book belong to the author and are not necessarily those of Fortinet.

Errata: http://myitmicroblog.blogspot.com/2020/04/errata.html

About the Book

FortiGate – Troubleshooting Guide Quick Reference presents easy to understand techniques of troubleshooting on FortiGate platform. There are many debug command examples, which explain, how to read and understand the command output. The intention of the book is not to teach you how presented technologies work. I do not explain configuration examples but if you do not feel confident to perform troubleshooting effectively, the book is for you.

Contents

1 Preface

Troubleshooting, next to network design and implementation, is one of the most demanding skill in computer network industry. One of the differences between them is time expected to complete the task. In troubleshooting you always work under time pressure. Everyone wants to know root cause and fix the incident as soon as possible. In my 20+ year career, I participated in hundreds of troubleshooting calls and I learnt the following principals regarding what you should know before you start:

- what is the problem?
- understand the network design/network diagram
- understand common protocols/applications
- well-structured troubleshooting approach
- troubleshooting commands

The first seems to be obvious, but believe me, after 10 hours on call with many people talking about their views on the problem, you can get lost. Do not be afraid to ask questions, until you have a good overview of the problem. Today's networks are complex. Applications are complex too. We, network/security engineers, are not application or database specialists. We need to know all symptoms, provided in the format: "no access between source and destination", IP addresses, transport protocol, ports, etc.

The second point on the list is the network design. By this one, I mean understanding of all components, or at least those ones, which are relevant to the traffic flow. I know that sometimes we do not have such privilege to know the network well before the incident appears. We can be engaged to join a bridge to solve an issue in totally foreign environment. That is fine, just spend more time on understanding the traffic flow.

The first two items on the list can be learnt or asked during the troubleshooting. At the beginning of the call it is totally fine to spend some time to gather all the details needed. About the next three, you should know before the problem appears. When you understand the protocols, and what should be checked first, more time can be spent on effective investigation. When you do not understand the protocol or the framework of protocols, such as IPsec, probably you jump from one place to another without any structured approach. Once you understand all protocols/applications and the troubleshooting approach, then it will just become obvious. For example, understanding IPsec phases, and their variations (i.e.: phase 1 aggressive vs. main mode), you know that it makes more sense to verify the phase1 first. If it fails, there is no way to negotiate IPsec SAs. If the Internet connection is failing every couple of minutes, tearing down the VPN tunnel, it does not make sense to test the application, which is running on top of the VPN. If something does not work make sure the underlying service is fine: Internet connection -> VPN phase1 -> VPN phase2 -> connectivity between hosts, etc. There are couple of troubleshooting methods you can follow (from bottom layer up, from the top down, follow the path, etc.), but I prefer to take final decision once I gather all information about the incident I am working on.

The last one is understanding the troubleshooting commands for a specific device. When the platform is new for you, instead of thinking about the problem, you spend most of the time on guessing or searching how to get the information from the system. It is very ineffective approach and it should be avoided. There is no need to remember all commands. It is fine to have them handy. Today, with so many systems with different syntax we must support, I do not even try to remember all commands. For this reason, I decided to write this book to help you with it. I hope you find the information presented here useful during your troubleshooting. Good luck!

2 Traffic flow

The most common problem in the network is lack of connectivity between two hosts. Probably you heard it many times: “it doesn’t work” or “I don’t have access to …”. Sometimes it is caused by a node or link failure, and sometimes because devices, like firewalls, block the connection intentionally. It could be your task to find it out. On FortiGate there are many tools, which can help you during the investigation. In the next few chapters I will show you ‘how’ and ‘when’ to use them.

2.1 Diagnose session

Before we start the troubleshooting, you should learn how to verify if the traffic is allowed or denied. There are couple of methods:

a) Verify the session table – simple view

Example 2.1

```
forti (test) # get system session list
PROTO   EXPIRE SOURCE                 SOURCE-NAT  DESTINATION         DESTINATION-NAT
tcp     3559   10.0.48.139:55506 -                192.168.40.20:27017 -
udp     132    172.16.52.11:36094 -               172.16.161.22:53 -
tcp     4      192.168.134.152:60655 -            10.2.3.45:135 -
udp     118    192.168.134.87:52408 -             10.2.3.20:53  -
tcp     9      172.16.105.252:3178 -              192.168.2.188:443 -
```

The above session list is the confirmation that the traffic is allowed by the firewall. Let’s analyze the output:

- PROTO is a protocol, and in the above example you can see TCP and UDP sessions.

- EXPIRE, this is TTL session timer (counting down). The default value for TCP is 3600 seconds.
- SOURCE - in the third column, you can see source IP along with the source port.
- SOURCE-NAT - there is no source NAT configured, so we do not see any value in the column.
- DESTINATION - in the fifth column you see the destination IP and the port number.
- DESTINATION-NAT - there is no destination NAT set for these sessions.

I explain source and destination NAT later in this chapter.

In production the list of sessions can be very long. There is no filter built-in the command, but you can use **grep** to filter out the list:

Example 2.2

```
forti (test) # get system session list | grep tcp
```

b) Verify the session table – detailed view

There is a version of the **session list** command, with much more details. I marked the most important information in the below output.

Example 2.3

```
forti (test) # diagnose sys session list



session info: proto=6 proto_state=05 duration=2 expire=0 timeout=3600 flags=00000000
sockflag=00000000 sockport=0 av_idx=0 use=5

origin-shaper=

reply-shaper=

per_ip_shaper=

ha_id=0 policy_dir=0 tunnel=/ vlan_cos=0/255

state=log may_dirty npu synced f00
```