Doc type / Publish Date
Whitepaper / 6.2.2020

Firepower NGFW Multi-Instance Performance on 4100 & 9300 Series Appliances v1.08

Security Business Group
Network Security Technical Marketing Engineering

26 June 2020

Contents

Introduction

This document covers the capabilities available in Cisco's 4100 and 9300 series appliances and their support of **multitenancy**, which is a requirement from many of our customers in one form or the other. Gartner defines Multitenancy in the following way:

"**Multitenancy** is a reference to the mode of operation of software where multiple independent instances of one or multiple applications operate in a shared environment. The instances (tenants) are logically isolated but physically integrated. The degree of logical isolation must be complete, but the degree of physical integration will vary. The more physical the integration, the harder it is to preserve the logical isolation. The tenants (application instances) can be representations of organizations that obtained access to the multitenant application (this is the scenario of an ISV offering services of an application to multiple customer organizations). The tenants may also be multiple applications competing for shared underlying resources (this is the scenario of a private or public cloud where multiple applications are offered in a common cloud environment)."

Knowing this, the preferred degree of logical isolation should also vary based on use cases. From a firewall point of view, the following are the commonly sought-after use cases and thus, isolation requirements:

1. **Policy management separation**: For policy management separation, isolation is required at the security policy level for each tenant to be able to manage different security policies for every tenant. All tenants share all hardware resources in this case, which could lead to one tenant affecting the performance of another tenant; it is essential to note that administrators for one tenant could access the traffic from other tenants.

2. **Traffic processing isolation**: For traffic processing, each tenant requires full policy and resource isolation. Any action or state of one of the tenants should not have any effect(s) on the rest of the tenants.

3. **Full management separation**: For full management separation, administrators of tenants aren't allowed access to view or edit the policies of other tenants. Therefore, full policy segregation, along with restricted access control to the policy management section, is required

4. **Routing separation**: For routing separation, isolation is required only at the routing/forwarding plane to have different routing protocols/policies across different segments but with the same security policy across all tenants.

   NOTE: FTD v6.6 supports VRF, which allows for routing plane separation.

Multi-Instance overview and internals

The 6.3 release of Cisco Firepower Threat Defense (FTD) software added support for Multi-Instance capability. With Multi-Instance support, administrators can create and run multiple independent instances of the FTD software on the same hardware appliance (the Cisco Firepower 4100 series and the Cisco Firepower 9300 series appliances support Multi-Instance). Each instance of FTD running on the hardware appliance has dedicated hardware resources, which provides the benefit of guaranteed performance per instance and that one instance cannot affect the performance of another instance. This ensures meeting the requirements for traffic processing isolation use cases. In addition to meeting the traffic processing isolation requirements, Multi-Instance also ensures that the management plane for each instance is entirely independent of the other instances.

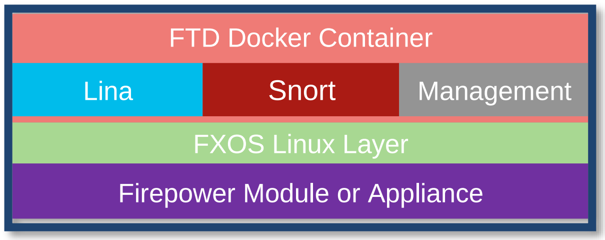
The creation of instances relies on Cisco's Firepower Extensible Operating System (FXOS) that allows the administrator to create more than one FTD logical device on the Firepower 4100 and 9300 series appliances running FXOS version 2.4 or higher. The process of creating a logical device remains the same as that of previous versions of FXOS software. What changes is that while creating an FTD logical device, the administrator will choose whether they want to create a "**native instance**" or a "**container instance**."

A **native instance** is an instance of FTD that is installed on the bare metal hardware of the appliance(s) where all available resources are used. The FTD logical devices of customers migrating from older versions of FXOS will migrate to a native instance type of logical device.

A **container instance**, on the other hand, is an instance that doesn't consume all the available hardware resources on the appliance. Instead, it only consumes the number of hardware resources that are specified by the administrator.

NOTE: To be able to create multiple FTD logical devices on the hardware appliance, the first instance (and subsequent instances) needs to be the logical device type "container instance." To create a container instance, a "Resource profile" will need to have been created by the administrators beforehand (the resource profile is what determines the number of hardware resources that must be allocated to create the container instance).

When you create a container instance, FXOS running on the supervisor uses the FXOS agent running on the security module to generate a Docker container. Then FTD is installed inside this new Docker container.



NOTE: Each Docker container has its own dedicated CPU, RAM, and hard disk that are not available to other FTD instances running in separate Docker containers on the same security module.

Hardware resources not supported in a Docker Container

In Release 6.3, the hardware resources distributed among FTD instances are CPU, RAM, and hard disk only. No other hardware resources such as Flow Offload (used for hardware trusted traffic acceleration) and Crypto (used for improving encryption and decryption performance by offloading these operations to dedicated hardware) are allotted to any of the FTD instances. Also, no hardware-based features such as TLS offload and Flow offload will be available on container instances. In Release 6.4, one instance can use the hardware crypto resources. In Release 6.5, up to 15 instances can use the hardware crypto resources (split according to the % of CPUs to which the instance has been assigned).

Network interfaces are another hardware resource that container instances need. With the creation of multiple FTD logical devices on the same security module and only a limited number of interfaces available on the appliance, FXOS now allows the sharing of interfaces between logical devices and the creation of sub-interfaces on FXOS that are allottable to the logical devices.

Factors limiting the creation of instances on an appliance

The maximum number of instances created is determined by the availability of hardware resources because each container instance requires dedicated hardware resources assigned to it.

NOTE: Instances primarily use dedicated hardware resources from the security module, but there are cases where they may need resources from the supervisor.

Distribution of hardware resources available on the supervisor

Supervisor module resources contribute to limiting the number of possible container instances because some of the hardware resources on the module get consumed for each container instance generated. Since all traffic to the security module comes through the hardware switch available on the supervisor, the availability of these two hardware resources (listed below) becomes a deciding factor for the maximum number of interfaces and, in turn, possible instances on an appliance:

- **Switch Forwarding Path entries** available on the hardware switch
- **Ingress VLAN group entry** counts

The hardware switch on the supervisor has a fixed number of **Switch Forwarding path entries** available to program the path from physical interfaces available on the supervisor to logical interfaces on specific container instances. Typically, a maximum of 1021 switch forwarding path entries are available on all appliances, with a few of these entries consumed to create a path between a physical interface and a non-shared logical interface assigned to an instance. In other words, you have a limited number of non-shared interfaces available for allotment to the instances.

Additionally, the number of switch forwarding path entries becomes more critical when sharing interfaces between instances. In that case, inter-instance traffic flows through the shared interfaces via the hardware switch on the supervisor. For this, the supervisor needs to program a path between every pair of instances using every pair of shared interfaces between them. This exponentially increases the consumption of switch forwarding path entries and thereby limits the number of possible instances.

Also, the **Ingress VLAN group entry** count has the potential to restrict the maximum number of VLAN sub-interfaces created on the supervisor, which may restrict the number of instances created overall. The VLAN group entry table tracks ingress VLAN IDs on the sub-interfaces configured on a physical interface on the supervisor. The maximum number is 500 entries, and at least one entry from this table is consumed for every VLAN sub-interface created.

NOTE: In Firepower 9300 series appliances, the supervisor hardware-based limits apply for the cumulative number of instances created across all three security modules on the chassis.

Distribution of hardware resources available on the security module

The creation of the actual Docker container(s) occurs on the security module. The hardware resources that are available on the security module are then allocated independently amongst the various container instances. The hardware resources are:

- CPU cores
- RAM
- Disk space

Since each of these resources are available in limited quantities on the security module, the number or amount of these resources dictates the maximum number of possible instances on the security module.

Disk space as the limiting factor

Disk space allocated to an instance has a fixed amount and does not depend on the size of the instance. Every instance, irrespective of its size, obtains about 48GB of dedicated disk space. This way, if there are a sufficient number of CPUs and enough RAM, the maximum number of instances possible would be **<DISK_SIZE>/48**, where:

- **DISK_SIZE** is the size of the hard disk on the security module available for allotment and
- **48GB** is the amount of disk space consumed for the instances' FXOS infrastructure, host operating system, and Docker runtime environment.

Size of an instance—resource profile

The administrator is responsible for identifying the distribution ratio and assignment of security module hardware resources to an instance - which determines the instance's overall size. To assign hardware resources to an instance, the administrator uses the **Resource Profile** during its creation.

The Resource Profile specifies:

- The number of logical CPU cores needed for the instance (when a resource profile is associated with an instance, the specified number of logical CPU cores are assigned to it)
- The amount of RAM allotted to an instance, which depends upon the resource profile or number of CPUs assigned to the instance

The ratio of the total number of logical CPU cores available on the security module to the logical CPU cores assigned to an instance is proportional to the total amount of RAM available on the security module to the amount of RAM assigned to the instance. This way, the resource profile becomes the only parameter that controls the size of an instance. To illustrate, if a security module had 18 CPU cores available and 120MB of RAM, and we created an instance that consumed 6 logical CPU cores, that instance would consume 40MB of RAM (6/18 = 1/3, 1/3 x 120MB = 40MB).

Resource profile limits

As described above, the resource profile defines the number of logical CPU cores needed for an instance. However, there are a few restrictions for choosing a valid value for the resource profile:

- The security module requires a **minimum of two** logical CPU cores for FXOS needs. In other words, the total number of application CPU cores available for allotment from the security module to instances is two logical CPU cores less than the total number of logical CPU cores available on the security module.
- The maximum value selectable in the resource profile for the number of CPUs assignable to an instance is the total number of application CPU cores available on the security module.
- The number of CPU cores assigned to an instance must be an even number.
- Instances require a minimum of six logical CPU cores (two Mgmt, two Data, two Snort)
- Before the FXOS 2.7.1 release, creating a resource profile with eight logical CPU cores was not allowed (for FXOS 2.7.1 and later, this restriction does not exist).

Maximum possible instances on a security module

Based on all the information above, the main factors deciding the maximum number of possible instances on a security module are:

- The number of logical CPU cores available on the security module
- The size of the hard disk on the security module
- The number of possible interfaces on the supervisor
- A combination of the number of shared interfaces and the number of instances sharing those interfaces

By omitting the last two factors above and focusing only on the logical CPU cores and hard disk space, the maximum number of possible instances on the security module is easily calculated – following these two factors:

- Total CPU cores divided by at least **six** cores per instance
- Total Disk space divided by **48GB** of required disk space per instance

Based on this, the following table shows the maximum number of instances possible on the Firepower 4100 and 9300 series platforms:

Platform	CPU Cores Available for Instances	Total Disk Space	Maximum FTD Instances
Firepower 4112	22	400 GB	3
Firepower 4115	46	400 GB	7
Firepower 4125	62	800 GB	10
Firepower 4145	86	800 GB	15
Firepower 4110	22	200 GB	3
Firepower 4120	46	200 GB	3
Firepower 4140	70	400 GB	7
Firepower 4150	86	400 GB	7
Firepower 9300 SM-40	78	1.5 TB	13
Firepower 9300 SM-44	94	1.5 TB	15
Firepower 9300 SM-56	110	1.5 TB	18
Firepower 9300 SM-24	46	800GB	7
Firepower 9300 SM-36	70	800GB	11
Firepower 9300 SM-44	86	800GB	14

Factors affecting container instance performance

Previously, we discussed factors that determine how many FTD instances are possible on an appliance. Now, we consider the factors that affect the performance of an FTD instance.

Disabled hardware acceleration support

One of the first and more obvious things that affect the performance of specific types of flows is the temporary lack of support for hardware acceleration within the container instances in Release 6.3. Two hardware acceleration components are available on the Firepower 4100 and 9300 series appliances:

- The **Crypto Engine** provides encryption and decryption acceleration to support TLS/IPsec VPN for site to site and RA VPN and TLS traffic inspection at scale. Container instances can use hardware crypto acceleration in Release 6.4 (one instance only can use the hardware crypto), and Release 6.5 (up to 15 instances can share the hardware crypto). However, in Release 6.3, the hardware crypto acceleration is unassignable to a container instance and is not distributable between container instances. Lack of access to the hardware crypto engine impacts performance of any instance requiring crypto.

- The **Hardware Flow Offload Engine** helps provide very high throughput and extremely low latencies for elephant flows

NOTE: Because of the absence of these hardware components in container instances, there is a noticeable reduction in the max performance of elephant flows compared to running FTD in native mode until hardware offload can be used by an instance (future release).

Inter-instance communications

If instances use shared interfaces, then communication between those instances happens through the hardware switch on the supervisor module, reducing the need for traffic to leave the appliance and be forwarded back into it externally. However, it does contribute to the traffic that flows over the backplane.

NOTE: The backplane connecting a security module to the supervisor on all Firepower 4100 and 9300 series appliances consists of 2x40Gig interfaces with the maximum data transfer capacity between the supervisor and security module being 80Gbps (except for 4110 where its backplane is just 1x40Gig interface). This means that all the instances created on the security module share this bandwidth not only to process traffic coming in and out of the instances from outside the box, but also to process traffic flowing between instances that don't need to leave the appliance.

Performance overhead of running within Docker containers

Each independent FTD instance runs within a Docker container. It is well-known that the performance of an application running within one is almost identical to the performance of the same application running natively on bare metal (several independent tests validate this fact, and several published white papers explain this in detail).

In tests performed internally, the results match the global observations. When creating a container instance on our hardware that consumes all the hardware resources available, the performance delivered by that instance is nearly identical to the performance created on a native instance.

NOTE: The performance of smaller individual instances will depend on the number of CPU cores assigned to it. Among the CPU cores allotted to the instance, two CPU cores are assigned for management processes, with the rest divided among data plane threads and Snort processes in a ratio between 1:2 to 1:1.

Cost of independent instances

Almost zero performance overhead is incurred in running FTD instances in Docker containers. Multi-Instance aims to provide complete independence between instances, which ensures that no container instance can affect another under any circumstances. Also, this requires that management applications such as Firepower Management Center (FMC), and SSH connections to the CLI are fully independent and able to reach the container instance even if other instances are overloaded or down. For this, as mentioned in the previous section, two CPU cores from every instance are dedicated to running management processes for that instance to guarantee the independent and predictable manageability of each instance.

However, this independent and predictable management of each instance has an associated cost: the allocation of CPU cores for management. Every core that is not running either a data plane thread or a Snort process is effectively reducing the total traffic processing performance of the appliance.

So, although the performance of a single container instance consuming all the hardware resources available on the security module is equal to the performance of a native instance, the same is not true as we increase the number of instances. To illustrate, if hypothetically the throughput is 10Gbps from a native FTD instance, then 10Gbs is the expected throughput of a container instance if allocating all the hardware resources to that single container instance.

Conversely, in creating three container instances on the same hardware that share all the hardware resources, we cannot get a cumulative throughput of 10Gbps because the total number of CPU cores assigned to management processes in the case of three container instances will be six CPU cores compared to two CPU cores in the case of a single instance. This means four additional CPU cores are not used for traffic processing, thus reducing the overall throughput of the box.

Overall, the price paid for independence is the number of container instances needed. As the number of instances goes up, the overall throughput of the box decreases because each instance requires two CPU cores for management processes.

Estimating the performance of a specific instance

While creating an instance, it is essential to estimate the performance obtainable from the instance correctly. When we talk about the performance of a firewall, the three central values discussed are:

- Throughput of the firewall
- Maximum concurrent connections supported by a firewall
- Maximum connections per second supported by the firewall

To calculate these performance values for specific instance size, the formulas that describe the estimation of performance values, the variables used are as follows:

- **<MAX_CPU>** - represents the total number of logical CPU cores available on the particular hardware platform
- **<RE_PROFILE>** - represents the number of logical CPU cores selected in the resource profile associated with the container instance
- **<MAX_CON_CONN>** - represents the maximum number of connections supported on the platform when FTD is installed as a native instance (this is obtainable from the datasheets)
- **<MAX_THRU>** - represents the maximum throughput of a specific appliance when FTD is installed on it, in native mode (this is also obtainable from the datasheets)

Estimating the throughput of a container instance

In general, the performance of individual Snort processes limits the throughput on FTD. To calculate the maximum throughput of an FTD instance, multiply the throughput of a single Snort process on the specific hardware by the total number of Snort processes running on it.

NOTE: You can determine the number of Snort processes running by entering the "show asp inspect-dp snort" command.

The following table provides the number of data plane cores and Snort cores for the different Firepower platforms:

Platform	Total Available CPU Cores	Management Cores	Lina Cores	Snort Cores
Firepower 4112	22	2	8	12
Firepower 4115	46	2	16	28
Firepower 4125	62	2	24	36
Firepower 4145	86	2	32	52
Firepower 4110	22	2	8	12
Firepower 4120	46	2	20	24
Firepower 4140	70	2	32	36
Firepower 4150	86	2	36	48
Firepower 9300 SM-40	78	2	32	44
Firepower 9300 SM-48	94	2	40	52
Firepower 9300 SM-56	110	2	44	64
Firepower 9300 SM-24	46	2	20	24
Firepower 9300 SM-36	70	2	32	36
Firepower 9300 SM-44	86	2	36	48

You can calculate the maximum throughput achievable on a container instance using this formula:

Maximum throughput =
(<MAX_THRU>/<SNORT_CORE_COUNT>)* <SNORT_CORE_COUNT_IN>

Where:

- <SNORT_CORE_COUNT> is the number of logical CPU cores running Snort on a native FTD instance (obtainable from the table above or if you have access to a similar box running FTD as a native instance).
- <SNORT_CORE_COUNT_IN> is the number of logical CPU cores running Snort when you choose a specific resource profile (obtainable in Appendix 1).

The following tables provide the max throughput (Firewall + AVC: 1024B) values for three cases on each available security module for the 9300 and each appliance from the 4100 series appliances. The three cases covered are:

- Maximum throughput when we create one container instance consuming all resources
- Throughput of the smallest size instance on the security module
- Throughput of an instance that occupies half the resources

These values provide the high, low, and mid-performance guidelines for each platform.

Platform	Max throughput of the largest instance (Cores)	Max throughput (Instance size 6 Cores)	Max throughput of the mid-sized instance (Cores)
Firepower 4112	14 Gbps (22)	2.33 Gbps	9.33 Gbps (14)
Firepower 4115	27 Gbps (46)	1.93 Gbps	15.43 Gbps (26)
Firepower 4125	40 Gbps (62)	2.22 Gbps	20 Gbps (32)
Firepower 4145	53 Gbps (86)	2.04 Gbps	28.53 Gbps (46)
Firepower 4110	13 Gbps (22)	2.17 Gbps	8.66 Gbps (14)
Firepower 4120	22 Gbps (46)	1.83 Gbps	12.83 Gbps (26)
Firepower 4140	32 Gbps (70)	1.78 Gbps	16 Gbps (36)
Firepower 4150	45 Gbps (86)	1.88 Gbps	24.37 Gbps (46)
Firepower 9300 SM-40	54 Gbps (78)	2.45 Gbps	29.45 Gbps (42)
Firepower 9300 SM-48	64 Gbps (94)	2.46 Gbps	34.46 Gbps (50)
Firepower 9300 SM-56	70 Gbps (110)	2.19 Gbps	37.18 Gbps (58)
Firepower 9300 SM-24	25 Gbps (46)	2.08 Gbps	14.58 Gbps (26)
Firepower 9300 SM-36	34 Gbps (70)	1.89 Gbps	17 Gbps (36)
Firepower 9300 SM-44	50 Gbps (86)	2.08 Gbps	27.08 Gbps (46)

Estimating the maximum concurrent connections supported by an instance

The estimation of the maximum concurrent connections supported on a specific instance is straightforward because this value almost entirely depends on the size of RAM allotted to the container instance.

The following formula provides the maximum concurrent sessions value for a container instance:

Maximum concurrent sessions = (<RE_PROFILE> * <MAX_CON_CONN>)/<MAX_CPU>

Benefits of Multi-Instance solution

Apart from the obvious benefit of instantiating several independent FTD logical devices on the same hardware appliance, the significant advantages of the Multi-Instance solution arise from the fact that the container instances are entirely independent.

These container instances have dedicated hardware assigned to them. As a result of this independence, the following are the major benefits of the Multi-Instance solution:

- **Hardware-level traffic processing isolation:** Since each container instance has dedicated hardware allotted to it, one container instance can never affect the performance of another container instance. Also, one tenant cannot access the traffic of another tenant running in a different container.
- **Hardware-level fault isolation:** Each container runs software independently, which means that problems in one container instance only affect that container. In contrast, the rest of the container instances continue to run without any problems.
- **Independent software version management:** Each container is fully isolated at the hardware level and can run different versions of FTD software in separate containers.
- **Independent upgrades and restarts:** Each container is fully isolated at the hardware level, so FTD container instances can be independently upgraded and restarted without affecting other container instances.
- **Full management isolation:** Each FTD container instance acts as a fully independent FTD NGFW that can be managed as independent devices either from the same FMC or from different FMCs.
- **Full feature parity between a container and native instance(s):** Every feature that is not dependent on specific hardware and supported in native instance mode is supported in container instance mode.

Recommendations for getting the best out of Multi-Instance

Based on the architecture of the Multi-instance solution, there are a few frequently asked questions and recommendations for getting the best out of container instances:

Do I create a native instance or container instance?

This decision depends entirely on the probability of your having more instances in the future. If you need only one instance, then you may have to decide if you want to run that instance as a native instance or a container instance, since the performance of a native instance is almost equal to the performance of a container instance.

If you expect to have more than one instance on the same security module in the future, then it might be easier to start with one container instance consuming all the hardware resources. If, at a later point, when you are ready to create another instance, the resource profile associated with the existing container instance can be changed to allot fewer CPU cores to the container instance, thus freeing up resources for the creation of more container instances.

NOTE: Keep in mind that resizing a container instance requires rebooting FTD and may lead to temporary traffic outages, which can be avoided by configuring instance-level HA. Also, remember that by decreasing the size of an instance, upon reboot, the policies may not fit in the new smaller RAM.

If you are positive that you will never have multiple instances running on the same security module, then it is recommended to run FTD as a native instance. The reason for this is that, in the current release, a container instance cannot use the hardware acceleration capabilities like flow offload and hardware TLS decryption.

NOTE: In the future, when we start supporting the allotment of hardware acceleration capabilities to container instances, the recommendation will change to a container instance because they allow the flexibility to shrink the size of an instance if needed.

Sub-interface creation on FMC or FXOS?

With the Multi-Instance solution, an administrator has the option to assign physical interfaces to a container instance and then create sub-interfaces or VLAN interfaces in the FMC. Alternatively, one can also create sub-interfaces or VLAN interfaces on FXOS and assign the sub-interfaces to container instances.

If you have fewer instances, need to create a large number of sub-interfaces for each interface, and all the sub-interfaces will be part of the same container instance, then it makes sense to assign the physical interface to a container instance and create sub-interfaces from the FMC. Doing so avoids hitting the ingress VLAN Group table entries limit or switch forwarding path entries table limits on the supervisor.

If you must create sub-interfaces that are shared across instances or are planning to assign them as dedicated interfaces to different container instances, then your only option is to create these sub-interfaces in FXOS.

Shared interface-related recommendations

In general, it is recommended not to have many interfaces shared between many container instances. However, in many practical, real-world scenarios, sharing an interface often becomes necessary. In these cases, the sharing of a single interface between container instances should be limited to as few as possible.

If your needs dictate sharing an interface across several instances, it is preferred to share a sub-interface rather than a physical interface. Additionally, for the best scalability, it is recommended to share sub-interfaces from a single parent rather than sharing sub-interfaces from different parent interfaces.

Sharing a failover/stateful interface for inter-instance HA configuration

High availability (HA) configuration is supported between instances, and, in general, NGFW is rarely deployed in production networks without HA configuration (should FTD be deployed in NGIPS mode, HA might be optional).

If several instances exist on the same appliance, and each instance HA pair requires a dedicated logical interface as its HA link and state link, assigning a physical interface to each pair is almost impossible and impractical.

NOTE: It is recommended to create a port-channel interface with two physical member interfaces for redundancy, and then to create one pair for VLAN sub-interfaces for each HA pair.

Are all resource profile values valid?

Theoretically, all resource profile sizes (starting from 6 to the maximum CPU count for the hardware) are valid if they adhere to the validity rules mentioned in the Resource Profiles section. But, as mentioned above, the expectation when creating a container instance is that at least two container instances will be created on the security module. With that condition, any resource profile value that does not leave enough CPU cores to create another instance becomes a logically invalid value. Also, it is best not to leave CPU cores unassigned and to use all available CPU cores for creating container instances, unless there is a plan to utilize them later.

Appendix 1: Data plane and snort core distribution

The following table specifies the number of logical CPU cores assigned to data plane threads and snort processes, respectively, for a given resource profile value:

Instance Size	4110		SM24/4120		SM36/4140		SM44/4150	
	Data Plane Cores	Snort Cores	Data Plane Cores	Snort Cores	Data Plane Cores	Snort Cores	Data Plane Cores	Snort Cores
6	2	2	2	2	2	2	2	2
8	2	4	2	4	2	4	2	4
10	4	4	4	4	4	4	4	4
12	4	6	4	6	4	6	4	6
14	4	8	6	6	6	6	6	6
16	6	8	6	8	6	8	6	8
18	6	10	8	8	8	8	8	8
20	8	10	8	10	8	10	8	10
22	8	12	10	10	10	10	8	12
24			10	12	10	12	10	12
26			10	14	12	12	10	14
28			12	14	12	14	12	14
30			12	16	14	14	12	16
32			14	16	14	16	14	16
34			14	18	16	16	14	18
36			16	18	16	18	14	20
38			16	20	16	20	16	20
40			18	20	18	20	16	22
42			18	22	18	22	18	22
44			20	22	20	22	18	24
46			20	24	20	24	18	26
48					22	24	20	26
50					22	26	20	28
52					24	26	22	28
54					24	28	22	30
56					26	28	24	30
58					26	30	24	32
60					28	30	26	32
62					28	32	26	34
64					30	32	26	36
66					30	34	28	36
68					32	34	28	38
70					32	36	30	38
72							30	40
74							30	42
76							32	42
78							32	44
80							34	44
82							34	46
84							36	46
86							36	48

Instance Size	4112		4115		4125		4145	
	Data Plane Cores	Snort Cores	Data Plane Cores	Snort Cores	Data Plane Cores	Snort Cores	Data Plane Cores	Snort Cores
6	2	2	2	2	2	2	2	2
8	2	4	2	4	2	4	2	4
10	4	4	4	4	4	4	4	4
12	4	6	4	6	4	6	4	6
14	4	8	4	8	4	8	4	8
16	6	8	6	8	6	8	6	8
18	6	10	6	10	6	10	6	10
20	8	10	6	12	8	10	8	10
22	8	12	8	12	8	12	8	12
24			8	14	8	14	8	14
26			8	16	10	14	10	14
28			10	16	10	16	10	16
30			10	18	12	16	10	18
32			12	18	12	18	12	18
34			12	20	12	20	12	20
36			12	22	14	20	14	20
38			14	22	14	22	14	22
40			14	24	16	22	14	24
42			14	26	16	24	16	24
44			16	26	16	26	16	26
46			16	28	18	26	16	28
48					18	28	18	28
50					20	28	18	30
52					20	30	20	30
54					20	32	20	32
56					22	32	20	34
58					22	34	22	34
60					24	34	22	36
62					24	36	24	36
64							24	38
66							24	40
68							26	40
70							26	42
72							26	44
74							28	44
76							28	46
78							30	46
80							30	48
82							30	50
84							32	50
86							32	52

Instance Size	SM 40		SM 48		SM 56	
	Data Plane Cores	Snort Cores	Data Plane Cores	Snort Cores	Data Plane Cores	Snort Cores
6	2	2	2	2	2	2
8	2	4	2	4	2	4
10	4	4	4	4	4	4
12	4	6	4	6	4	6
14	6	6	6	6	6	6
16	6	8	6	8	6	8
18	6	10	8	8	6	10
20	8	10	8	10	8	10
22	8	12	8	12	8	12
24	10	12	10	12	10	12
26	10	14	10	14	10	14
28	12	14	12	14	10	16
30	12	16	12	16	12	16
32	12	18	14	16	12	18
34	14	18	14	18	14	18
36	14	20	14	20	14	20
38	16	20	16	20	14	22
40	16	22	16	22	16	22
42	16	24	18	22	16	24
44	18	24	18	24	18	24
46	18	26	20	24	18	26
48	20	26	20	26	18	28
50	20	28	20	28	20	28
52	22	28	22	28	20	30
54	22	30	22	30	22	30
56	22	32	24	30	22	32
58	24	32	24	32	22	34
60	24	34	26	32	24	34
62	26	34	26	34	24	36
64	26	36	28	34	26	36
66	28	36	28	36	26	38
68	28	38	28	38	28	38
70	28	40	30	38	28	40
72	30	40	30	40	28	42
74	30	42	32	40	30	42
76	32	42	32	42	30	44
78	32	44	34	42	32	44
80			34	44	32	46
82			34	46	32	48
84			36	46	34	48
86			36	48	34	50
88			38	48	36	50
90			38	50	36	52
92			40	50	36	54
94			40	52	38	54
96					38	56
98					40	56
100					40	58
102					40	60
104					42	60
106					42	62
108					44	62
110					44	64

Appendix 2: Throughput instance calculation on a 4145

An example of calculating the Firewall + AVC + NGIPS throughput obtainable from an instance with a resource profile value of 20 that is running on a 4145.

The formula to obtain the throughput value is:

Maximum throughput (1024B) =
(<MAX_THRU>/<SNORT_CORE_COUNT>)* <SNORT_CORE_COUNT_IN>

Put the values into the formula:

<MAX_THRU> = 45Gbps (obtained from the 4145 datasheet)

<SNORT_CORE_COUNT> = 45 (obtained from Table 2 in this document)
<SNORT_CORE_COUNT_IN> = 10 (obtained from the table in Appendix 1 for instance size 20 on 4145)

Replacing the values in the formula gives us the following:

Maximum throughput = (2/36)*10 = 7.5Gbps

As a result, the Firewall + AVC + NGIPS (1024B) maximum throughput expected from an instance with a resource profile value of 20 running on a 4145 is 7.5Gbps.