
# AADL Model Based Cyber Resiliency Key Performance Parameter Determination

## AADL/ACVIP USER DAY
## June 2, 2022

Ricky Rodriguez, Ph.D.

**DISTRIBUTION STATEMENT A**: Approved for public release; distribution is unlimited.

This work was associated with U.S. Army DEVCOM AvMC Contract # W911W6-19-2-0005.

RTX Approved for Public Release – eTPCR RIS-019707

# Agenda

- Architecture Analysis & Design Language (AADL) Attack Modeling Overview
- Denial-of-Service (DoS) Attack AADL Model
- DoS Attack Case Study
- Questions

# AADL Attack Modeling

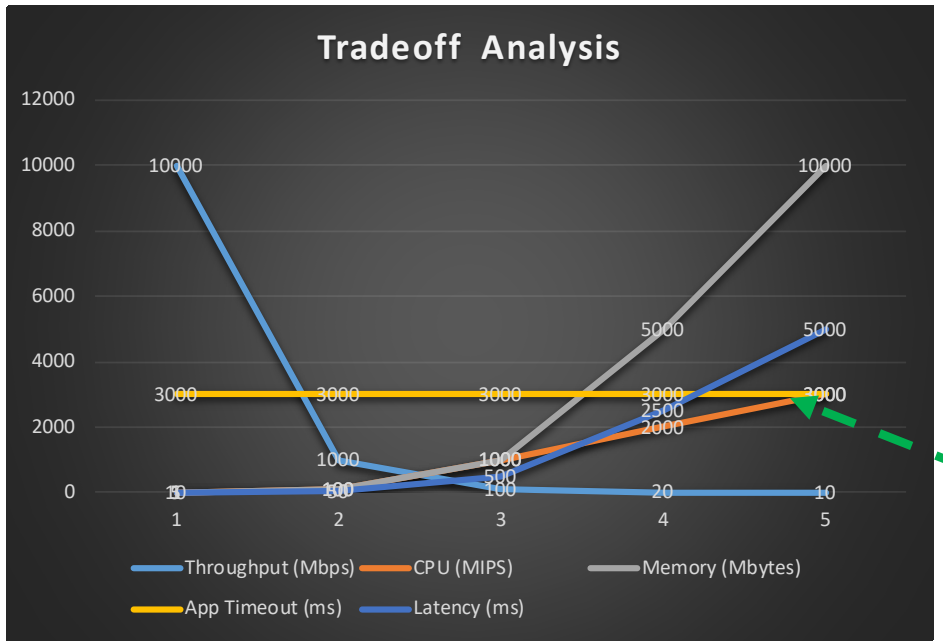
- AADL enhances attack analysis by including behavior modeling
  - Helps measure impact of attacks on resources such as Memory, CPU, Bandwidth, and Latency
  - Leverages Resource Budget Allocation and Latency Analyses
  - Output enables trade-off analyses
- Reusable and Extensible Process
  - Can accommodate many types of attacks from CAPEC and other sources
  - Approach can be overlaid on other Models
- Enables generation of Cyber Survivability KPPs
  - Helps implement Cyber Resiliency “Withstand” within specific bounds
  - Increases options for implementing cybersecurity
- Enhances Defense-in-Depth Layered Approach
  - Provides quantitative data to determine proper countermeasures implementation (e.g., No. Concurrent Connections, Sync Timeout)

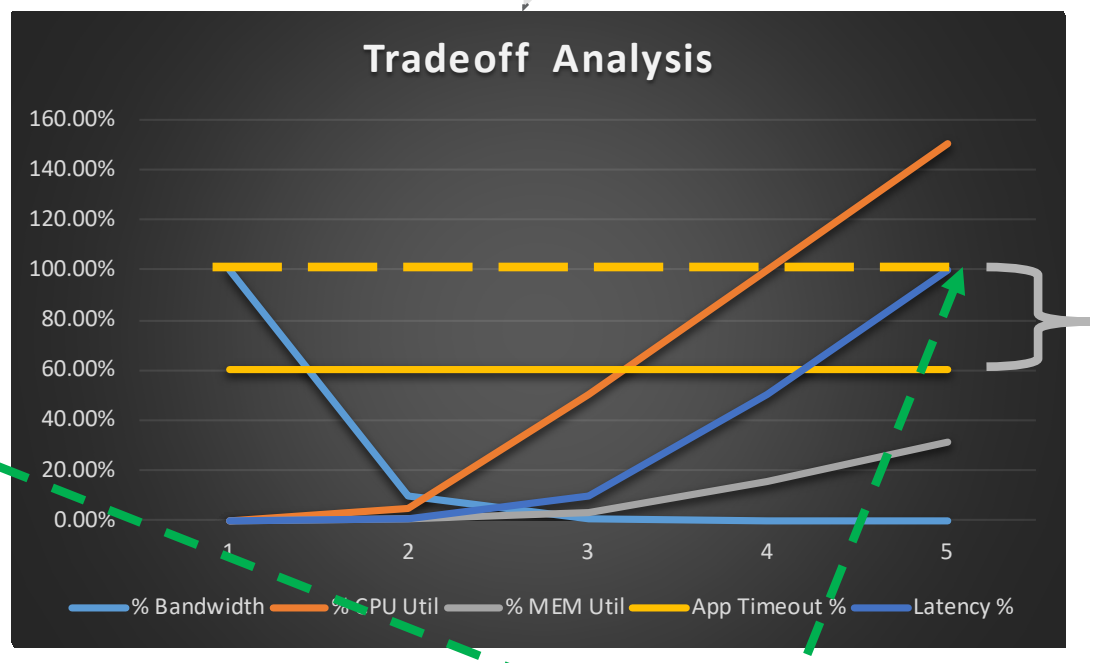
CAPEC = Common Attack Pattern Enumeration and Classification

KPP = Key Performance Parameters

# AADL - End Goal – Denial-of-Service (DoS - Notional)

T1 → T2



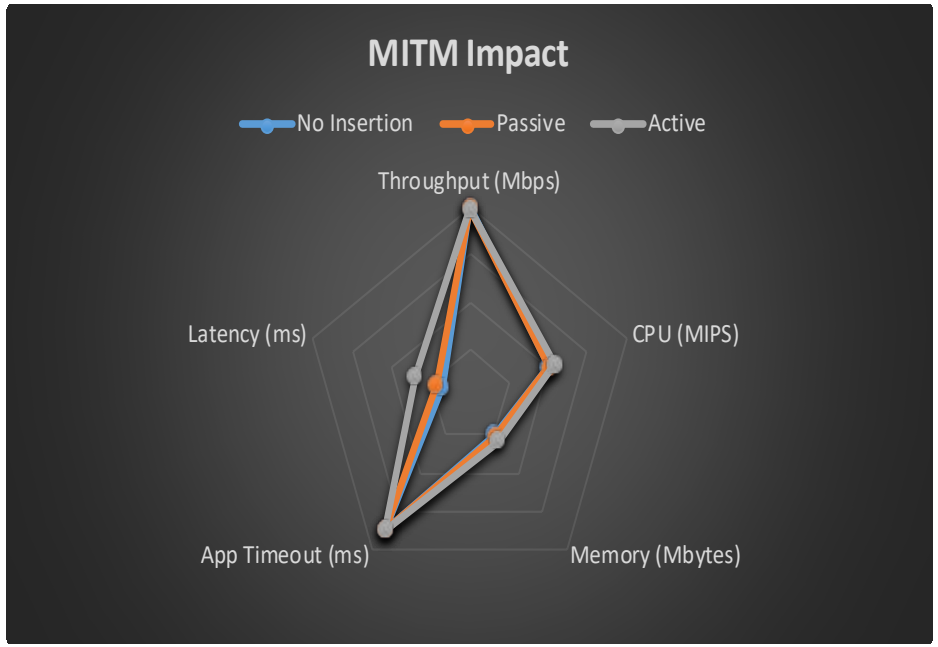

ΔTimeout

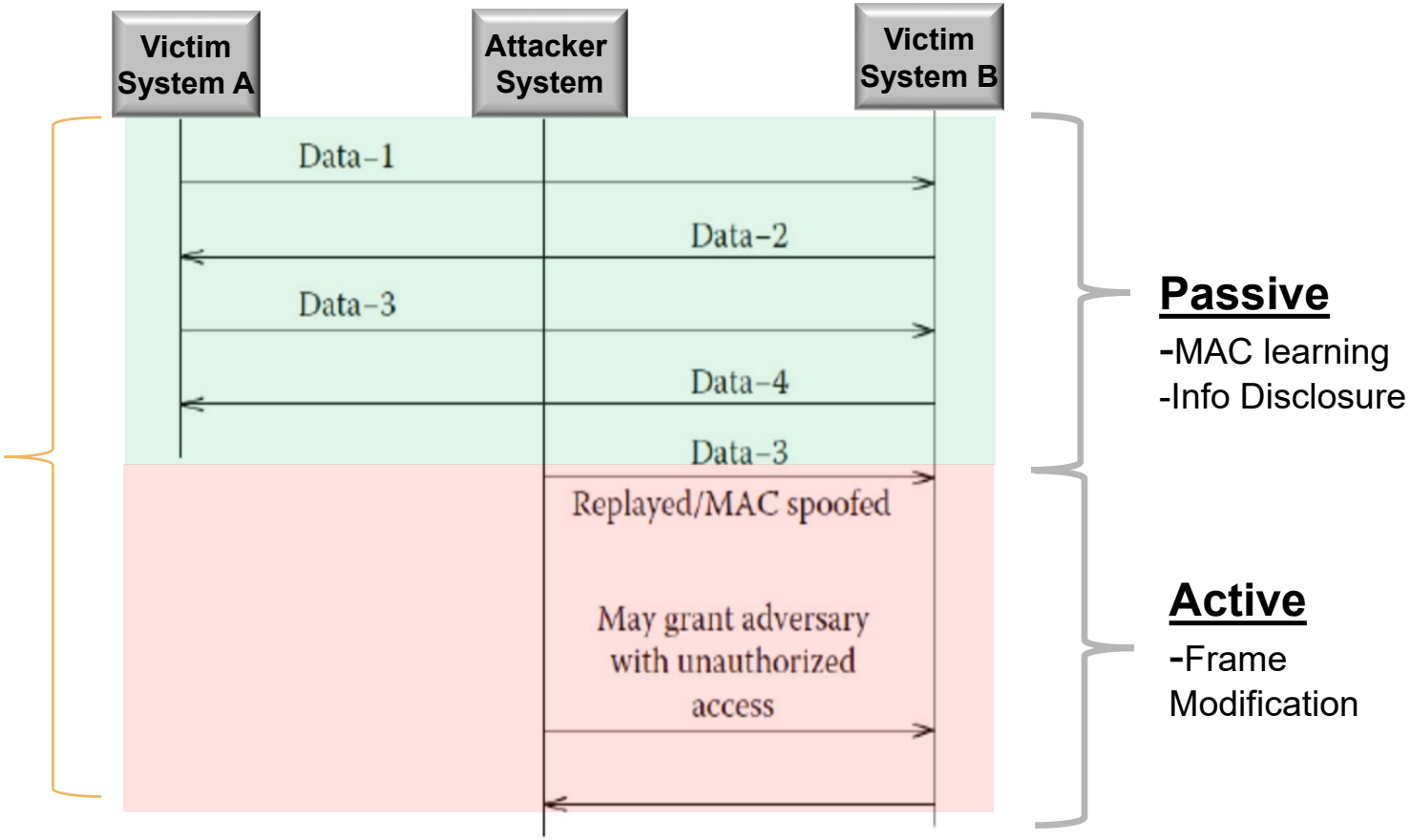
Susceptibility Reduction of Cyberattack:

- While under attack, **<u>latency can be tolerated up to a certain value</u>**.
- Increase App Timeout to maximum possible
  - **Reduces susceptibility** to DoS attack
  - **Mitigates** DoS attack
- KPPs can be determined to enable True Cyber Resiliency

Raytheon Intelligence & Space

# AADL- End Goal – Man-In-The-Middle (MITM - Notional)



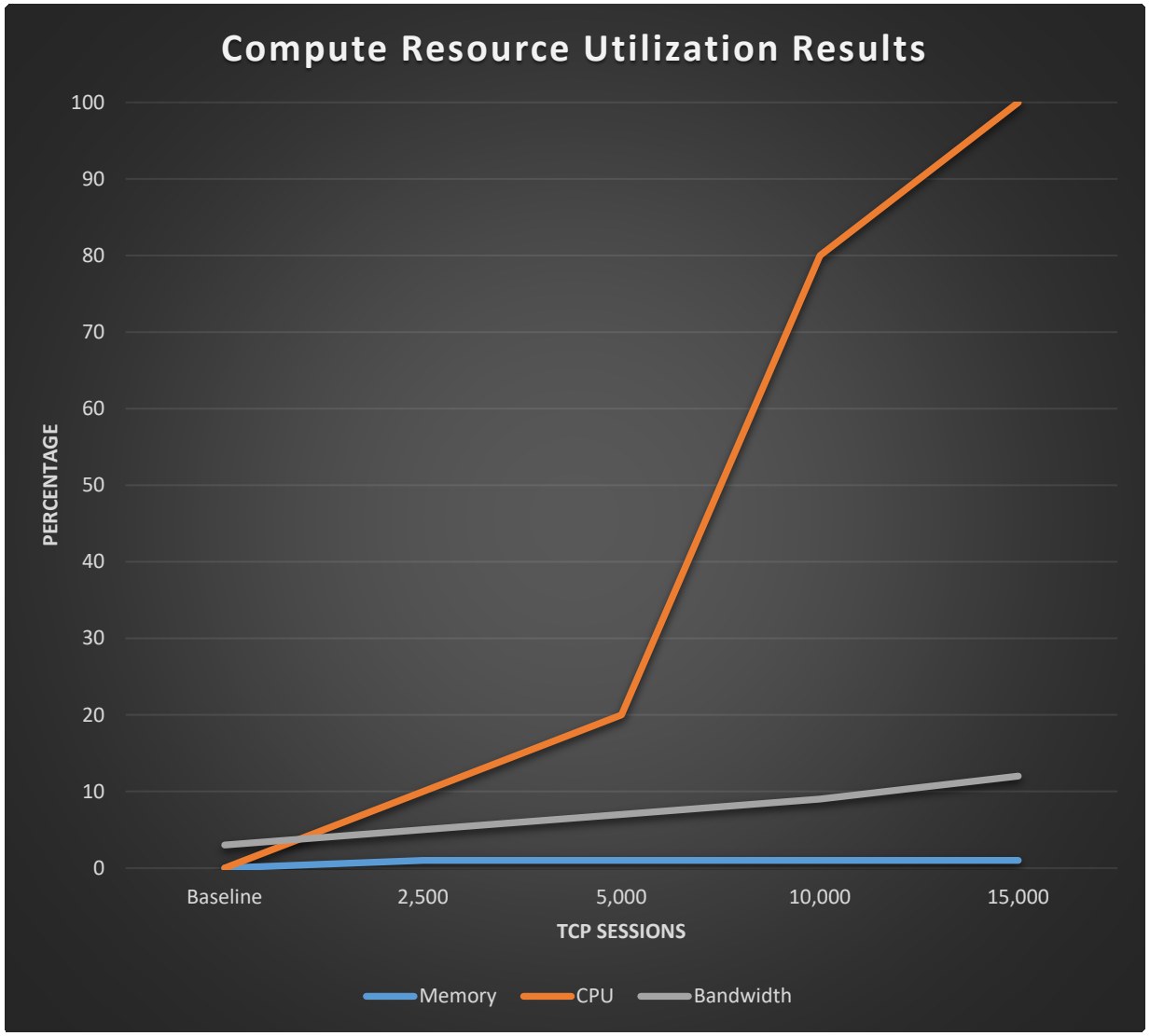




MITM detection capability:

- ΔLatency implies change
- Passive vs Active ΔLatency analysis
- Anomaly detection capability

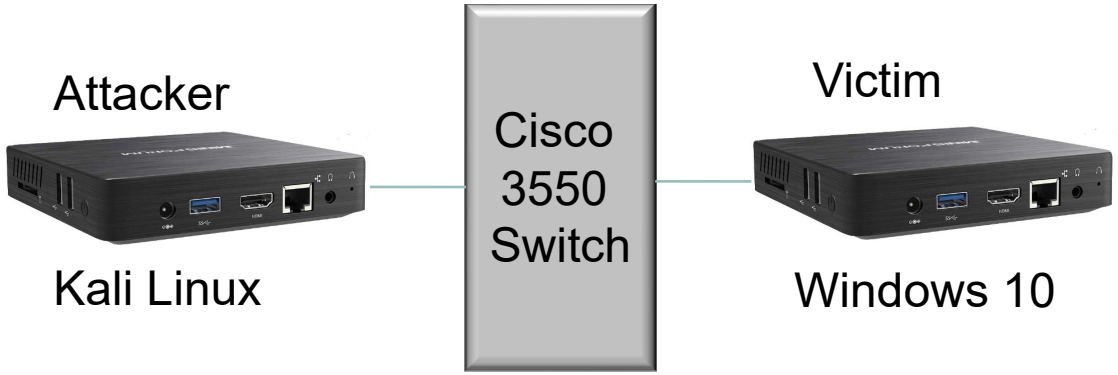
MAC = Media Access Control

# Measurements from Test Lab - DoS



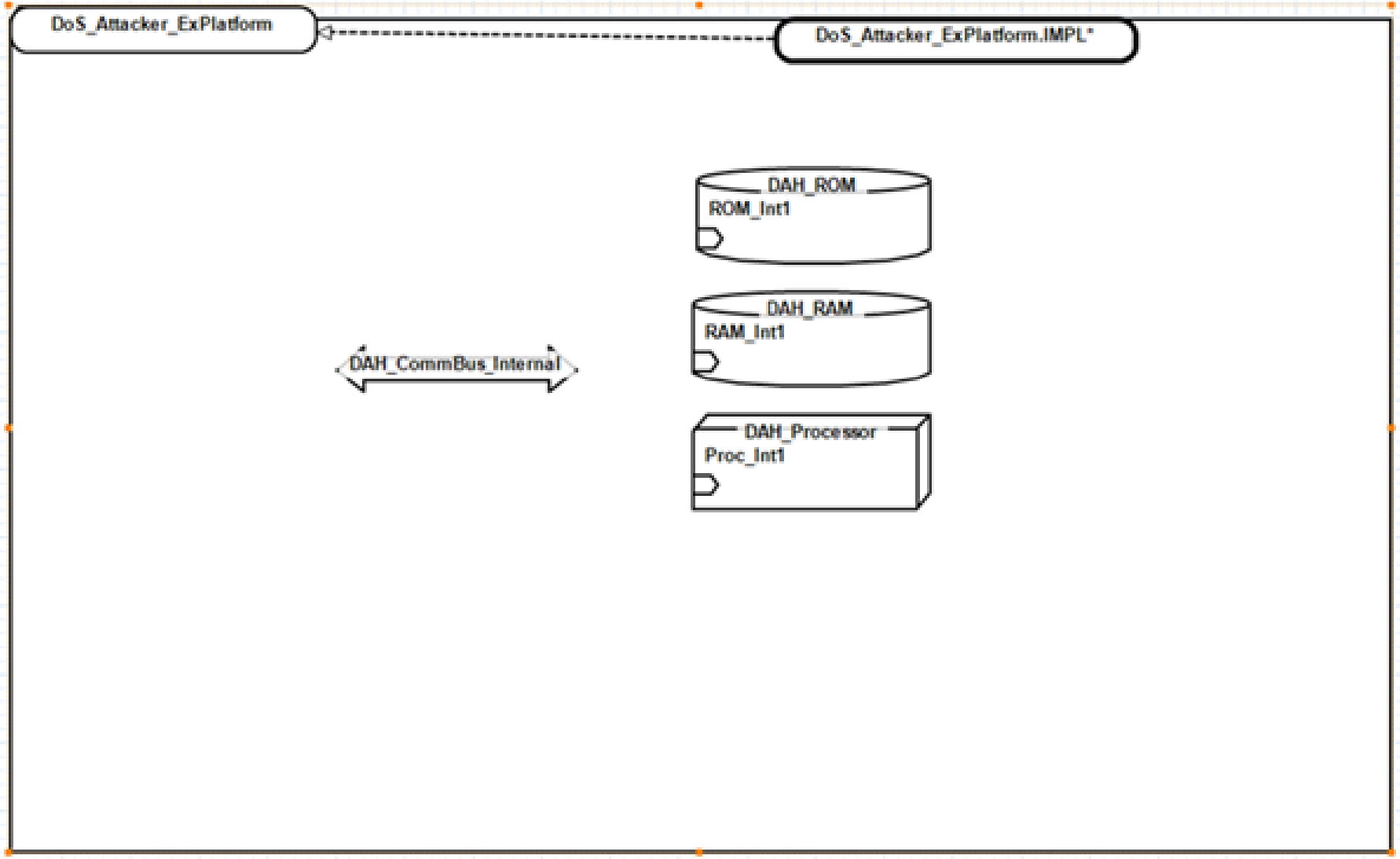

Product Specification


| Processor and Memory | | | |
|---|---|---|---|
| OS | Support Windows 10 system | CPU | Intel Atom x5-Z8350 Processor (2M Cache, up to 1.92 GHz) |
| System Disk | Windows(C:) 64GB | Installed RAM | DDR3 4GB |
| Processor Graphics | Intel HD Graphics 400 | Ethernet | 1000Mbps LAN |
| WIFI | IEEE 802.11a/b/g/n，2.4G+5.8G | Bluetooth | BT 4.1 |



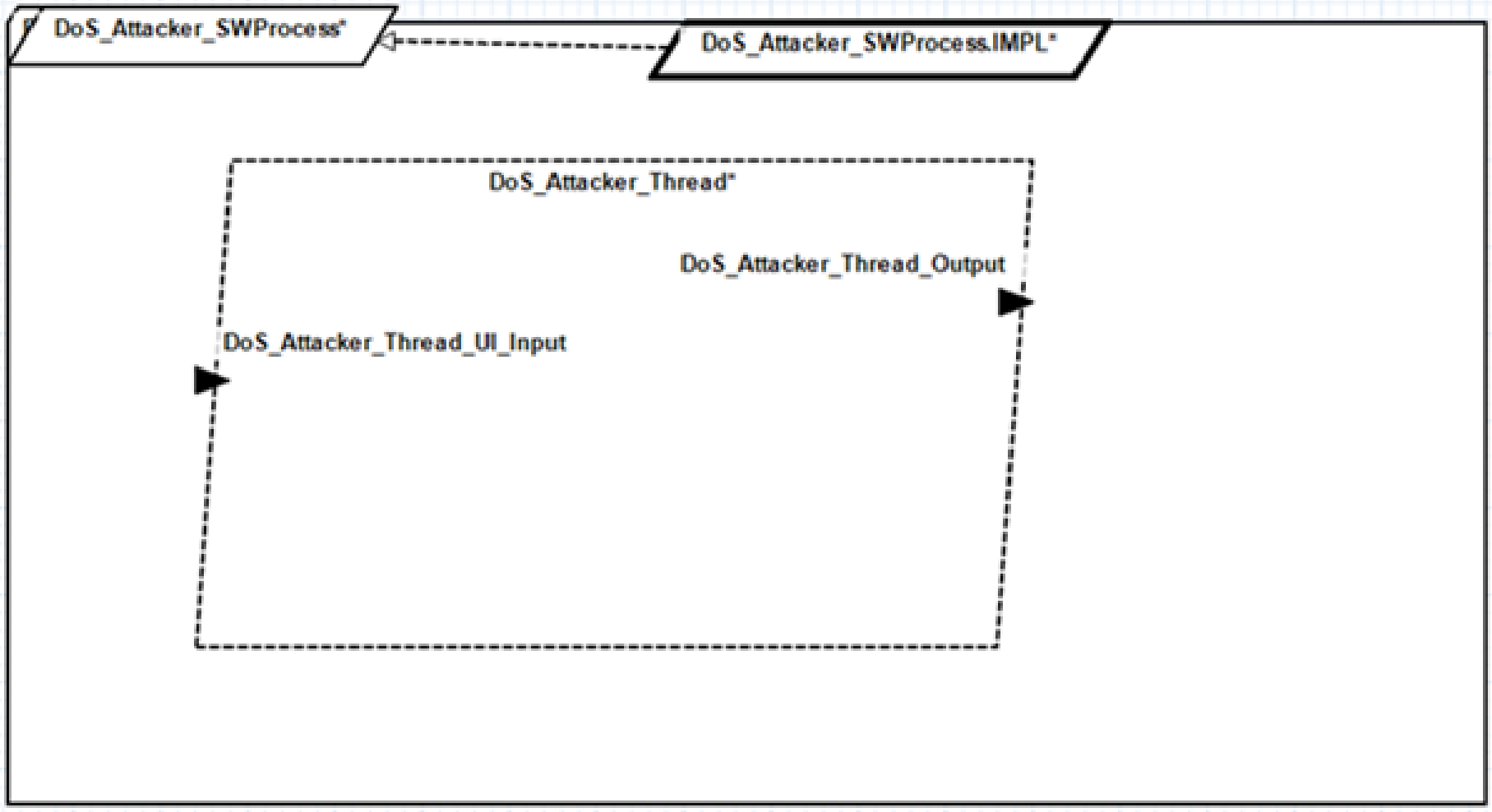

- Memory utilization was minimal at max CPU utilization
- CPU is the limiting factor in a DoS attack in this scenario

# DoS Attacker Hardware



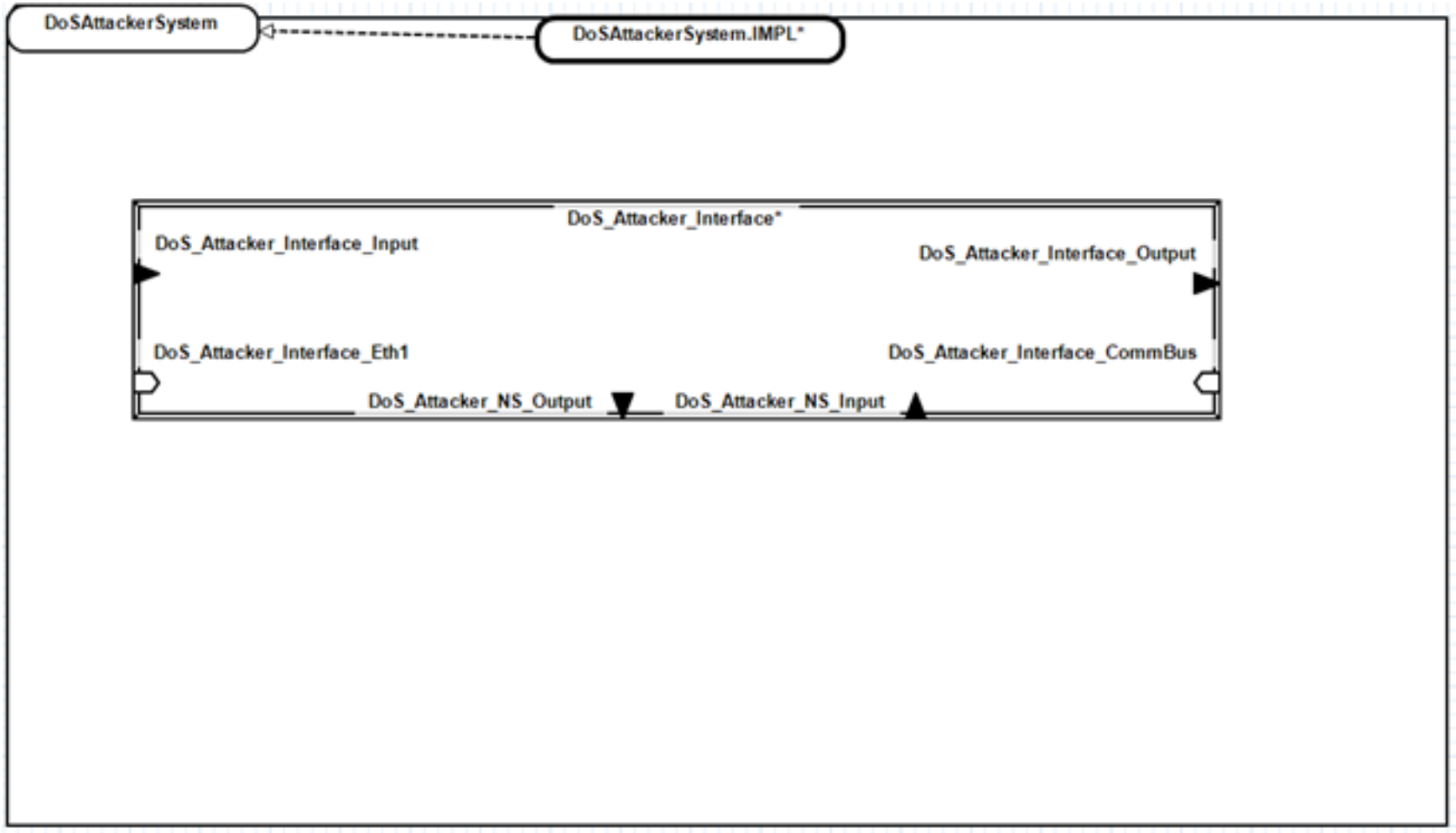

- Represents the execution platform components of the attacker node.
  - Processor
  - RAM Memory
  - ROM Memory
  - Internal Comm Bus
- Each component was given a bus access port to access the internal communication bus.
- A system implementation is created in the model allowing these components to be used in the DoS Attacker System Implementation model.

# DoS Attacker Software



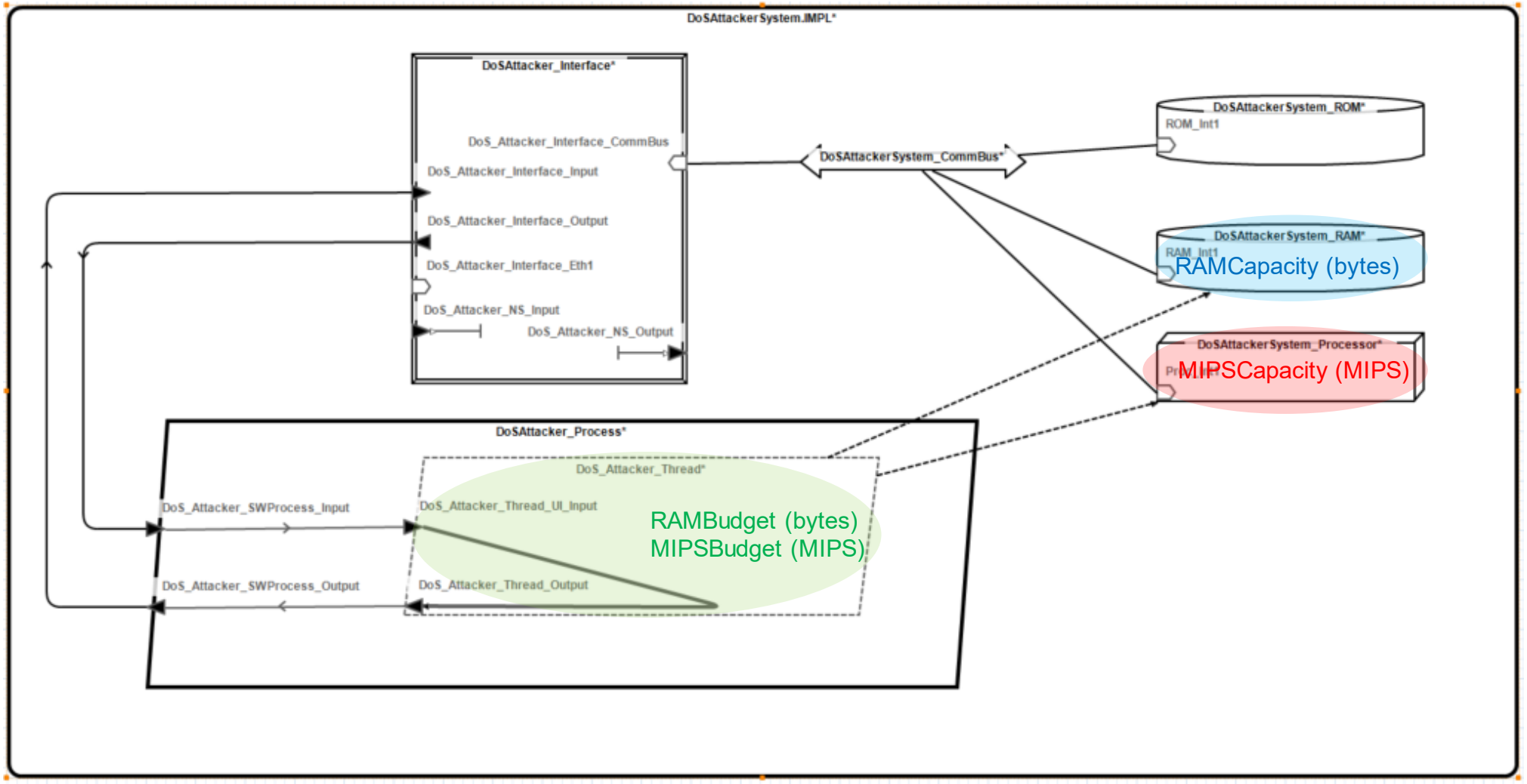

- Represents the process and thread used to execute a DoS attack.
- The thread receives input from the attack executor and creates the output used for the attack.
- A process implementation is created in the model allowing the thread and its higher level process to be used in the DoS Attacker System Implementation model.

# DoS Attacker System



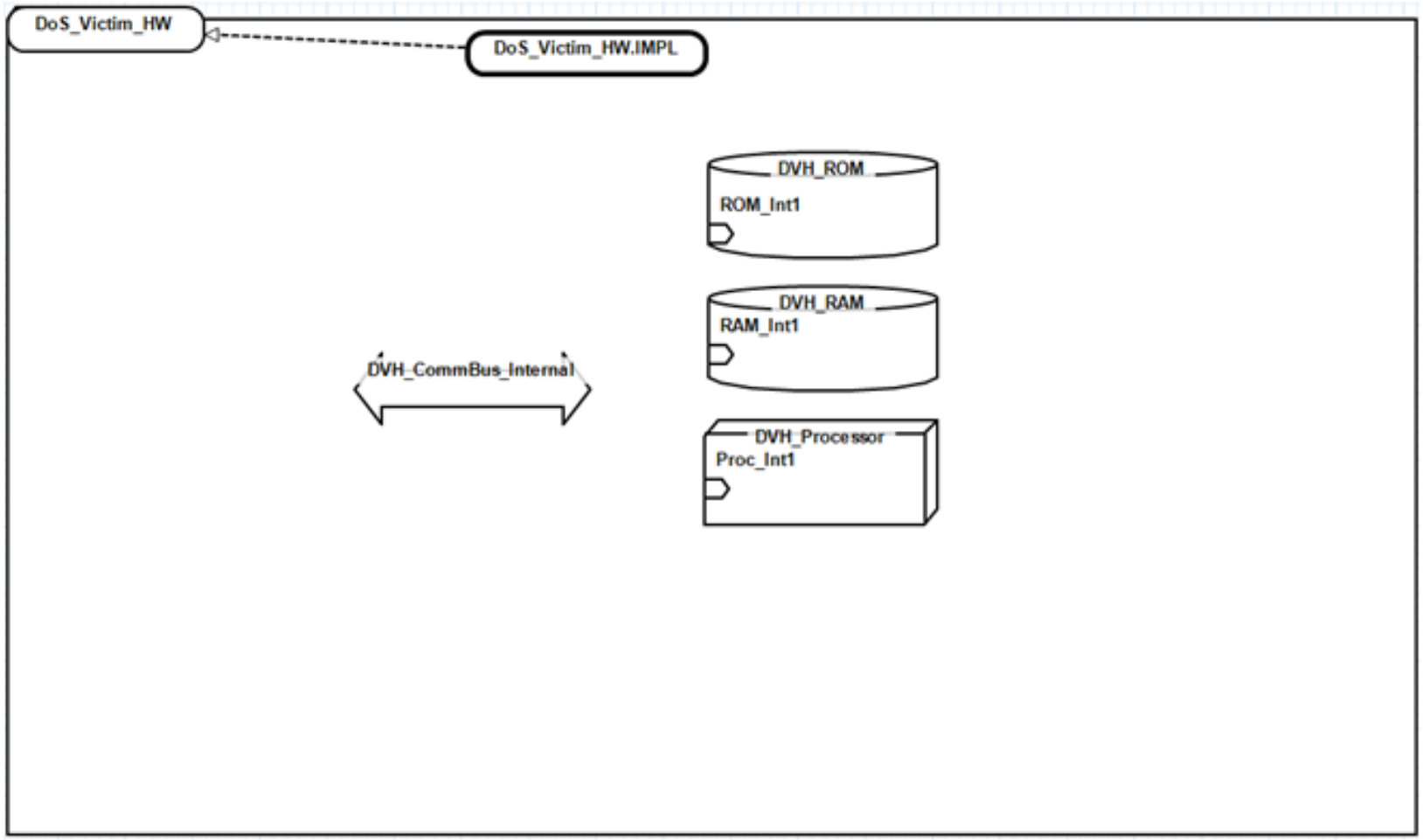

- Represents an AADL ‘device’ with the internal interfaces supporting the hardware and software models and the external interfaces required to model latency and bandwidth between the attacker and victim nodes.

# DoS Attacker System Implementation



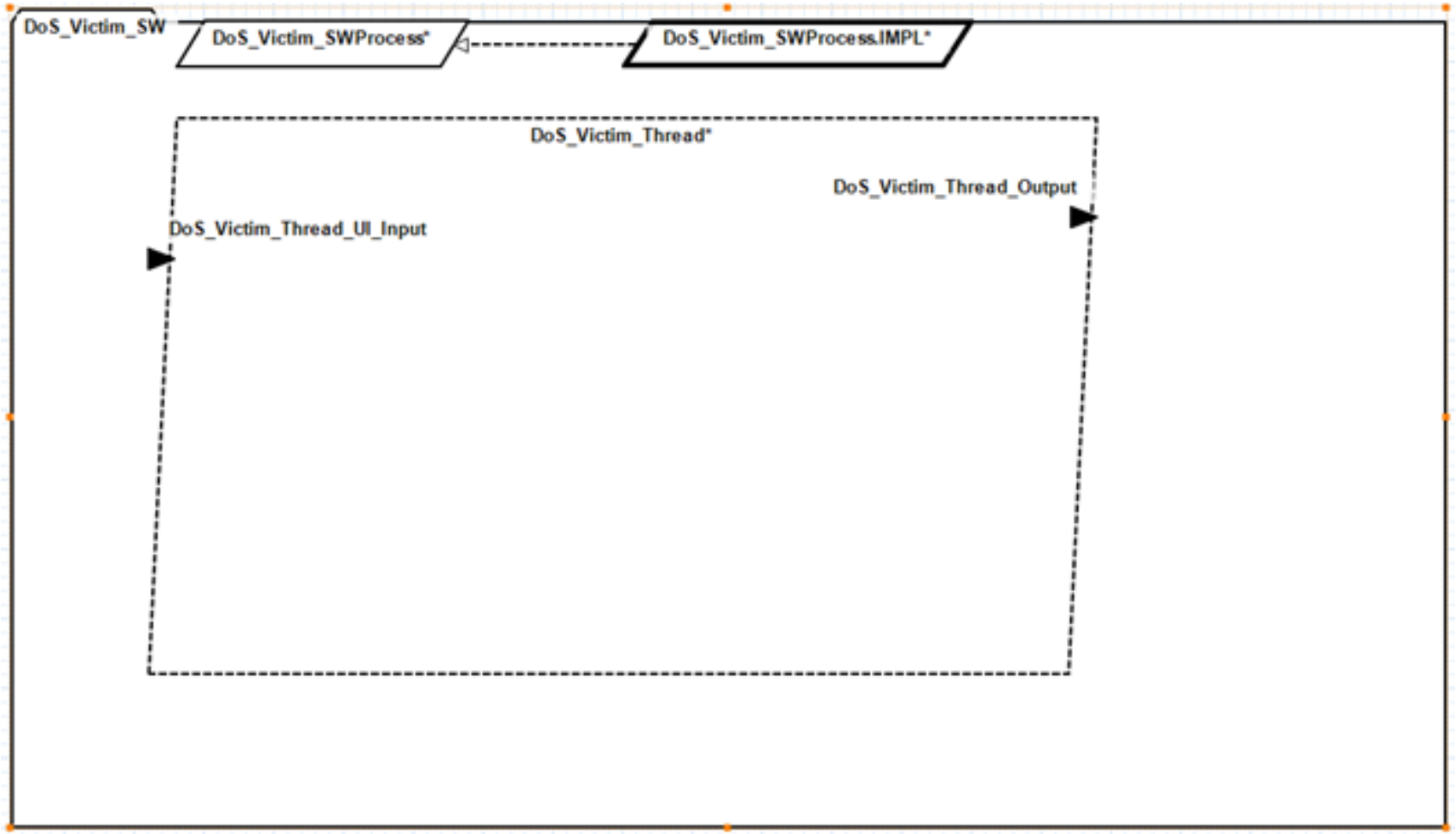

- Implements the other DoS attacker models into a comprehensive system model for the attacker node.
- It includes:
  - Execution platform components
  - The software
  - Devices input/output interfaces.

# DoS Victim Hardware



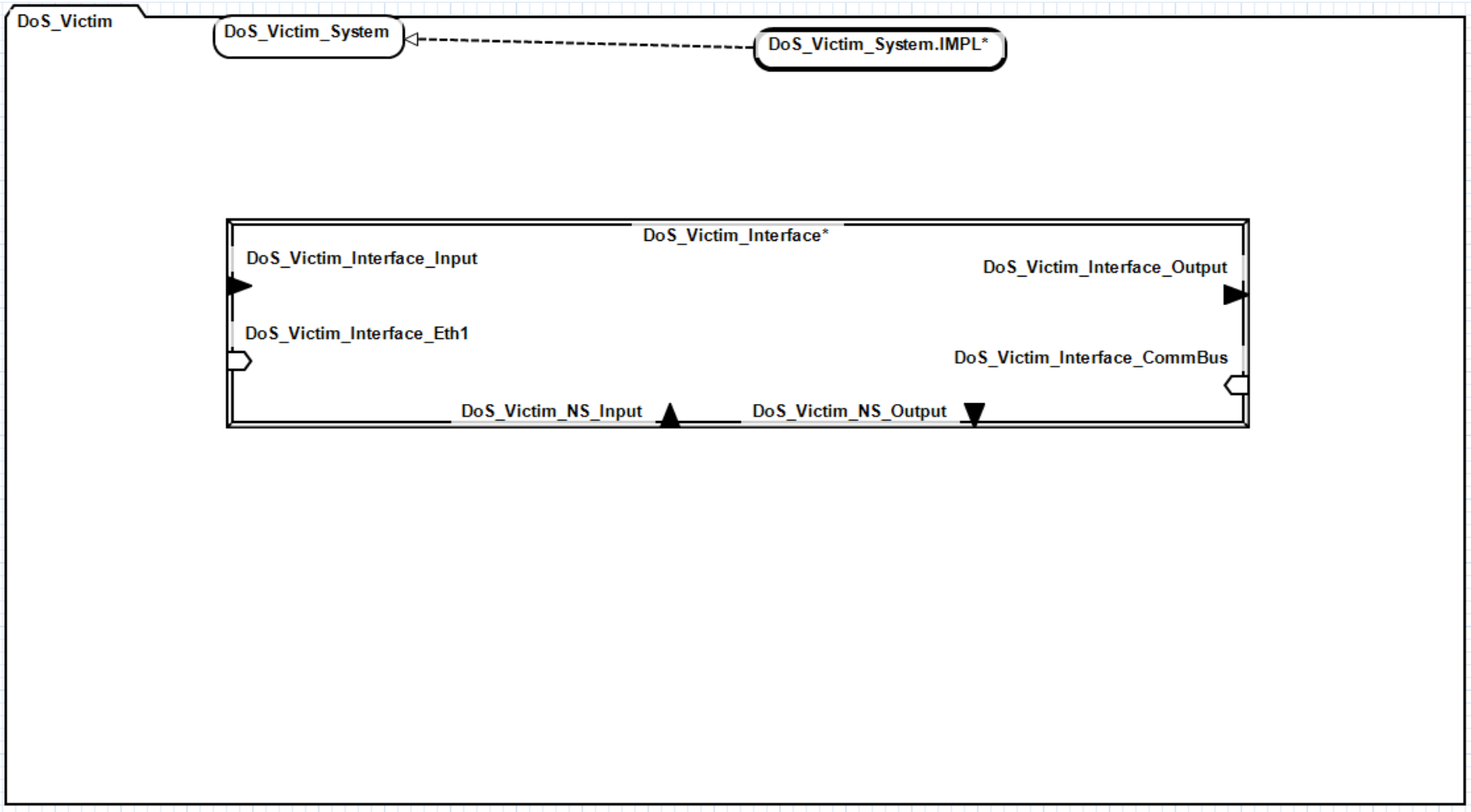

- Represents the execution platform components of the Victim node
  - Processor,
  - RAM Memory
  - ROM Memory
  - Internal Comm Bus
- Each component was given a bus access port to access the Internal Comm Bus
- A system implementation is created in the model allowing these components to be used in the DoS Victim System Implementation model.

# DoS Victim Software



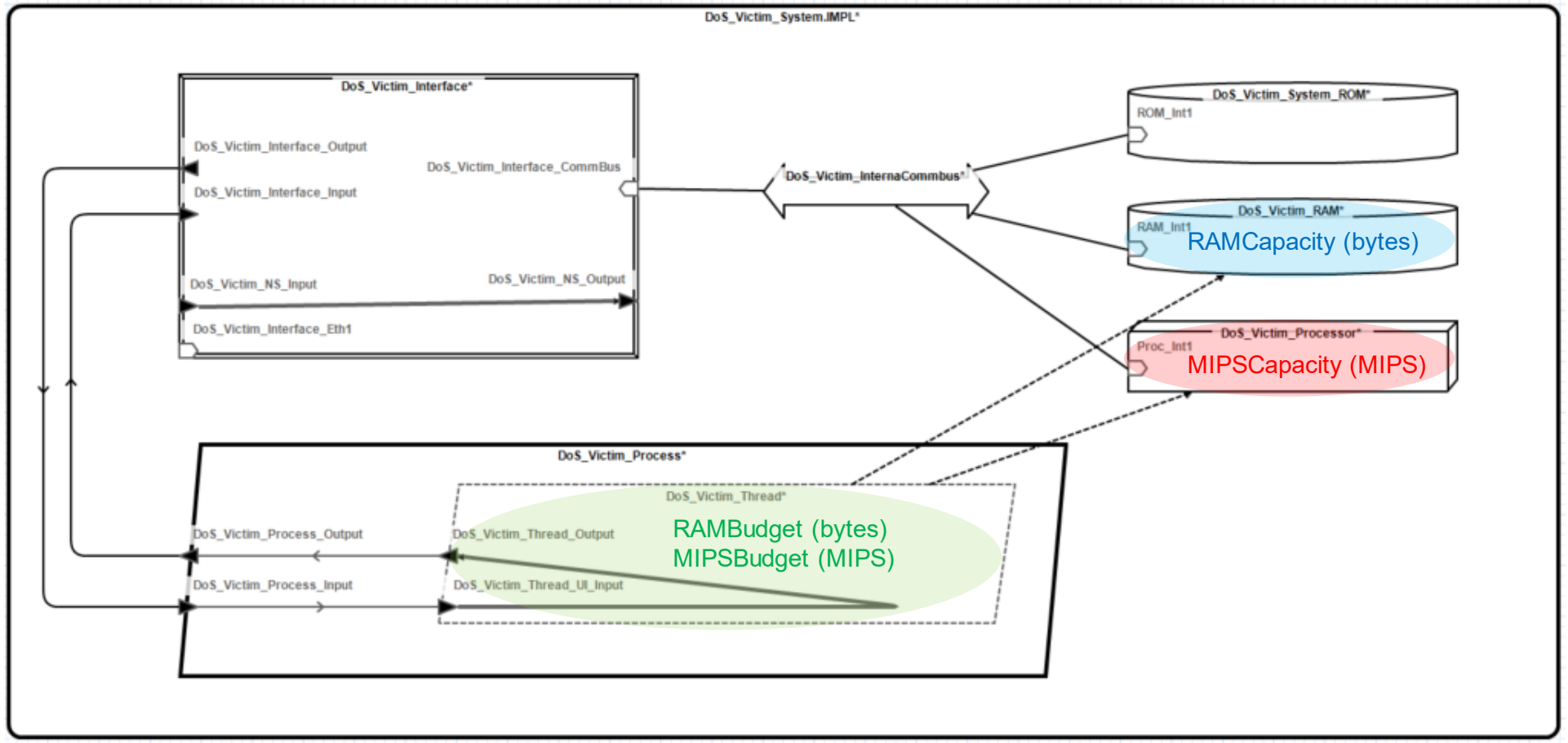

- Represents the process and thread targeted during a DoS attack.
- The thread receives input from the user and creates output in response to the attack.
- A process implementation is created in the model allowing the thread and its higher level process to be used in the DoS Victim System Implementation model.

# DoS Victim System



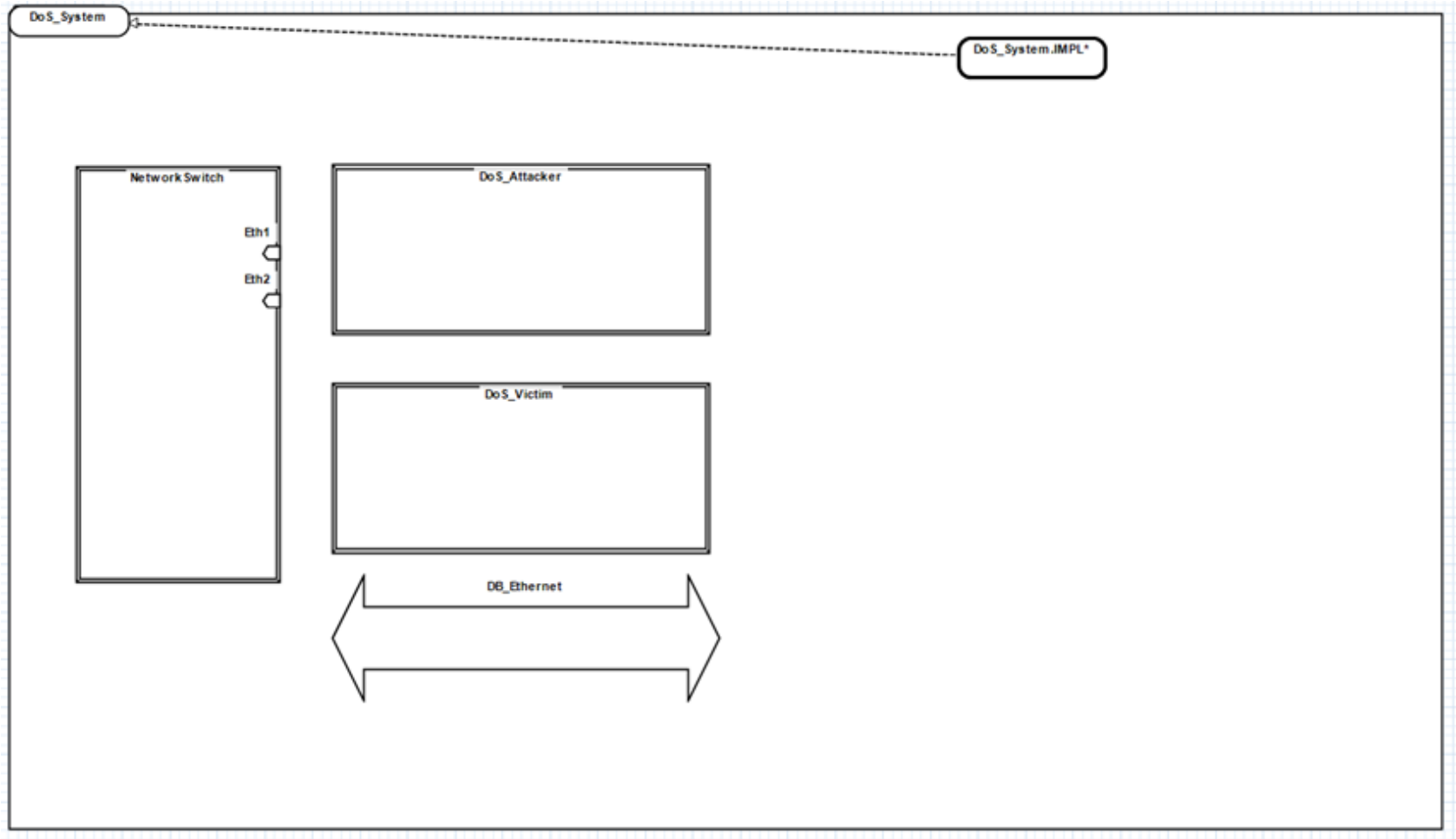

- AADL ‘device’ with the internal interfaces supporting the hardware and software models
- External interfaces required to model latency and bandwidth between the victim and attacker nodes.

# DoS Victim System Implementation



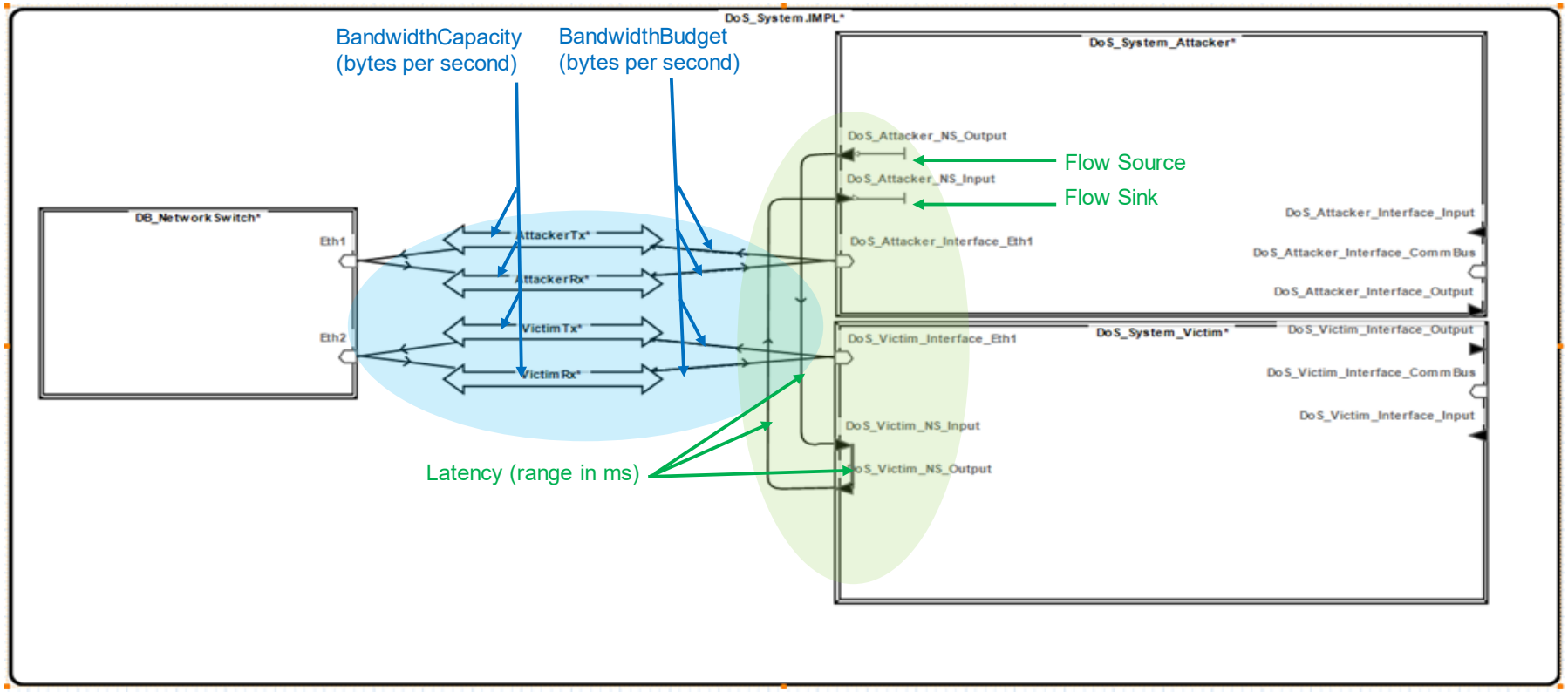

- Implements the other DoS victim models into a comprehensive system model for the victim node.
- Includes
  - Execution platform components
  - Software
  - Devices input/output interfaces.

# DoS System




- Defines the components used to create the attacker and victim nodes to perform a latency and bandwidth analysis.
- The DoS Attacker and DoS Victim devices defined in the DoS Attacker System and DoS Victim System are used here.
- An external bus and device representing an Ethernet network switch are also part of the components defined for use in the implementation diagram.

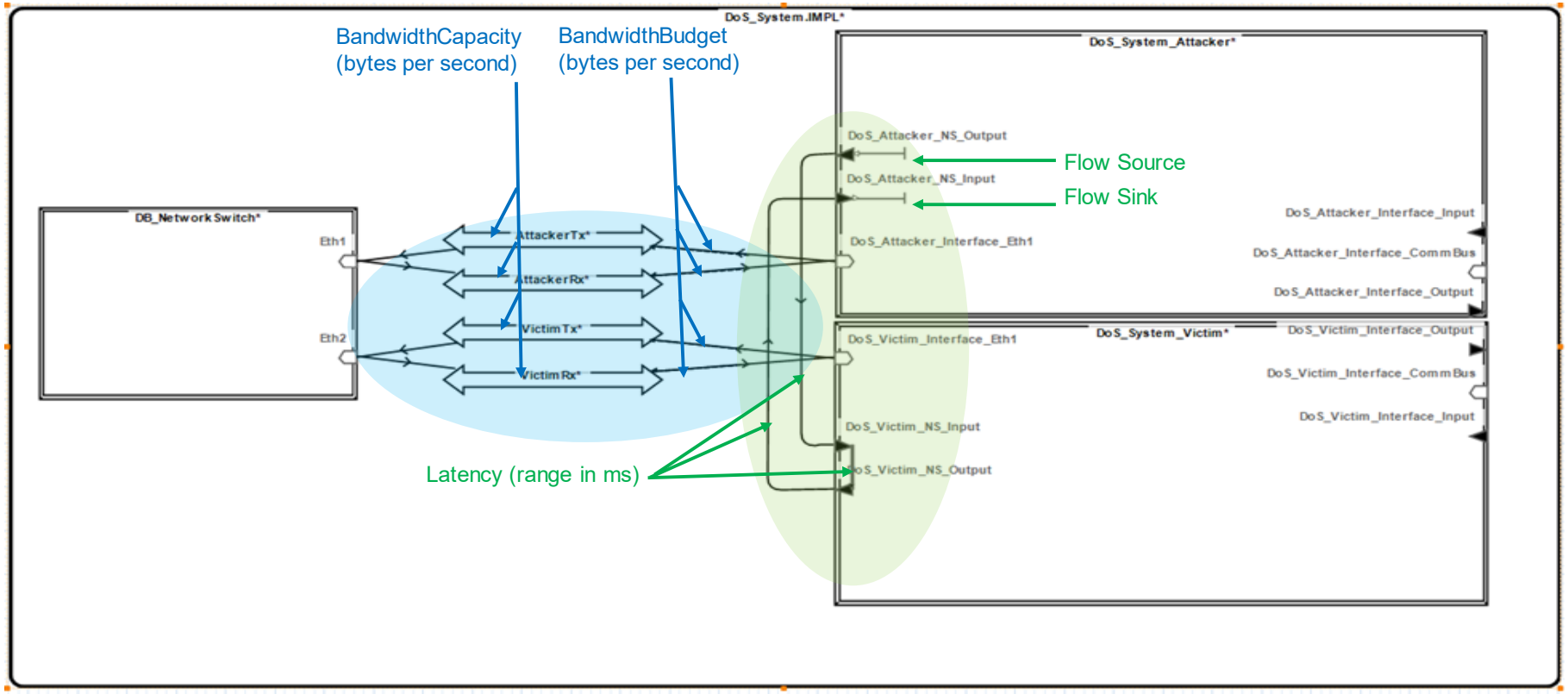
# DoS System Implementation (1 of 2)




- Contains the device representing the attacker and the device representing the victim.
- The devices were defined and used in other models.
- To accurately model switched Ethernet connections, two bus subcomponents were defined.
  - One bus subcomponent represents the transmit path from the node to the Ethernet switch.
  - One bus represent the receive path from the Ethernet switch to the node.
  - One connection binding was created to bind to the bus subcomponent.

This implementation model allows OSATE to analyze the bandwidth on the Ethernet links and the latency associated with the round trip time between the attacker node and victim node.

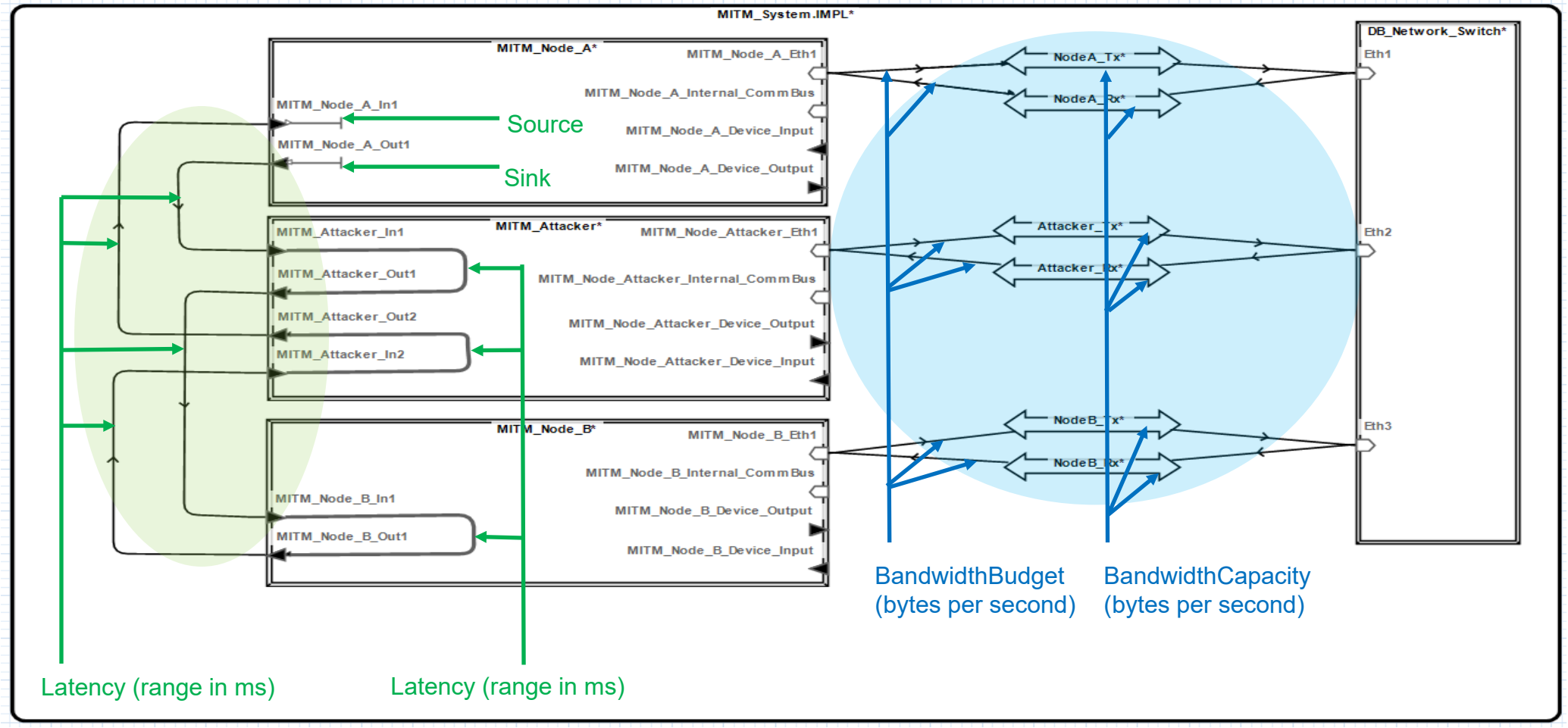
# DoS System Implementation (2 of 2)




Latency values were added to each leg of the path including the flow path defined in the victim node.

- The data port connections between the attacker node and victim node are used for the OSATE latency analysis.
- An end-to-end flow specification was defined allowing latency analysis. A flow source (start point) and a flow sink (end point) were defined on the output (start point) and input (end point) data ports of the attacker.
- A flow path was defined between the victim node's input and output ports.
- The end-to-end flow specification was defined from the attacker's output port to the victim's input port through the flow path, into the victim's output port, and finally back to the attacker's input port.

# MITM System



# DoS Attack Demo – Victim Node

- AADL Parameters (Victim Node)
  - RAM Capacity: 4 GBytes
  - RAM Budget: From Test Data
  - CPU Capacity: 1.44 GHz Intel Atom x5 Z8350
  - CPU Budget: From Test Data
  - Bandwidth Capacity: 1 Gbps
  - Bandwidth Budget: From Test Data
- AADL Parameters (System)
  - Latency Baseline: From Test Data
  - Latency Attack: From Test Data

# DoS Attack Demo - Execute Attack

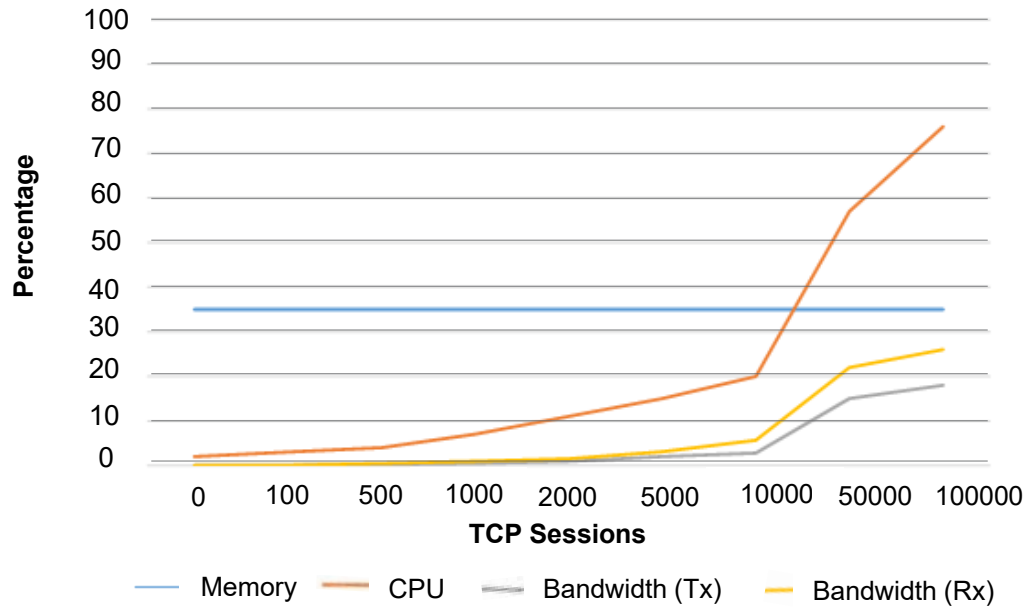
- NMAP Scan
- Start ‘Ping’ to measure latency
  - ping 192.168.1.20
- Start DoS SYN Flood
  - hping3 192.168.1.20 –c 1000 – faster –S –p 135

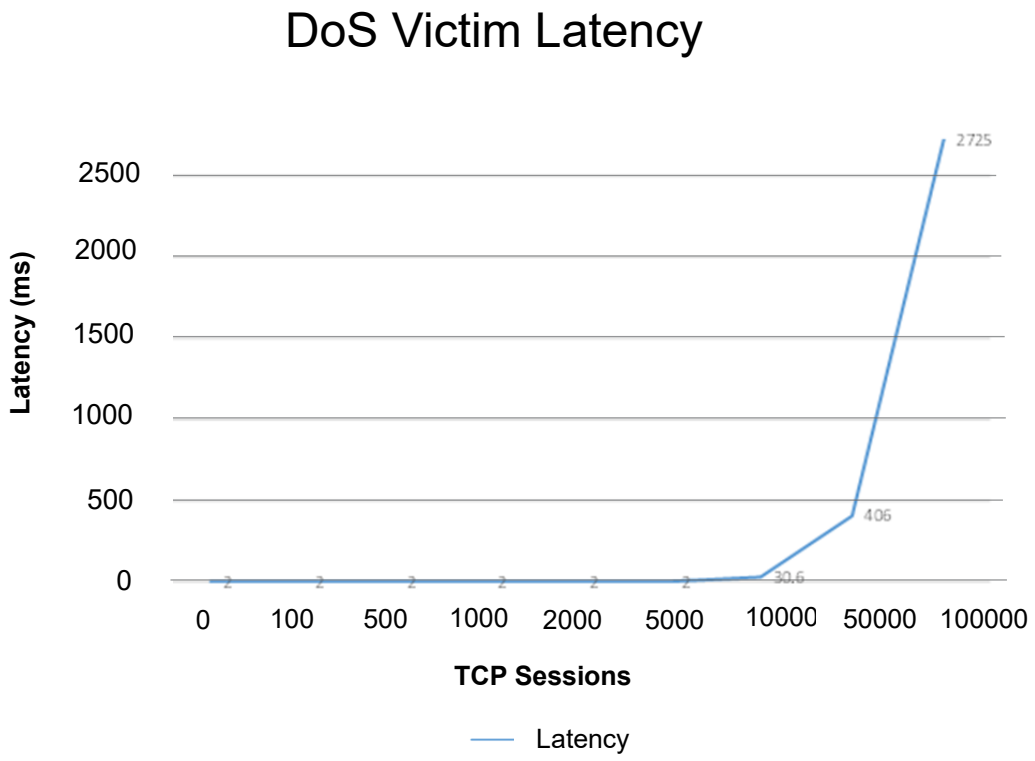
No. Packets SYN Flag Port 135 (Remote Procedure Call)

# DoS Attack Demo – Collected Victim Data

| Analysis: Victim Node<br>RAM: 4 Gbytes<br>CPU: Intel Atom x5 1.44 GHz Z8350<br>Bandwidth: 1 Gbits per second/125 Mbytes per second<br>Latency: 2.0 ms | RAM Baseline | RAM Measured | CPU Baseline | CPU Measured | Bandwidth Baseline | Bandwidth Measured (Rx/Tx) | Latency Baseline | Latency Measured |
|---|---|---|---|---|---|---|---|---|
| Number of Concurrent Half-Open Connections: 100 | 1.4 Gbytes | 1.4 Gbytes | 2.00% | 3.00% | 0 kbps | 96k kbps/40 kbps | 2.0 ms | 2.0 ms |
| Number of Concurrent Half-Open Connections: 500 | 1.3 Gbytes | 1.3 Gbytes | 2.00% | 4.00% | 0 kbps | 472 kbps/232 kbps | 2.0 ms | 2.0 ms |
| Number of Concurrent Half-Open Connections: 1000 | 1.4 Gbytes | 1.4 Gbytes | 3.00% | 7.00% | 0 kbps | 960 kbps/460 kbps | 2.0 ms | 2.0 ms |
| Number of Concurrent Half-Open Connections: 2000 | 1.4 Gbytes | 1.4 Gbytes | 2.00% | 11.00% | 0 kbps | 1.5 Mbps/928 kbps | 2.0 ms | 2.0 ms |
| Number of Concurrent Half-Open Connections: 5000 | 1.4 Gbytes | 1.4 Gbytes | 2.00% | 15.00% | 0 kbps | 3.1 Mbps/2.0 Mbps | 2.0 ms | 2.0 ms |
| Number of Concurrent Half-Open Connections: 10000 | 1.4 Gbytes | 1.4 Gbytes | 2.00% | 20.00% | 0 kbps | 5.7 Mbps/2.8 Mbps | 2.0 ms | 30.6 ms |
| Number of Concurrent Half-Open Connections: 50000 | 1.4 Gbytes | 1.4 Gbytes | 2.00% | 57.00% | 0 kbps | 22 Mbps/13 Mbps | 2.0 ms | 406.0 ms |
| Number of Concurrent Half-Open Connections: 100000 | 1.4 Gbytes | 1.4 Gbytes | 2.00% | 76.00% | 0 kbps | 26 Mbps/16 Mbps | 2.0 ms | 2,725.0 ms |

# DoS Attack Demo – Victim Resource Performance and Latency



# Case Study

Process Steps

1. Threat and Attack selection
2. Develop AADL models
3. Lab system and measurements
4. Perform calculations
5. Add values to the AADL models
6. Perform the OSATE analyses
7. Perform Cybersecurity Analysis

# Case Study Steps 1 - 4

- **Step 1 - Threat and Attack selection**
  - DoS Syn Flood Attack with 15,000 Simultaneous Sessions
- **Step 2 - Develop AADL models**
  - Use DoS ADDL Models
- **Step 3 - Lab System and Measurements**
  - Lab System (Kali Linux VM , Windows 7 VM, Oracle VirtualBox)
    - RAM = 2 Gbytes
    - CPU = 35174 MIPS
    - Bandwidth = 1 Gbps / 125 Mbytesps
  - Measurements at 15,000 sessions with 120 byte payload
    - RAM = Negligible change over baseline
    - CPU = 50% to 75% utilization
    - Bandwidth = 12%
- **Step 4 - Perform Calculations**
  - Memory Estimated
    - 15 Mbytes = 15,000 sessions * 1 kByte
  - Δ Memory Observed
    - Negligible Change
    - Note: Prediction of 15 Mbytes is less than 1% of RAM
  - Δ CPU Observed
    - 26,008.78 MIPS = 35147 MIPS * (75% – 1%)
  - Bandwidth Estimated
    - 44 Mbps = 2 ((184 bytes * 8 bits per byte) * 15,000 sessions)
  - Bandwidth Observed
    - 120 Mbps = 1 Gbps * 12%
  - Latency
    - Latency Delta (ms) = Measured Latency (ms) – Baseline Latency (ms)

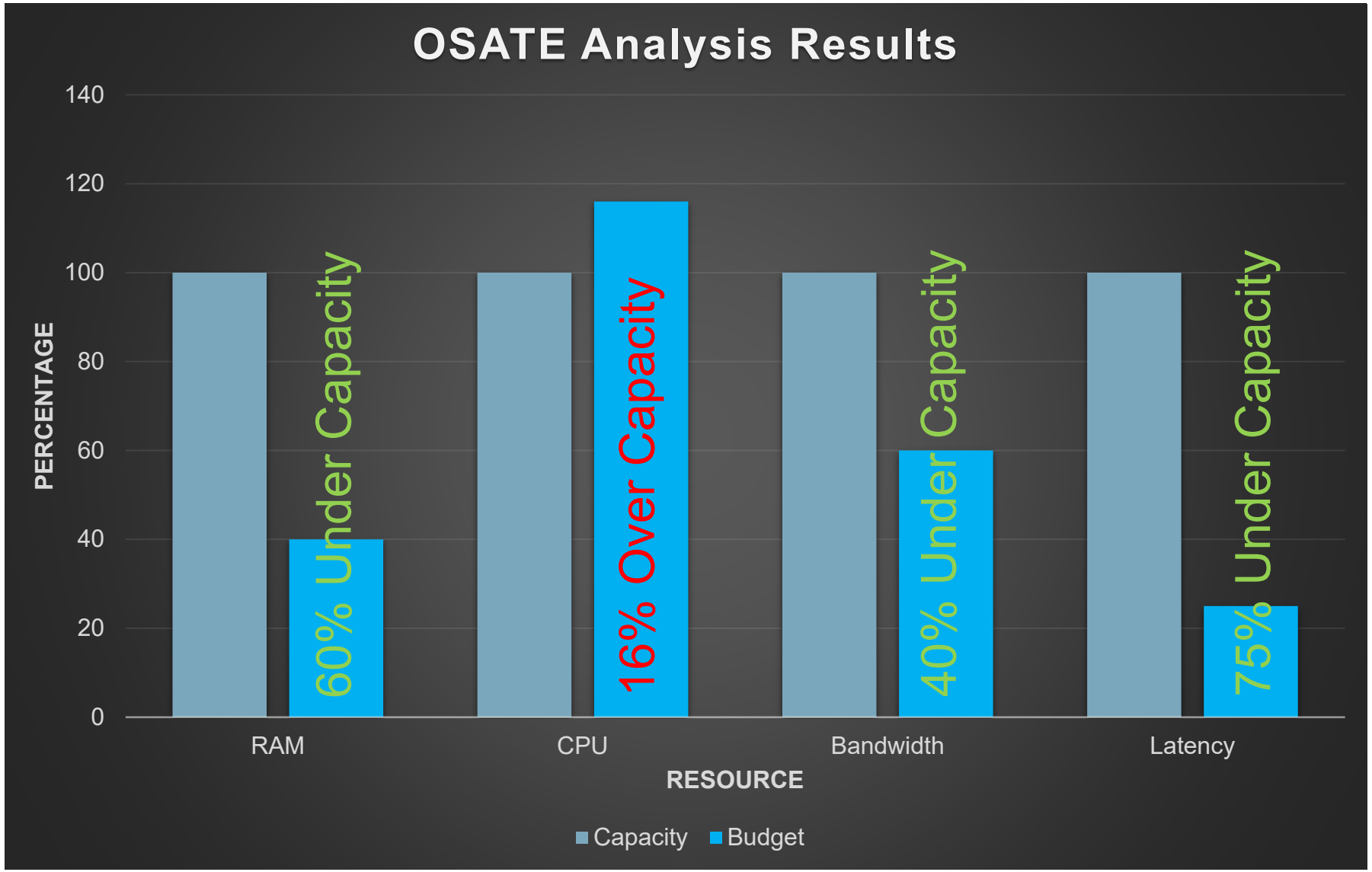
# Case Study Step 5: Add Values to AADL Models

- Add ‘capacity’ values to the AADL models
  - RAMCapacity = 2 Gbytes
  - MIPSCapacity = 35174 MIPS
  - BandwidthCapacity = 125 Mbytesps
  - Latency (end-to-end) = 20ms
- Calculate ‘Attack’ Budget
  - Based on 15,000 Session DoS Attack
  - RAMBudget = 0 Mbytes
  - MIPSBudget = 26,009 MIPS
  - BandwidthBudget = 15 Mbytesps
  - Latency = 2ms

- Add ‘budget’ values to the AADL models
- RAM
  - RAMBudget Thread 1 = 300 Mbytes
  - RAMBudget Thread 2 = 500 Mbytes
  - RAMBudget Attack = 0 bytes
- CPU
  - MIPSBudget Thread 1 = 8000 MIPS
  - MIPSBudget Thread 2 = 7000 MIPS
  - MIPSBudget Attack = 26,009 MIPS
- Bandwidth
  - BandwidthBudget at peak = 60 Mbytesps
  - BandwidthBudget Attack = 15 Mbytesps
- Latency
  - Latency Baseline = 3ms
  - Latency Attack = 2ms

# Case Study Step 6: OSATE Analyses

- RAMBudget = 800 Mbytes
- MIPSBudget = 41,009 MIPS
- BandwidthBudget = 75 Mbytesps
- Latency = 5ms




Raytheon Intelligence & Space

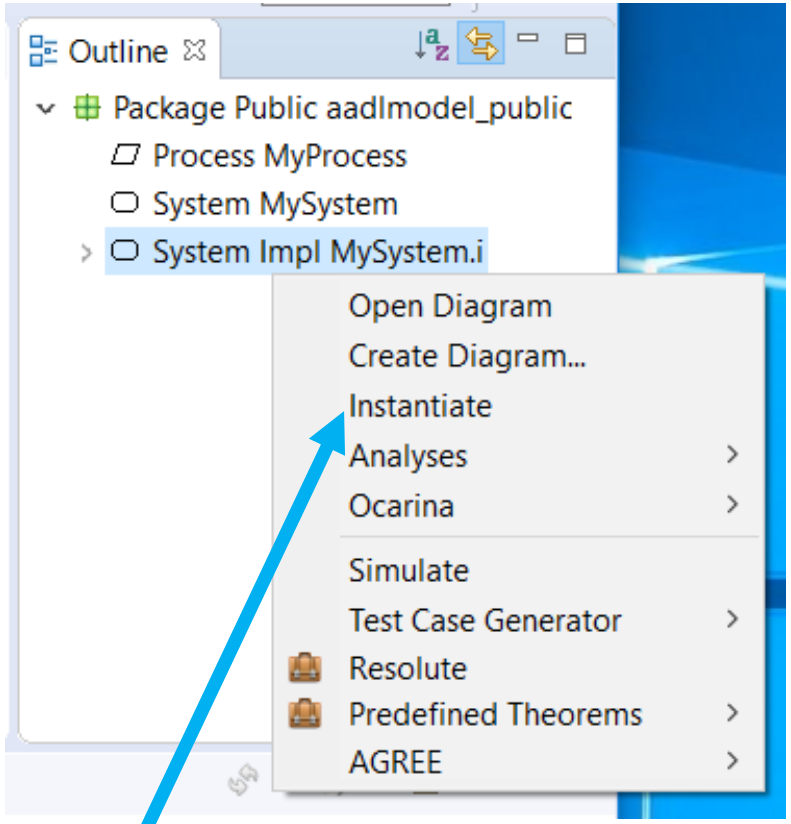
# Case Study Step 7: Cybersecurity Analysis

- Attack causes CPU Margin to be exceeded by 16% (5835 MIPS)
- Other resources are under capacity
- Determine Defense-In-Depth Strategy
  - Determine CPU capacity
  - Determine number of maximum open sessions
  - Adjust Syn-Ack timeout value
  - Additional Options
    - **Micro blocks**—administrators can allocate a micro-record (as few as 16 bytes) in the server memory for each incoming SYN request instead of a complete connection object.
    - **SYN cookies**—using cryptographic hashing, the server sends its SYN-ACK response with a sequence number (seqno) that is constructed from the client IP address, port number, and possibly other unique identifying information. When the client responds, this hash is included in the ACK packet. The server verifies the ACK, and only then allocates memory for the connection.
    - **RST cookies**—for the first request from a given client, the server intentionally sends an invalid SYN-ACK. This should result in the client generating an RST packet, which tells the server something is wrong. If this is received, the server knows the request is legitimate, logs the client, and accepts subsequent incoming connections from it.
    - **Stack tweaking**—administrators can tweak TCP stacks to mitigate the effect of SYN floods. This can either involve reducing the timeout until a stack frees memory allocated to a connection, or selectively dropping incoming connections.

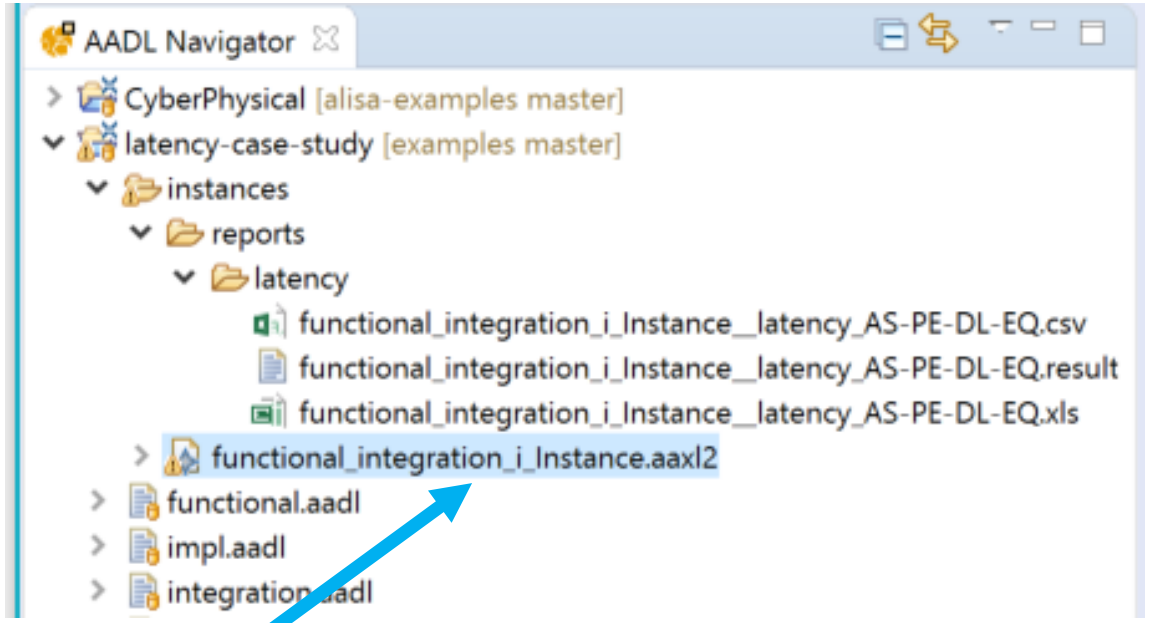
# Questions

# Backup

# OSATE Analysis – Model Instantiation



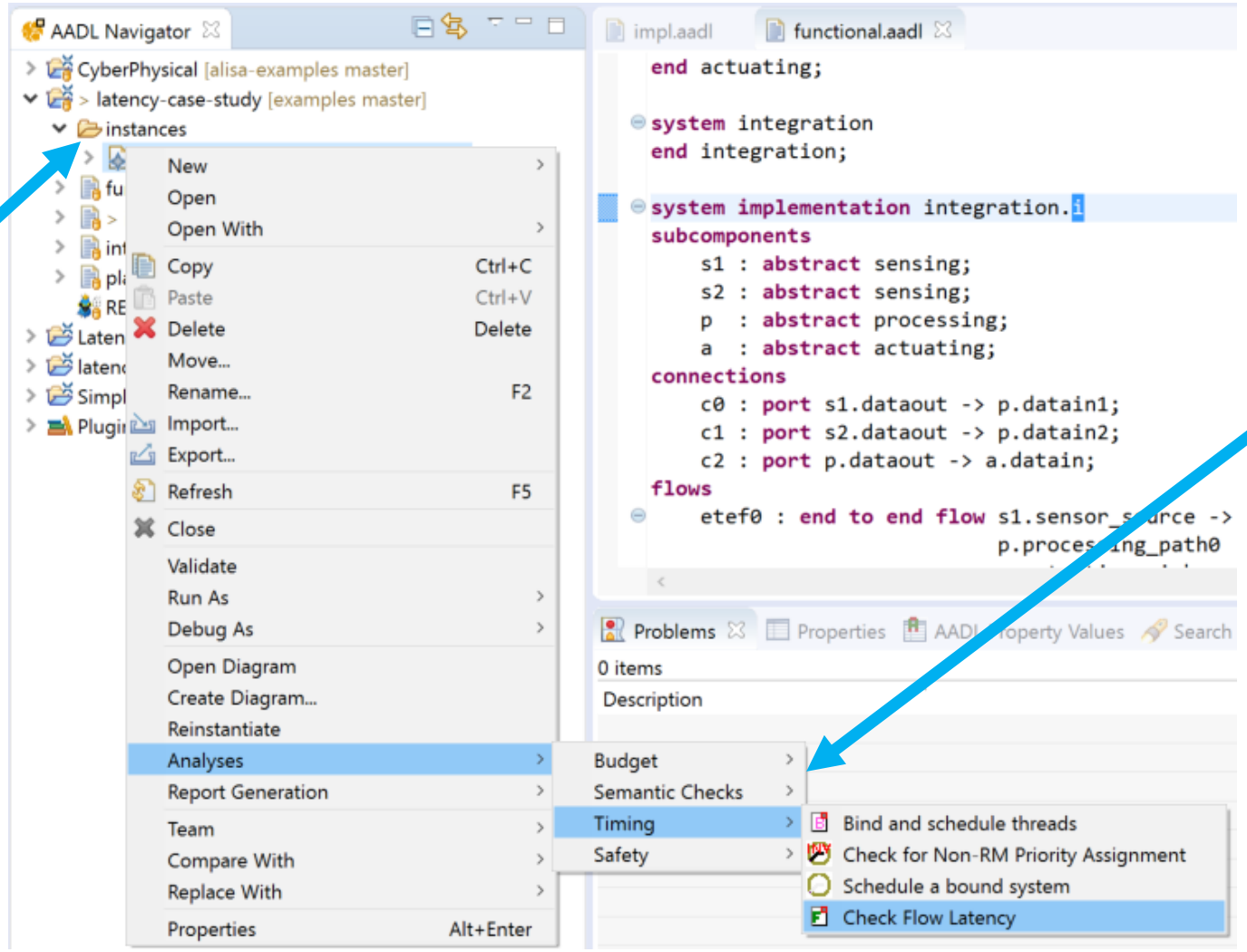

Instantiate Implementation Model




Model Instance Created

Raytheon Intelligence & Space

# OSATE Analysis – Select Analysis




Select Model Instance

Select Analysis

# OSATE Analysis – Bound Resource Budget

Processor Summary Report:

Processor DoSAttackerSystem_Processor: Total MIPS 500.000 MIPS of bound tasks within MIPS capacity 35.174 GIPS of DoSAttackerSystem_Processor

Memory Summary Report:

Total RAM 1000000.0 KB of bound tasks within Memory capacity 2000000.0 KB of DoSAttackerSystem_RAM

No application components bound to DoSAttackerSystem_ROM with ROM capacity 100.000 GByte

Detailed Workload Report: for Processor DoSAttackerSystem_Processor with Capacity 35.174 GIPS

| Component | Budget | Actual |
|---|---|---|
| thread DoSAttacker_Process.DoS_Attacker_Thread | 500.000 MIPS | 0.000 MIPS |
| Total | | 500.000 MIPS |

SUMMATIONS

Detailed Workload Report: for memory DoSAttackerSystem_RAM with Capacity 2.000 GByte

Assume Report Error (KByte not KBytesps)

| Component | Budget | Actual | |
|---|---|---|---|
| DoSAttacker_Process.DoS_Attacker_Thread | 1000000.0 KBytesps | 1000000.0 KBytesps | No actual. Added budget to total. |
| Total | | 1000000.000 KByte | |

```
memory DAH_ROM
    features
        ROM_Int1: requires bus access DAH_CommBus_Internal;
    properties
        SEI::ROMCapacity => 100.0 GByte;
end DAH_ROM;

memory DAH_RAM
    features
        RAM_Int1: requires bus access DAH_CommBus_Internal;
    properties
        SEI::RAMCapacity => 2.0 GByte;
end DAH_RAM;

processor DAH_Processor
    features
        Proc_Int1: requires bus access DAH_CommBus_Internal;
    properties
        SEI::MIPSCapacity => 35.174 Gips;
```

```
thread DoS_Attacker_Thread
    features
        DoS_Attacker_Thread_UI_Input: in data port;
        DoS_Attacker_Thread_Output: out data port;
    flows
        DoS_Attacker_Thread_new_flow_spec: flow path DoS_Att
    properties
        Dispatch_Protocol => Periodic;
        Period => 5Ms;
        SEI::MIPSBudget => 500.0 Mips;
        SEI::RAMBudget => 1000.0 MByte;
        SEI::ROMBudget => 1500.0 MByte;
end DoS_Attacker_Thread;
```

# OSATE Bus Load Analysis

| | | | |
|---|---|---|---|
| Connection Budget Details for bus AttackerTx with capacity 125000.0 KBytesps | | | |
| Connection | Budget | | |
| DoS_System_Attacker.DoS_Attacker_Interface_Eth1 -> AttackerTx | 100.0 KBytesps | | |
| Total | | | |
| Connection Budget Details for bus AttackerRx with capacity 125000.0 KBytesps | | | |
| Connection | Budget | Actual (Data Size * Sender Rate) | Note |
| AttackerRx -> DoS_System_Attacker.DoS_Attacker_Interface_Eth1 | 50.0 KBytesps | 0.0 KBytesps | Using budget bandwidth |
| Total | | 50.000 KBytesps | |
| Connection Budget Details for bus VictimTx with capacity 125000.0 KBytesps | | | |
| Connection | Budget | Actual (Data Size * Sender Rate) | Note |
| DoS_System_Victim.DoS_Victim_Interface_Eth1 -> VictimTx | 50.0 KBytesps | 0.0 KBytesps | Using budget bandwidth |
| Total | | 50.000 KBytesps | |
| Connection Budget Details for bus VictimRx with capacity 125000.0 KBytesps | | | |
| Connection | Budget | Actual (Data Size * Sender Rate) | Note |
| VictimRx -> DoS_System_Victim.DoS_Victim_Interface_Eth1 | 100.0 KBytesps | 0.0 KBytesps | Using budget bandwidth |
| Total | | 100.000 KBytesps | |

```
DoS_System_IMPL_new_connection11: bus access VictimRx -> DoS_System_Victim.DoS_Victim_Interface_Eth1 {
        SEI::BandWidthBudget => 100.0 kbytesps;
    };
DoS_System_IMPL_new_connection14: bus access DoS_System_Victim.DoS_Victim_Interface_Eth1 -> VictimTx {
        SEI::BandWidthBudget => 50.0 kbytesps;
    };
DoS_System_IMPL_new_connection15: bus access AttackerRx -> DoS_System_Attacker.DoS_Attacker_Interface_Eth1 {
        SEI::BandWidthBudget => 50.0 kbytesps;
    };
DoS_System_IMPL_new_connection17: bus access DoS_System_Attacker.DoS_Attacker_Interface_Eth1 -> AttackerTx {
        SEI::BandWidthBudget => 100.0 kbytesps;
    };
```

```
bus DB_Ethernet
    properties
        SEI::BandWidthCapacity => 125.0 MBytesps;
end DB_Ethernet;
```

# OSATE Flow Latency Analysis

```
connections
    DoS_System_IMPL_new_connection5: port DoS_System_Attacker.DoS
                Latency => 5ms .. 5ms;};
    DoS_System_IMPL_new_connection6: port DoS_System_Victim.DoS_V
                Latency => 4ms .. 4ms;};
```

```
flows
    DoS_Victim_Interface_new_flow_spec3: flow path DoS_Vi
                Latency => 6ms .. 6ms;
            };
```

Latency analysis with preference settings: asynchronous system/major partition frame/worst case as deadline/best case as empty queue

Latency results for end-to-end flow 'Attacker_to_Victim_Roundtrip' of system 'DoS_System.IMPL'

| Result | Min Specified | Min Actual | Min Method | Max Specified | Max Actual | Max Method |
|---|---|---|---|---|---|---|
| device DoS_System_Attacker | | 0.0ms | no latency | | 0.0ms | no latency |
| connection DoS_System_Attacker.DoS_Attacker_NS_Output -> DoS_System_Victim.DoS_Victim_NS_Input | 5.0ms | 5.0ms | specified | 5.0ms | 5.0ms | specified |
| device DoS_System_Victim | 6.0ms | 6.0ms | specified | 6.0ms | 6.0ms | specified |
| connection DoS_System_Victim.DoS_Victim_NS_Output -> DoS_System_Attacker.DoS_Attacker_NS_Input | 4.0ms | 4.0ms | specified | 4.0ms | 4.0ms | specified |
| device DoS_System_Attacker | | 0.0ms | no latency | | 0.0ms | no latency |
| Latency Total | 6.0ms | 15.0ms | | 6.0ms | 15.0ms | |
| Specified End To End Latency | | 0.0ms | | | 0.0ms | |
| End to end Latency Summary | | | | | | |
| WARNING | | | | | | |
| Expected end to end latency is not specified | | | | | | |

SUMMATIONS Min/Max

# MITM Attack Demo

- Round Trip Latency
  - Establish Baseline Latency
    - Ping 192.168.1.20 from 192.168.1.30
    - Ping 192.168.1.30 from 192.168.1.20
  - Execute MITM Attack with Ettercap
  - Test MITM Latency
    - Ping 192.168.1.20 from 192.168.1.30
    - Ping 192.168.1.30 from 192.168.1.20
- Eavesdropping
  - Execute MITM Attack with Ettercap
  - Start Wireshark on MITM Node
  - Start Python Server Script
  - Start Python Client Script
  - Show data eavesdropping