EXata 5.1
Cyber Model Library

August 2013

SCALABLE Network Technologies, Inc.

600 Corporate Pointe, Suite 1200
Culver City, CA 90230

+1.310.338.3318 TEL
+1.310.338.7213 FAX

SCALABLE-NETWORKS.COM

SCALABLE Network Technologies, Inc.

600 Corporate Pointe, Suit 1200

Culver City, CA 90230

+1.310.338.3318 TEL
+1.310.338.7213 FAX

SCALABLE-NETWORKS.COM

Table of Contents

1 Overview of Model Library

1.1 List of Models in the Library

The models described in the Cyber Model Library are listed in Table 1-1.

TABLE 1-1. Cyber Library Models

Model Name	Model Type	Section Number
Adversary Model	Multi-layer	Section 5.1
ANODR Model	Routing Protocol	Section 4.1
Certificate Model	Network Layer	Section 3.1
CPU and Memory Resource Model	OS Resource Model	Section 7.1
Denial of Service (DoS) Attack Model	Attack Model	Section 6.2
Firewall Model	Network Layer	Section 3.2
Information Assurance Hierarchical Encryption Protocol (IAHEP) Model	Network Layer	Section 3.3
Internet Protocol Security (IPSec) Model	Network Layer	Section 3.4
Internet Security Association and Key Management Protocol with Internet Key Exchange (ISAKMP-IKE) Model	Network Layer	Section 3.5
Public Key Infrastructure (PKI) Model	Network Layer	Section 3.6
Secure Neighbor Model	Network Layer	Section 3.7
Signal Intelligence (SIGINT) Model	Attack Model	Section 6.3
Virus Attack Model	Attack Model	Section 6.4
WEP and CCMP Model	MAC Layer	Section 2.1
Wireless Eavesdropping Attack Model	Attack Model	Section 6.5
Wireless Jamming Attack Model	Attack Model	Section 6.6

1.2 Conventions Used

1.2.1 Format for Command Line Configuration

This section describes the general format for specifying parameters in input files, the precedence rules for parameters, and the conventions used in the description of command line configuration for each model.

1.2.1.1 General Format of Parameter Declaration

The general format for specifying a parameter in an input file is:

```
[<Qualifier>] <Parameter Name> [<Index>] <Parameter Value>
```

where

`<Qualifier>` The qualifier is optional and defines the scope of the parameter declaration. The scope can be one of the following: Global, Node, Subnet, and Interface. Multiple instances of a parameter with different qualifiers can be included in an input file. Precedence rules (see Section 1.2.1.2) determine the parameter value for a node or interface.

Global: The parameter declaration is applicable to the entire scenario (to all nodes and interfaces), subject to precedence rules. The scope of a parameter declaration is global if the qualifier is not included in the declaration.

Example:

```
MAC-PROTOCOL            MACDOT11
```

Node: The parameter declaration is applicable to specified nodes, subject to precedence rules. The qualifier for a node-level declaration is a list of space-separated node IDs or a range of node IDs (specified by using the keyword `thru`) enclosed in square brackets.

Example:

```
[5 thru 10] MAC-PROTOCOL            MACDOT11
```

Subnet: The parameter declaration is applicable to all interfaces in specified subnets, subject to precedence rules. The qualifier for a subnet-level declaration is a space-separated list of subnet addresses enclosed in square brackets. A subnet address can be specified in the IP dot notation or in the EXata N syntax.

Example:

```
[N8-1.0 N2-1.0] MAC-PROTOCOL       MACDOT11
```

Interface: The parameter declaration is applicable to specified interfaces. The qualifier for an interface-level declaration is a space-separated list of subnet addresses enclosed in square brackets.

Example:

```
[192.168.2.1 192.168.2.4] MAC-PROTOCOL MACDOT11
```

`<Parameter Name>`	Name of the parameter.
`<Index>`	Instance of the parameter to which this parameter declaration is applicable, enclosed in square brackets. This should be in the range 0 to *n* -1, where *n* is the number of instances of the parameter. The instance specification is optional in a parameter declaration. If an instance is not included, then the parameter declaration is applicable to all instances of the parameter, unless otherwise specified.
`<Parameter Value>`	Value of the parameter.

Note: There should not be any spaces between the parameter name and the index.

Examples of parameter declarations in input files are:

```
PHY-MODEL                                         PHY802.11b
[1] PHY-MODEL                                     PHY802.11a
[N8-1.0] PHY-RX-MODEL                             BER-BASED
[8 thru 10] ROUTING-PROTOCOL                      RIP
[192.168.2.1 192.168.2.4] MAC-PROTOCOL            GENERICMAC
NODE-POSITION-FILE                                ./default.nodes
PROPAGATION-CHANNEL-FREQUENCY[0]                  2.4e9
[1 2] QUEUE-WEIGHT[1]                             0.3
```

Note: In the rest of this document, we will not use the qualifier or the index in a parameter's description. Users should use a qualifier and/or index to restrict the scope of a parameter, as appropriate.

1.2.1.2 Precedence Rules

Parameters without Instances

If the parameter declarations do not include instances, then the following rules of precedence apply when determining the parameter values for specific nodes and interfaces:

Interface > Subnet > Node > Global

This can be interpreted as follows:

- The value specified for an interface takes precedence over the value specified for a subnet, if any.
- The value specified for a subnet takes precedence over the value specified for a node, if any.
- The value specified for a node takes precedence over the value specified for the scenario (global value), if any.

Parameters with Instances

If the parameter declarations are a combination of declarations with and without instances, then the following precedence rules apply (unless otherwise stated):

Interface[i] > Subnet[i] > Node[i] > Global[i] > Interface > Subnet > Node > Global

This can be interpreted as follows:

- Values specified for a specific instance (at the interface, subnet, node, or global level) take precedence over values specified without the instance.

- For values specified for the same instance at different levels, the following precedence rules apply:
  - The value specified for an interface takes precedence over the value specified for a subnet, if any, if both declarations are for the same instance.
  - The value specified for a subnet takes precedence over the value specified for a node, if any, if both declarations are for the same instance.
  - The value specified for a node takes precedence over the value specified for the scenario (global value), if any, if both declarations are for the same instance.

1.2.1.3 Parameter Description Format

In the Model Library, most parameters are described using a tabular format described below. The parameter description tables have three columns labeled “Parameter”, “Values”, and “Description”. Table 1-2 shows the format of parameter tables. Table 1-4 shows examples of parameter descriptions in this format.

TABLE 1-2. Parameter Table Format

Parameter	Values	Description
<Parameter Name> <Designation> <Scope> [<Instances>]	<Type> [<Range>] [<Default Value>] [<Unit>]	<Description>

Parameter Column

The first column contains the following entries:

- ***<Parameter Name>*:** The first entry is the parameter name (this is the exact name of the parameter to be used in the input files).

- ***<Designation>:*** This entry can be *Optional* or *Required*. These terms are explained below.
  - ***Optional***: This indicates that the parameter is optional and may be omitted from the configuration file. (If applicable, the default value for this parameter is included in the second column.)
  - ***Required***: This indicates that the parameter is mandatory and must be included in the configuration file.

- ***<Scope>:*** This entry specifies the possible scope of the parameter, i.e., if the parameter can be specified at the global, node, subnet, or interface levels. Any combination of these levels is possible.If the parameter can be specified at all four levels, the keyword “All” is used to indicate that.

  Examples of scope specification are:

  Scope: All

  Scope: Subnet, Interface

  Scope: Global, Node

- ***<Instances>:*** If the parameter can have multiple instances, this entry indicates the type of index. If the parameter can not have multiple instances, then this entry is omitted.

Examples of instance specification are:

Instances: channel number

Instances: interface index

Instances: queue index

Values Column

The second column contains the following information:

- ***<Type>*:** The first entry is the parameter type and can be one of the following: Integer, Real, String, Time, Filename, IP Address, Coordinates, Node-list, or List. If the type is a List, then all possible values in the list are enumerated below the word “List”. (In some cases, the values are listed in a separate table and a reference to that table is included in place of the enumeration.)

  Table 1-3 shows the values a parameter can take for each type.

TABLE 1-3. Parameter Types

Type	Description
Integer	Integer value Examples: `2`, `10`
Real	Real value Examples: `15.0`, `-23.5`, `2.0e9`
String	String value Examples: `TEST`, `SWITCH1`
Time	Time value expressed in EXata time syntax (refer to *EXata User's Guide*) Examples: `1.5S`, `200MS`, `10US`

Filename	Name of a file in EXata filename syntax (refer to *EXata User's Guide*)
	Examples:
	`../../data/terrain/los-angeles-w`
	(For Windows and UNIX)
	`C:\scalable\exata\5.1\scenarios\WF\WF.nodes`
	(For Windows)
	`/root/scalable/exata/5.1/scenarios/WF/WF.nodes`
	(For UNIX)

Path	Path to a directory in EXata path syntax (refer to *EXata User's Guide*)
	Examples:
	`../../data/terrain` (For Windows and UNIX)
	`C:\scalable\exata\5.1\scenarios\default`
	(For Windows)
	`/root/scalable/exata/5.1/scenarios/default`
	(For UNIX)

IP Address	IPv4 or IPv6 address Examples: `192.168.2.1`, `2000:0:0:0::1`
IPv4 Address	IPv4 address Examples: `192.168.2.1`
IPv6 Address	IPv6 address Examples: `2000:0:0:0::1`
Coordinates	Coordinates in Cartesian or Lat-Lon-Alt system. The altitude is optional. Examples: `(100, 200, 2.5)`, `(-25.3478, 25.28976)`
Node-list	List of node IDs separated by commas and enclosed in "`{`" and "`}`". Examples: `{2, 5, 10}`, `{1, 3 thru 6}`
List	One of the enumerated values. Example: See the parameter `MOBILITY` in Table 1-4.

Note: If the parameter type is List, then options for the parameter available in EXata and the commonly used model libraries are enumerated. Additional options for the parameter may be available if some other model libraries or addons are installed. These additional options are not listed in this document but are described in the corresponding model library or addon documentation.

- ***<Range>*:** This is an optional entry and is used if the range of values that a parameter can take is restricted. The permissible range is listed after the label "*Range:*" The range can be specified by giving the minimum value, the maximum value, or both. If the range of values is not restricted, then this entry is omitted.

  If both the minimum and maximum values are specified, then the following convention is used to indicate whether the minimum and maximum values are included in the range:

`(min, max)`	`min` < parameter value < `max`
`[min, max)`	`min` ≤ parameter value < `max`
`(min, max]`	`min` < parameter value ≤ `max`
`[min, max]`	`min` ≤ parameter value ≤ `max`

`min` (or `max`) can be a parameter name, in which case it denotes the value of that parameter.

Examples of range specification are:

Range: ≥ `0`

Range: `(0.0, 1.0]`

Range: `[1, MAX-COUNT]`

Range: `[1S, 200S]`

Note: If an upper limit is not specified in the range, then the maximum value that the parameter can take is the largest value of the type (integer, real, time) that can be stored in the system.

- ***<Default>*:** This is an optional entry which specifies the default value of an optional or conditional-optional parameter. The default value is listed after the label "*Default:*"
- ***<Unit>*:** This is an optional entry which specifies the unit for the parameter, if applicable. The unit is listed after the label "*Unit:*". Examples of units are: meters, dBm, slots.

Description Column

The third column contains a description of the parameter. The significance of different parameter values is explained here, where applicable. In some cases, references to notes, other tables, sections in the User's Guide, or to other model libraries may be included here.

Table 1-4 shows examples of parameter descriptions using the format described above.

TABLE 1-4. Example Parameter Table

Parameter	Values	Description
`MOBILITY` Optional *Scope:* Global, Node	List: • `NONE` • `FILE` • `GROUP-MOBILITY` • `RANDOM-WAYPOINT` Default: `NONE`	Mobility model used for the node. If `MOBILITY` is set to `NONE`, then the nodes remain fixed in one place for the duration of the simulation. See Table 7-11 for a description of mobility models.
`BACKOFF-LIMIT` Required *Scope:* Subnet, Interface	Integer *Range:* `[4,10)` *Unit:* slots	Upper limit of backoff interval after collision. A backoff interval is randomly chosen between 1 and this number following a collision.
`IP-QUEUE-PRIORITY-QUEUE-SIZE` Required *Scope:* All *Instances:* queue index	Integer *Range:* `[1, 65535]` *Unit:* bytes	Size of the output priority queue.
`MAC-DOT11-DIRECTIONAL-ANTENNA-MODE` Optional *Scope:* All	List • `YES` • `NO` *Default:* `NO`	Indicates whether the radio is to use a directional antenna for transmission and reception.

1.2.2 Format for GUI Configuration

The GUI configuration section for a model outlines the steps to configure the model using the GUI. The following conventions are used in the GUI configuration sections:

Path to a Parameter Group

As a shorthand, the location of a parameter group in a properties editor is represented as a path consisting of the name of the properties editor, name of the tab within the properties editor, name of the parameter group within the tab (if applicable), name of the parameter sub-group (if applicable), and so on.

Example

The following statement:

Go to **Default Device Properties Editor > Interfaces > Interface # > MAC Layer**

is equivalent to the following sequence of steps:

1. Open the Default Device Properties Editor for the node.
2. Click the **Interfaces** tab.

3. Expand the applicable Interface group.
4. Click the **MAC Layer** parameter group.

The above path is shown in Figure 1-1.

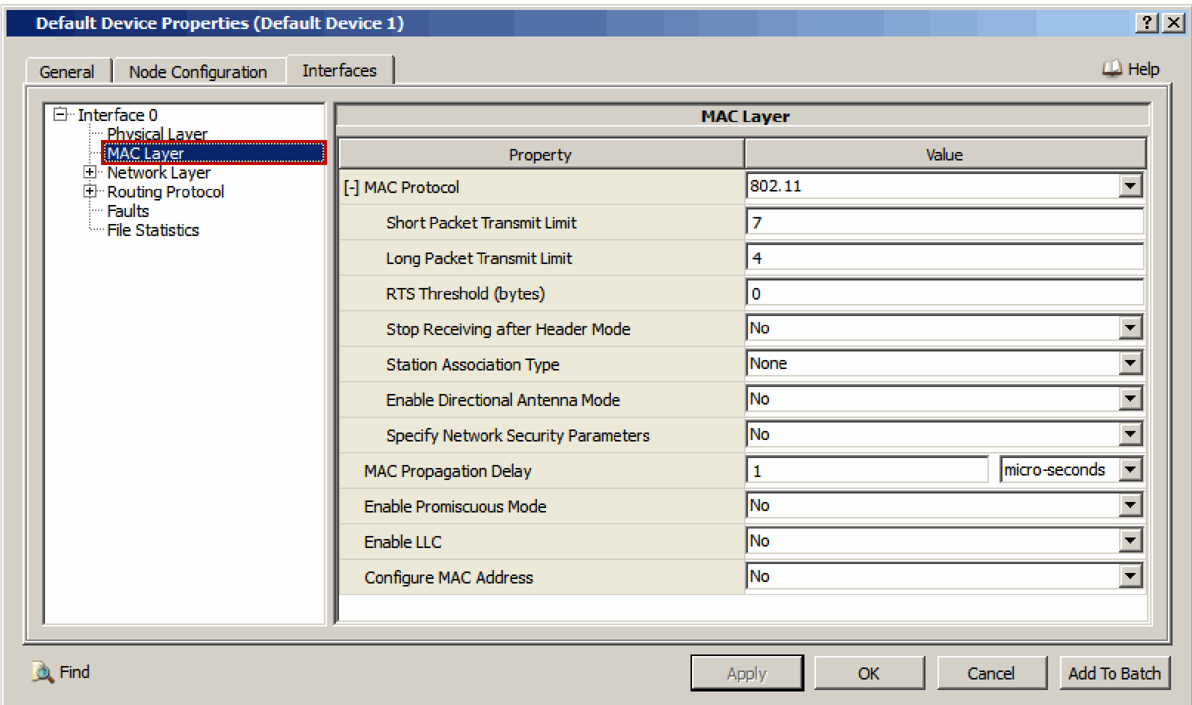


FIGURE 1-1. Path to a Parameter Group

Path to a Specific Parameter

As a shorthand, the location of a specific parameter within a parameter group is represented as a path consisting of all ancestor parameters and their corresponding values starting from the top-level parameter. The value of an ancestor parameter is enclosed in square brackets after the parameter name.

Example

The following statement:

Set **MAC Protocol** *[= 802.11]* **> Station Association Type** *[= Dynamic]* **> Set Access Point** *[= Yes]* **>** *Enable Power Save Mode* to *Yes*

is equivalent to the following sequence of steps:

1. Set **MAC Protocol** to *802.11*.
2. Set **Station Association Type** to *Dynamic.*
3. Set **Set Access Point** to *Yes*.
4. Set **Enable Power Save Mode** to *Yes*.

The above path is shown in Figure 1-2.

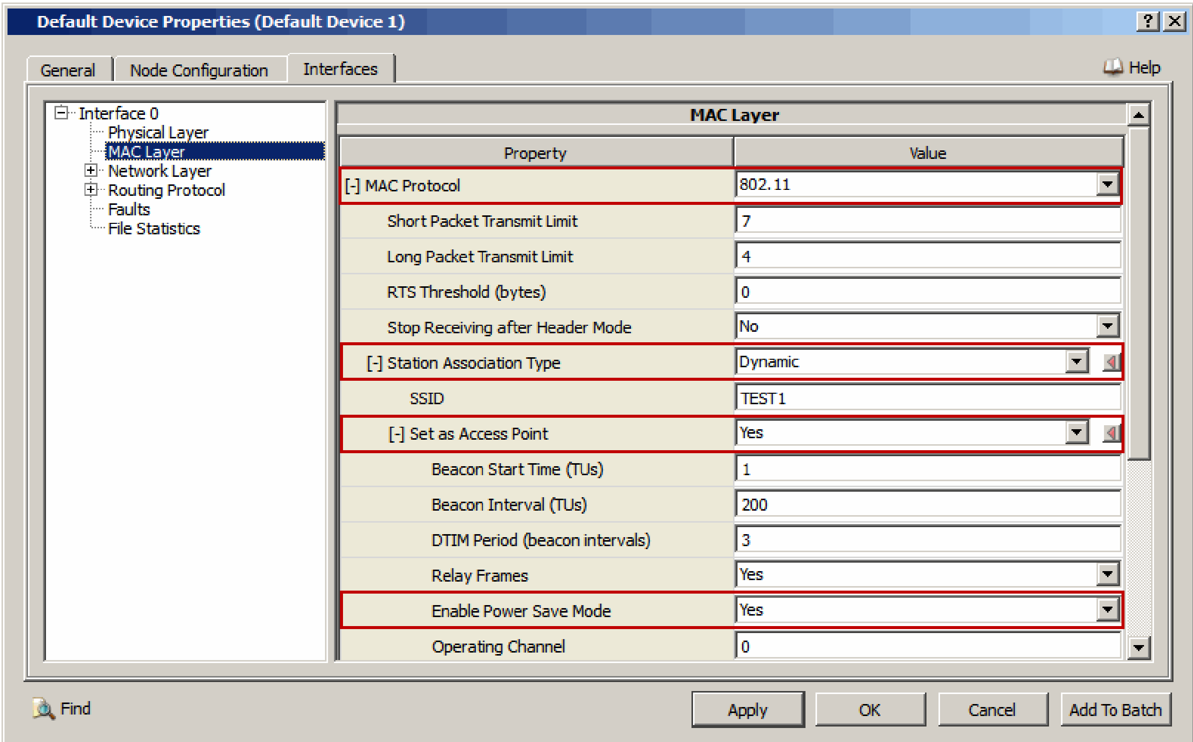


FIGURE 1-2. Path to a Specific Parameter

Parameter Table

GUI configuration of a model is described as a series of a steps. Each step describes how to configure one or more parameters. Since the GUI display name of a parameter may be different from the name in the configuration file, each step also includes a table that shows the mapping between the GUI names and command line names of parameters configured in that step. For a description of a GUI parameter, see the description of the equivalent command line parameter in the command line configuration section.

The format of a parameter mapping table is shown in Table 1-5.

TABLE 1-5. Mapping Table

GUI Parameter	Scope of GUI Parameter	Command Line Parameter
<GUI Display Name>	<Scope>	*<Command Line Parameter Name>*

The first column, labeled “GUI Parameter”, lists the name of the parameter as it is displayed in the GUI.

The second column, labeled “Scope of GUI Parameter”, lists the level(s) at which the parameter can be configured. <*Scope*> can be any combination of: Global, Node, Subnet, Wired Subnet, Wireless Subnet, Point-to-point Link, and Interface.

Table 1-6 lists the Properties Editors where parameters with different scopes can be set.

Notes: **1.** Unless otherwise stated, the “Subnet” scope refers to “Wireless Subnet”.

2. The scope column can also refer to Properties Editors for special devices and network components (such as ATM Device Properties Editor) which are not included in Table 1-6.

TABLE 1-6. Properties Editors for Different Scopes

Scope of GUI Parameter	Properties Editor
Global	Scenario Properties Editor
Node	Default Device Properties Editor (General and Node Configuration tabs)
Subnet Wireless Subnet	Wireless Subnet Properties Editor
Wired Subnet	Wired Subnet Properties Editor
Point-to-point Link	Point-to-point Link Properties Editor
Interface	Interface Properties Editor, Default Device Properties Editor (Interfaces tab)

The third column, labeled “Command Line Parameter”, lists the equivalent command line parameter.

Note: For some parameters, the scope may be different in command line and GUI configurations (a parameter may be configurable at fewer levels in the GUI than in the command line).

Table 1-7 is an example of a parameter mapping table.

TABLE 1-7. Example Mapping Table

GUI Parameter	Scope of GUI Parameter	Command Line Parameter
Define Area	Node	`OSPFv2-DEFINE-AREA`
OSPFv2 Configuration File	Node	`OSPFv2-CONFIG-FILE`
Specify Autonomous System	Node	`N/A`
Configure as Autonomous System Boundary Router	Node	`AS-BOUNDARY-ROUTER`
Inject External Route	Node	`N/A`
Enable Stagger Start	Node	`OSPFv2-STAGGER-START`

2 MAC Layer Models

This chapter describes features, configuration requirements and parameters, statistics, and scenarios for MAC Layer Models in the Cyber Model Library, and consists of the following section:

- Wired Equivalent Privacy and CTR with CBC-MAC Protocol

2.1 Wired Equivalent Privacy and CTR with CBC-MAC Protocol

The EXata WEP model is based on IEEE standard 802.11i-2004 and EXata CCMP model is based on IEEE standard 802.11-1997.

2.1.1 WEP/CCMP Description

WEP Description

Wired Equivalent Privacy (WEP) is a MAC layer security protocol that provides security for wireless LANs, equivalent to the security provided in wired LANs.

In WEP, a secret key is distributed to cooperating STAs using an external key management path, independent of the MAC layer. The secret key combined with an Initialization Vector (IV) resulting in a seed is given as an input to a Pseudo-Random Number Generator (PRNG). The PRNG outputs a key sequence (k) of pseudorandom octets.

An integrity algorithm operates on plaintext data to produce an ICV to protect against unauthorized data modification. The key sequence (k) is combined with the plaintext concatenated with the ICV to generate the cipher text. The secret key remains constant while the IV changes periodically. Thus, there is a one-to-one correspondence between the IV and k.

The WEP algorithm is applied to the frame body of an MPDU. The (IV, frame body, ICV) triplet forms the actual data to be sent in the data frame.

CTR with CBC-MAC Protocol (CCMP) Description

CCMP (CTR with CBC-MAC Protocol) is an RSNA data confidentiality and integrity protocol.

WEP is known to be insecure and is replaced by CCMP. CCMP is based on the CCM of the AES encryption algorithm. CCM is a generic authenticate-and-encrypt block cipher mode. A unique temporal key (for each session) and a unique nonce value (a value that’s used only once for each frame) are required for protecting the MPDUs. CCMP uses a 48-bit Packet Number (PN) to protect the MPDUs.

> **Note:** The PN is never repeated for a series of encrypted MPDUs using the same temporal key.

CCMP encrypts the payload of a plaintext MPDU and encapsulates the resulting cipher text using the following:

1. Increment the PN, so that each MPDU has a unique PN for the same temporal key.
2. Use the fields in the MPDU header to construct the additional authentication data (AAD) for CCM. The CCM algorithm provides integrity protection for the fields included in the AAD.
3. Construct the CCM Nonce block from the PN, A2, and the Priority field of the MPDU where A2 is MPDU Address 2. The Priority field has a reserved value set to 0.
4. Place the new PN and the key identifier into the 8-octet CCMP header.
5. Use the temporal key, AAD, nonce, and MPDU data to form the cipher text and MIC. This step is known as CCM originator processing.
6. Form the encrypted MPDU by combining the original MPDU header, the CCMP header, the encrypted data and MIC, as described in IEEE 802.11i-2004 Standard, Sec-8.3.3.2.

CCMP decrypts the payload of a cipher text MPDU and decapsulates plaintext MPDU using the following:

1. The encrypted MPDU is parsed to construct the AAD and nonce values.
2. The AAD is formed from the MPDU header of the encrypted MPDU.
3. The nonce value is constructed from the A2, PN, and Priority Octet fields (reserved and set to 0).
4. The MIC is extracted for use in the CCM integrity checking.
5. The CCM recipient processing uses the temporal key, AAD, nonce, MIC, and MPDU cipher text data to recover the MPDU plaintext data and, to check the integrity of the AAD and MPDU plaintext data.
6. The received MPDU header and the MPDU plaintext data from the CCM recipient processing can be concatenated to form a plaintext MPDU.
7. The decryption processing prevents replay of MPDUs by validating that the PN in the MPDU is greater than the replay counter maintained for the session.

The decapsulation process succeeds when the calculated MIC matches the MIC value obtained from decrypting the received encrypted MPDU. The original MPDU header is concatenated with the plaintext data resulting from the successful CCM recipient processing to create the plaintext MPDU.

2.1.2 Features and Assumptions

This section describes the implemented features, omitted features, assumptions and limitations of the WEP/CCMP model.

2.1.2.1 Implemented Features

- Generic WEP and CCMP encryption/decryption
- Crypto Latency

2.1.2.2 Omitted Features

- WEP and CCMP implementation for 802.11 broadcast traffic

2.1.2.3 Assumptions and Limitations

- All STAs running CCMP are RSNA capable.
- Every WEP/CCMP enable STA’s are listed in configuration file in the form of a table. For each STA in the table, a WEP/CCMP key is defined for every reachable destination from that STA. If an entry is not found for a RA in mappings table for a STA, WEP/CCMP is off for that STA.
- Default keys are not implemented in WEP and CCMP. Corresponding KeyID subfields in the respective headers will be zero.
- IV, ICV in WEP MPDU and PN, MIC in CCMP MPDU are dummy fields. ICV and MIC are not used to check erroneous packets.

2.1.3 Supplemental Information

WEP is flawed and is replaced by CCMP in 802.11i standard.

2.1.4 Command Line Configuration

To specify WEP as the MAC Security protocol, include the following parameter in the scenario configuration (.config) file:

```
[<Qualifier>]   MAC-SECURITY-PROTOCOL       WEP
```

To specify CCMP as the MAC Security protocol, include the following parameter in the scenario configuration file:

```
[<Qualifier>]   MAC-SECURITY-PROTOCOL       CCMP
```

The scope of this parameter declaration can be Global, Node, Subnet, or Interface. See Section 1.2.1.1 for a description of `<Qualifier>` for each scope.

Configuration Requirements

In order to run WEP or CCMP, the MAC protocol must be configured to be 802.11 MAC. See the 802.11 MAC protocol section of *Wireless Model Library* for details.

WEP/CCMP Parameters

Table 2-1 lists the WEP/CCMP parameters specified in the scenario configuration (.config) file. See Section 1.2.1.3 for a description of the format used for the parameter table.

TABLE 2-1. WEP/CCMP Parameters

Parameter	Value	Description
`WEP-RC4-DELAY` Optional *Scope:* Global, Node	Time *Range:* `[≥ 0]` *Default:* `10US`	Specifies the processing Delay for WEP's 'RC4' cryptographic algorithm.
`CCMP-AES-DELAY` Optional *Scope:* Global, Node	Time *Range:* `[≥ 0]` *Default:* `10US`	Specifies the processing Delay for CCMP's 'AES' encryption algorithm with CBC HMAC.
`WEP-CCMP-ALLOW-UNENC` Optional *Scope:* Global, Node	List: • `YES` • `NO` *Default:* `NO`	Specifies whether to allow 802.11 packets transmitted unencrypted without applying WEP or CCMP.

TABLE 2-1. WEP/CCMP Parameters (Continued)

Parameter	Value	Description
`WEP-CONFIG-FILE` Required *Scope:* All	Filename	Specifies the name of the WEP configuration file. The WEP configuration file contains the WEP protocol parameters. This file usually has the extension ".wep". The format of the .wep file is described in Section 2.1.4.1.
`CCMP-CONFIG-FILE` Required *Scope:* All	Filename	Specifies the name of the CCMP configuration file. The CCMP configuration file contains the CCMP protocol parameters. This file usually has the extension ".ccmp". The format of the .ccmp file is described in Section 2.1.4.1.

2.1.4.1 Format of the WEP and CCMP Configuration Files

All the security protocol related configuration parameters will be put into corresponding WEP or CCMP configuration files. These two files have the same format.

These files contain one-one key mappings table defined per destination (RA) for a given node as shown below:

```
KeyMappings <TA> <RA> <Key Type> <Key>
```

where

`<TA>`	Specifies the Transmitter address. It can be <node_id | interface address | subnet address>
`<RA>`	Specifies the receiver address. It can be <interface address | subnet address>
`<Key Type>`	Key Type is a string value. For .wep files the only possible value is WEP while for .ccmp it can be WEP or CCMP.
`<Key>`	Key is a string value which is the actual key value used for encryption/decryption.

For example, if you have two nodes (node 1 and 2). The entries in the file will be as follows:

```
KeyMappings 1 192.168.0.2 WEP ffa0
KeyMappings 192.168.0.1 192.168.0.2 CCMP ffa0
```

2.1.5 GUI Configuration

This section describes how to configure WEP or CCMP in the EXata GUI.

Configuration Requirements

To use WEP or CCMP in a scenario, the MAC Protocol must be set as *802.11.* Refer to *Wireless Model Library* for details of configuring MAC protocol parameters.

WEP/CCMP Configuration

To configure the WEP or CCMP, perform the following steps:

1. Go to one of the following locations:
    - To set properties at the subnet level, go to **Wireless Subnet Properties Editor > MAC Layer**.
    - To set properties at the interface level, go to one of the following locations:
        - **Interface Properties Editor > Interfaces > Interface # > Network Layer > MAC Layer**
        - **Default Device Properties Editor > Interfaces > Interface # > Network Layer > MAC Layer**.
2. To configure WEP, set **MAC Protocol** *[= 802.11]* **> Security Protocol** to *WEP* and set the dependent parameters listed in Table 2-2.

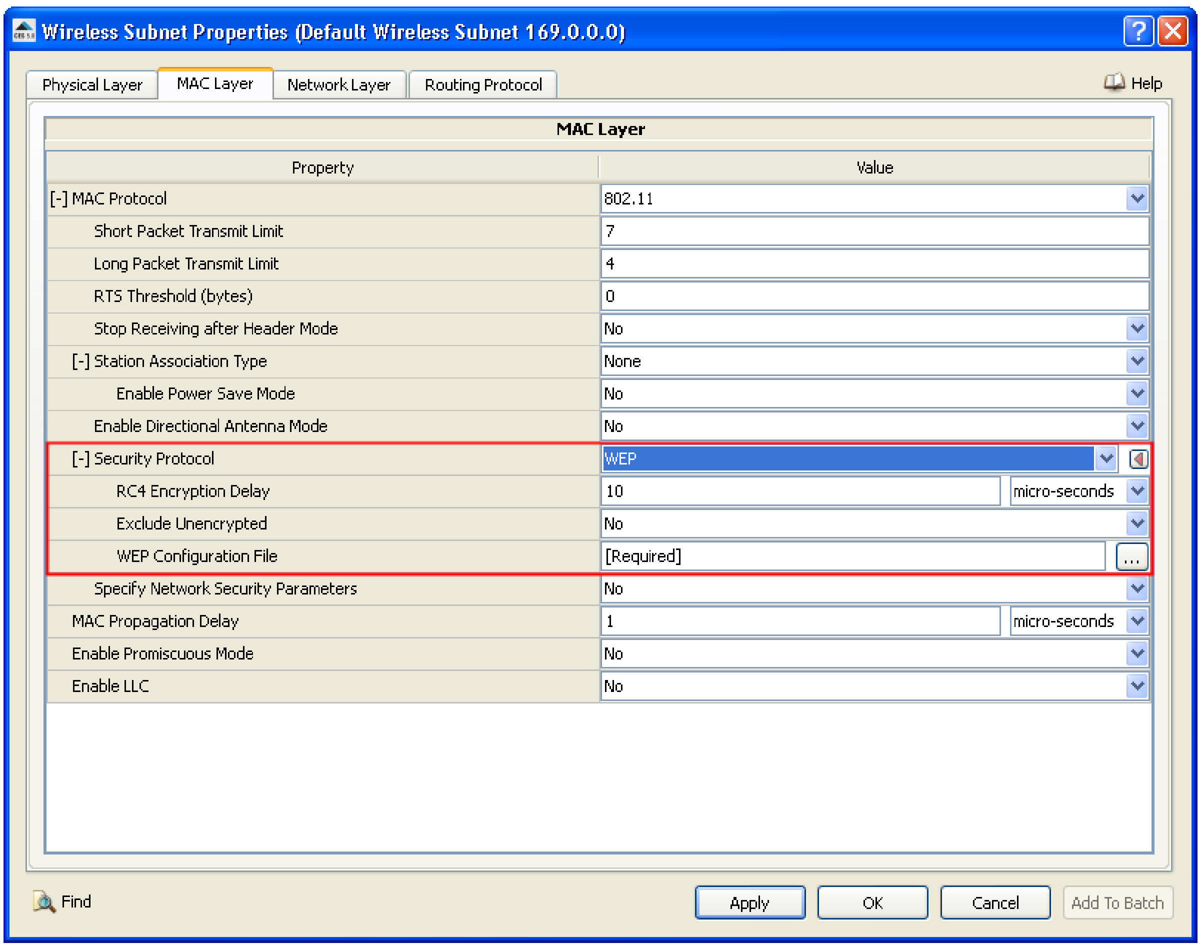


FIGURE 2-1. Setting WEP Parameters

TABLE 2-2. Command Line Equivalent of WEP Parameters

GUI Parameter	Scope of GUI Parameter	Command Line Parameter
RC4 Encryption Delay	Subnet, Interface	`WEP-RC4-DELAY`
Exclude Unencrypted	Subnet, Interface	`WEP-CCMP-ALLOW-UNENC`
WEP Configuration File	Subnet, Interface	`WEP-CONFIG-FILE`

3. To configure CCMP, set **MAC Protocol** *[= 802.11]* **> Security Protocol** to *CCMP* and set the dependent parameters listed in Table 2-3.

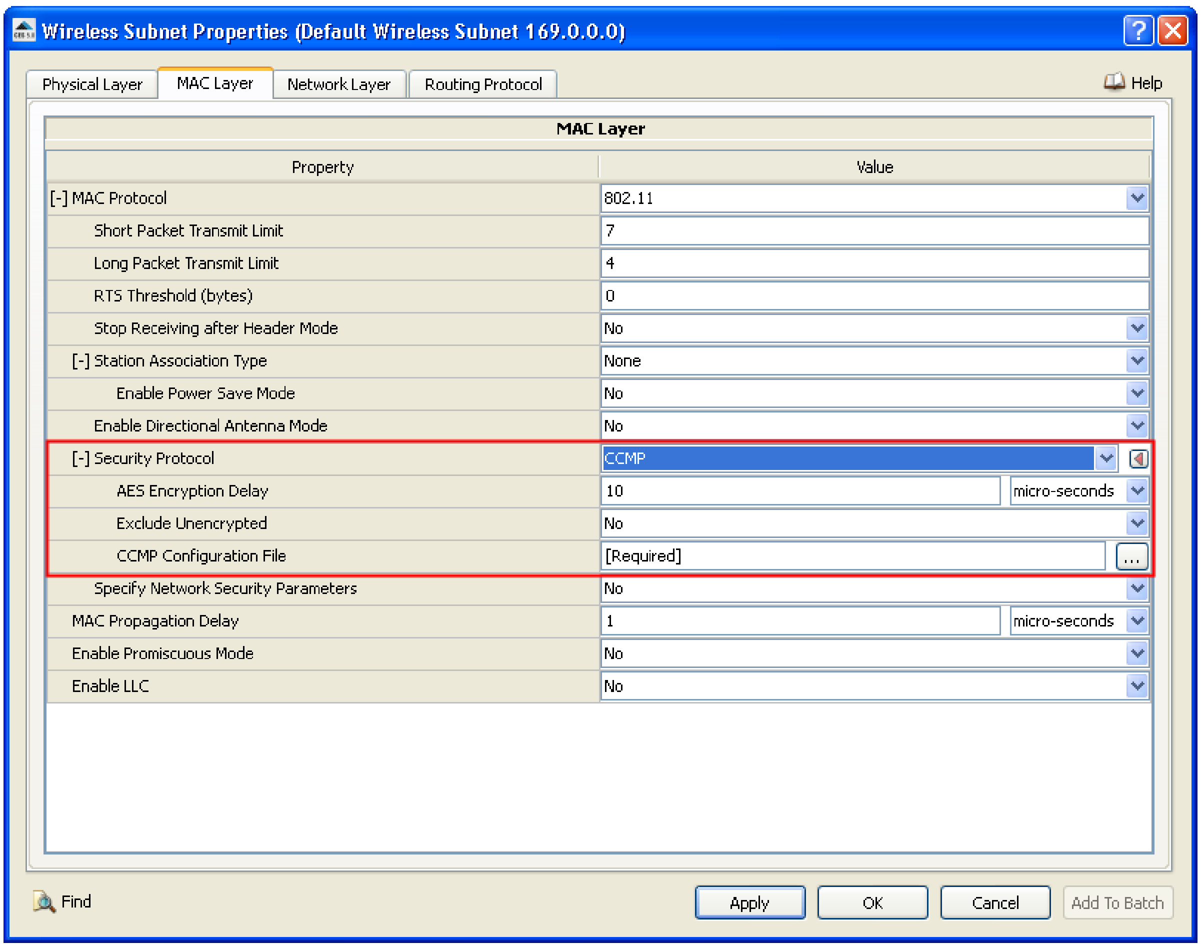


FIGURE 2-2. Setting CCMP Parameters

TABLE 2-3. Command Line Equivalent of CCMP Parameters

GUI Parameter	Scope of GUI Parameter	Command Line Parameter
AES Encryption Delay	Subnet, Interface	`CCMP-AES-DELAY`
Exclude Unencrypted	Subnet, Interface	`WEP-CCMP-ALLOW-UNENC`
CCMP Configuration File	Subnet, Interface	`CCMP-CONFIG-FILE`

2.1.6 Statistics

Table 2-4 lists the statistics collected for the WEP/CCMP model that are output to the statistics (.stat) file at the end of simulation..

TABLE 2-4. WEP/CCMP Statistics

Statistic	Description
WEP	
Packets Encrypted	Number of WEP encrypted packets.
Packets Decrypted	Number of WEP decrypted packets.
Packets Discarded	Number of non-WEP packets discarded by a STA on reception.
Packets Undecrypted	Number of protected packets unable to decrypt.
CCMP	
Packets Encrypted	Number of CCMP encrypted packets.
Packets Decrypted	Number of CCMP decrypted packets.
Packets Discarded	Number of non-WEP packets discarded by a STA on reception.
Packets Undecrypted	Number of protected packets unable to decrypt.

2.1.7 Sample Scenario

2.1.7.1 Scenario Description

In the sample scenario, five nodes (nodes 1 through 5) are connected through a wireless subnet. WEP or CCMP is enabled for the subnet.

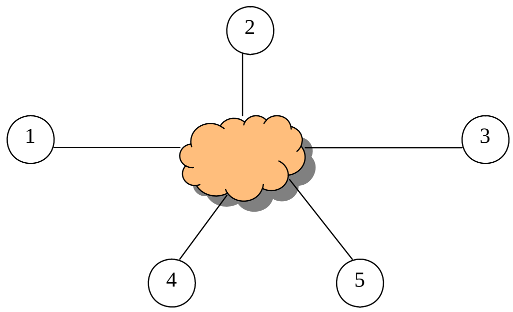


FIGURE 2-3. WEP/CCMP Sample Scenario

2.1.7.2 Command Line Configuration

WEP scenario

To configure the sample scenario using WEP, include the following lines in the scenario configuration (.config) file:

```
SUBNET N8-192.0.0.0 { 1 thru 5 } 451.95 1145.77 0.0
[ N8-192.0.0.0 ] MAC-PROTOCOL MACDOT11
[ N8-192.0.0.0 ] MAC-SECURITY-PROTOCOL WEP
[ N8-192.0.0.0 ] WEP-RC4-DELAY 5US
[ N8-192.0.0.0 ] WEP-CCMP-ALLOW-UNENC YES
[ N8-192.0.0.0 ] WEP-CONFIG-FILE wirelesssubnet-wep-on.wep
[ N8-192.0.0.0 ] NETWORK-PROTOCOL IP
```

Include the following lines int he WEP configuration file “wirelesssubnet-wep-on.wep”:

```
KeyMappings 1 192.0.0.3 WEP ffa0
KeyMappings 192.0.0.3 192.0.0.1 WEP ffa0
```

CCMP scenario

To configure the sample scenario using CCMP, include the following lines in the scenario configuration (.config) file:

```
SUBNET N8-192.0.0.0 { 1 thru 5 } 451.95 1145.77 0.0
[ N8-192.0.0.0 ] MAC-PROTOCOL MACDOT11
[ N8-192.0.0.0 ] MAC-SECURITY-PROTOCOL CCMP
[ N8-192.0.0.0 ] CCMP-AES-DELAY 5US
[ N8-192.0.0.0 ] CCMP-CONFIG-FILE wirelesssubnet-ccmp-on.ccmp
```

Include the following lines int he WEP configuration file “wirelesssubnet-ccmp-on.ccmp”:

```
KeyMappings 1 192.0.0.3 CCMP ffa0
KeyMappings 192.0.0.3 192.0.0.1 CCMP ffa0
```

2.1.7.3 GUI Configuration

WEP Scenario

Perform the following steps to create this sample scenario using the GUI:

1. Place five nodes of the Default device type and a wireless subnet on the canvas. Connect all the four nodes to the wireless subnet.
2. To configure WEP, go to the **Wireless Subnet Properties Editor > MAC Layer**. Set **MAC Protocol** *[= 802.11]* **> Security Protocol** to *WEP* as shown in Figure 2-1 and set the dependent parameters as below.
   - **RC4 Encryption Delay** to *5US*
   - **Exclude Unencrypted** to *YES*
   - **WEP Configuration File** to *wirelesssubnet-wep-on.wep*
3. Create the wirelesssubnet-wep-on.wep file as described in command line configuration section.

CCMP Scenario

Perform the following steps to create this sample scenario using the GUI:

1. Place five nodes of the Default device type and a wireless subnet on the canvas. Connect all the four nodes to the wireless subnet.
2. To configure CCMP, go to the **Wireless Subnet Properties Editor > MAC Layer**. Set **MAC Protocol** *[= 802.11]* **> Security Protocol** to *CCMP* as shown in Figure 2-1 and set the dependent parameters as below.
   - **AES Encryption Delay** to *5US*
   - **CCMP Configuration File** to *wirelesssubnet-ccmp-on.ccmp*
3. Create the wirelesssubnet-ccmp-on.ccmp file as described in command line configuration section.

2.1.8 Scenarios Included in EXata

The EXata distribution includes several sample scenarios for the WEP/CCMP model. All scenarios are located in the directory EXATA_HOME/scenarios/cyber/wep-ccmp. Table 2-5 lists the sub-directory where each scenario is located.

TABLE 2-5. WEP/CCMP Model Scenarios

Scenario Sub-directory	Description
Mixed-wep-ccmp-wep	Shows the wireless scenario with both WEP and CCMP capability
UnprotectedPackets-case-1	Shows the scenario with both WEP and CCMP configured
wirelesssubnets-wep-ccmp-on-mobility	Shows the mobility scenario with both WEP and CCMP configured

2.1.9 References

1. IEEE Std 802.11-1997. Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) specifications.
2. IEEE Std 802.11i-2004. Amendment 6: Medium Access Control (MAC) Security Enhancements.

3 Network Layer Models

This chapter describes features, configuration requirements and parameters, statistics, and scenarios for Network Layer Models in the Cyber Model Library, and consists of the following sections:

- Certificate Model
- Firewall Model
- Information Assurance Hierarchical Encryption Protocol (IAHEP) Model
- Internet Protocol Security (IPSec) Model
- Internet Security Association and Key Management Protocol with Internet Key Exchange (ISAKMP-IKE) Model
- Public Key Infrastructure Model
- Secure Neighbor Model

3.1 Credential Model: IFF Certificate

The EXata Certificate model is based on WTLSCert certificate defined in WAP WTLS WAP-199-WTLS Wireless Application Protocol Wireless Transport Layer Security Specification.

3.1.1 Description

The certificate model implements credentials for the purpose of authentication, IFF (Identification of Friend and Foe), authorization, access control, accounting and auditing. In digital signature systems built on top of public key crypto systems, a signature signed by private key SK can be verified by corresponding public key PK, and the signature cannot be forged by an adversary without knowing the signing key SK.

In a secured wireless network, each node must be capable of authenticating itself to its colleague network members, and vice versa. In EXata's Network Security modeling, every network member must acquire a signed credential from an offline authority or Certificate Authority (CA) prior to network operations. The credential is a certificate signed by the CA's private key SK_{CA}, and can be verified by the well-known public key PK_{CA}, which is assumed to be cached by every network member's local storage. In summary, at the time of *a priori* offline registration, network member *X* obtains PK_{CA} (CA's public key) and $CERT_X$ (X's own certificate signed by SK_{CA}).

The certificate $CERT_X$ is in the form of *[X,pk*$_X$*,validtime] signed_by_*SK_{CA} where unique id *X* is assigned to a node, pk_X is the certified public key of the id *X*, and *validtime* limits the valid period of the certificate. In EXata, *X* is a unique network address, like an IP address. For example, on a node having multiple network interfaces with IP addresses 11.11.11.11 and 22.22.22.22, the node must obtain two different certificates for both of its network interfaces, respectively.

This certificate modeling is provided for authentication services in the entire protocol stack. The current implementation uses a short certificate format defined by WTLS. Certificate renewal and revocation are not implemented. Distributed solutions of certificate renewal and revocation are discussed in Ubiquitous and Robust Security Architecture (URSA) and similar proposals relying on threshold cryptography. URSA proposes to distribute partial shares of the certificate signing key SK_{CA} to *n* nodes playing the role of partial CA, and *k* out of *n* partial CAs can produce *k* partial certificates which combine into a full certificate (or certificate-revocation/counter-certificate). The scheme tolerates up to *k-1* node intrusions and *n-k* node crashes.

3.1.2 Features and Assumptions

This section describes the implemented features, omitted features, assumptions and limitations of the Certificate model.

3.1.2.1 Implemented Features

- WAP WTLS certificate format
- Certificate for each IP interface

3.1.2.2 Omitted Features

- X.509 certificate format
- Actual cryptography
- Certificate Revocation List (CRL)

3.1.2.3 Assumptions and Limitations

- Actual cryptography can be added if the crypto module does not require SNT to open all sources.
- The required crypto modules are MD5, SHA1, AES, 3DES, and Elliptic Curve Cryptography.

3.1.3 Supplemental Information

None.

3.1.4 Command Line Configuration

To enable Certificate model, include the following parameter in the scenario configuration (.config) file:

```
[<Qualifier>] CERTIFICATE-ENABLED    YES
```

The scope of this parameter declaration can be Global, Node, Subnet, or Interface. See Section 1.2.1.1 for a description of `<Qualifier>` for each scope.

Note: The default value of this parameter is NO.

Certificate-specific Parameters

Table 3-1 lists the Certificate parameters specified in the scenario configuration (.config) file. See Section 1.2.1.3 for a description of the format used for the parameter table.

TABLE 3-1. Certificate-specific Parameters

Parameter	Value	Description
CERTIFICATE-FILE-LOG Optional *Scope:* All	List: • YES • NO *Default:* YES	Specifies whether the certificate contents are logged in a file. YES: If this parameter is set to YES, the certificate contents are logged in the file "default.certificate.<interface-address>". NO: If this parameter is set to NO, no certificate log file is generated.

Examples of Parameter Usage

The following configuration enables certificate model in a wireless subnet:

```
[ N8-192.0.0.0 ] CERTIFICATE-ENABLED YES
[ N8-192.0.0.0 ] CERTIFICATE-FILE-LOG YES
```

3.1.5 GUI Configuration

To configure the Certificate model parameters, perform the following steps:

1. Go to one of the following locations:
   - To set properties at the subnet level, go to **Wireless Subnet Properties Editor > Network Layer > Cyber**.
   - To set properties at the node level, go to **Default Device Properties Editor > Node Configuration > Cyber**.

- To set properties at the interface level, go to one of the following locations:
  - **Interface Properties Editor > Interfaces > Interface # > Network Layer > Cyber**.
  - **Default Device Properties Editor > Interfaces > Interface # > Network Layer > Cyber**.

In this section, we show how to configure IFF Certification parameters for a specific node using the Default Device Properties Editor. Parameters can be set in the other properties editors in a similar way.

2. Set Enable IFF Certification to Yes and set the dependent parameters listed in the Table 3-2.

- Set **Do Certificate File Log** to *Yes*, if certificate contents are needed to be logged in a file. Otherwise, set it to *No*.

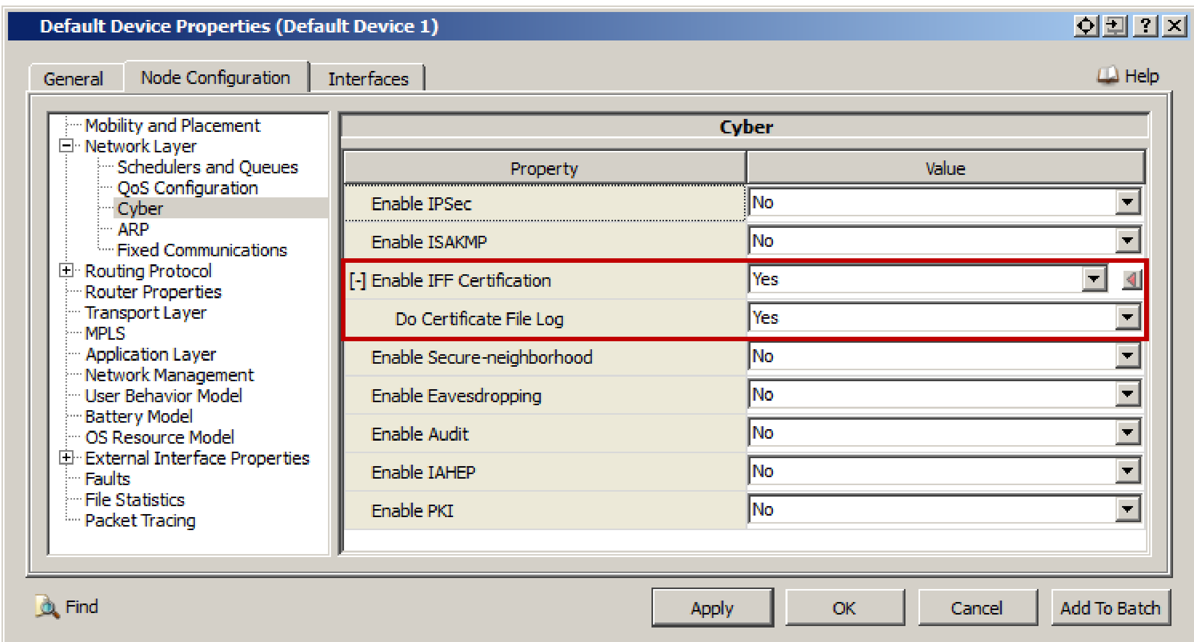


FIGURE 3-1. Enable Certificate Model

TABLE 3-2. Command Line Equivalent of IFF Certification Parameters

GUI Parameter	Scope of GUI Parameter	Command Line Parameter
Do Certificate File Log	Node, Subnet, Interface	`CERTIFICATE-FILE-LOG`

3.1.6 Statistics

There are no statistics generated for the Certificate model.

3.1.7 Scenarios Included in EXata

The EXata distribution includes several sample scenarios for the Certificate model. All scenarios are located in the directory EXATA_HOME/scenarios/cyber/certification. Table 3-3 lists the sub-directory where each scenario is located.

TABLE 3-3. Certificate Model Scenarios

Scenario Sub-directory	Description
wtls-interface-test	Shows the functionality of WTLS certification implementation in a scenario when certificate is enabled on the interfaces.
wtls-node-test	Shows the functionality of WTLS certification implementation in a wireless scenario when certificate is enabled on all the nodes.
wtls-subnet-test	Shows the functionality of WTLS certification implementation in a scenario when certificate is enabled on the subnet.
wtls-wired-test	Shows the functionality of WTLS certification implementation in a wired scenario when certificate is enabled on all the nodes.

3.1.8 References

1. [WTLS] WAP Forum. Wireless Transport Layer Security (Version 06-Apr-2001), Wireless Application Protocol, WAP-261-WTLS-20010406-a.
2. [SanzgiriDLSR02] Kimaya Sanzgiri, Bridget Dahill, Brian Neil Levine, Clay Shields, Elizabeth Royer, “A Secure Routing Protocol for Ad Hoc Networks”, pp.78-89, in Proceedings of The Tenth IEEE International Conference on Network Protocols (ICNP), 2002. November 12-15. Paris, France.

3.2 Firewall Model

3.2.1 Description

Firewalls are software or hardware components in a computer host or system that are used to implement network access and security policies. All traffic must pass through firewalls, which determines, based on access or security policies, the traffic that is allowed to pass through the network, or dropped at the firewalls.

The firewall model in EXata is a *packet-based stateless software* firewall. That is, the firewall model in EXata is a software process that inspects each packet to determine if the said packet should be allowed or denied access. The firewall model is stateless, that is, it does not retain state once a packet has been processed by the firewall.

The firewall model in EXata is based on the *iptables* packet filter software found in Linux/Unix based systems (see [1]). Specifically, the model in EXata models the *Filter* table of iptables, which is used for firewall actions (other tables, such as NAT, MANGLE, etc., are used for packet filtering and modification actions that are unrelated with firewalls).

3.2.2 Features and Assumptions

3.2.2.1 Implemented Features

- The FILTER table of iptables.
- The INPUT, OUTPUT, and FORWARD chains of the FILTER table.
- Table actions: create a chain.
- Chain actions: set default policy, append rule, insert rule.
- Firewall rule predicates:
  - From MAC header: source address, destination address.
  - From IP header: protocol, source address, destination address, source address type, destination address type, fragmentation flags.
  - From TCP/UDP header: source port, destination port.
  - From TCP header: TCP flags.
  - Incoming interface index.
  - Outgoing interface index.
- Firewall rule actions:
  - Drop a packet
  - Accept a packet
  - Jump to a custom chain
  - Goto a custom chain
  - Return to the parent chain.

3.2.2.2 Omitted Features

- Delete or rename custom chains
- Erase or replace rules in a chain

3.2.2.3 Assumptions and Limitations

- The firewall model is a stateless implementation, which implies rules cannot be configured, therefore relies on previous state of the firewall, such as connection tracking.

3.2.3 Supplemental Information

None.

3.2.4 Command Line Configuration

To specify Firewall model, include the following parameter in the scenario configuration (.config) file:

```
FIREWALL-CONFIG-FILE    <firewall-config-file>
```

where

`<firewall-config-file>`	Name of the file that enumerates the firewall rules for the nodes in the scenario. It is recommended that the filename extension of this file be *.firewall*.

3.2.4.1 Format of Firewall Configuration File

The firewall commands are defined in the Firewall Configuration file, one command per line. Commands for more than one node can be specified in the same Firewall Configuration file. The Firewall model is activated for those nodes in a scenario only for which there is at least one rule defined in the Firewall Configuration file.

There are three "chains" that handle different classes of traffic as follows:

- INPUT: This chain inspects all incoming packets that are successfully received by the host and are passed on to the Application layer.
- OUTPUT: This chain inspects all outgoing packets that were generated by applications on the host.
- FORWARD: This chain inspects all packets that are forwarded by the host. That is, these packets were neither generated by nor delivered to the applications on this host; rather these are the packets that are forwarded by the Network layer of the host.

Firewall commands are defined to configure the behavior of the firewall. A firewall command has two parts: Table Description and Chain Description. The syntax of these is identical to the iptables syntax.

- Table Description: Identify the table to which the command applies.
- Chain Description: Identify the chain within the table to which the command applies. These can be the three predefined chains: `INPUT`, `OUTPUT`, or `FORWARD`, or it can be a user-defined chain.

The chain description specifies the action to perform on the chain: create a new chain, rename a chain, or append a firewall rule to a chain. A firewall rule has two components: Conditions and Action.

- Conditions: Conditions on the properties or characteristics of a packet (e.g., the IP source address, the TCP flags, etc) that must match for this rule can be activated.
- Action: Action to be performed on a packet if it has been matched against the specified conditions. Two actions are supported in the Firewall model in EXata: drop the packet or accept the packet.

The syntax for specifying a firewall command is (all parameters are entered on the same line):

```
FIREWALL <node-id> <table-desc> <chain-desc>
```

where

`<node-id>`	Node ID of the node for which this rule is configured.
`<table-desc>`	Table description. The table description clauses are described in Table 3-4. **Note:** Table description is optional in a firewall command.
`<chain-desc>`	Chain description. This specifies the chain and the operation to perform on the chain. The chain description clauses are described in Table 3-5. **Note:** One and only one of the chain description commands must be specified.

Table Clauses

Table 3-4 describes the table clause of firewall commands.

TABLE 3-4. Table Clauses

Command	Description

`-t <table-name>`
or
`--table <table-name>`
	Indicates the table to which this rule applies.
	Since the firewall model in EXata supports only one table (the FILTER table), this command is optional.
	The only valid value for `<table-name>` is `FILTER`.

Chain Clauses

Table 3-5 describes the chain clauses of firewall commands.

TABLE 3-5. Chain Clauses

Command	Description

`-P <chain-name> <policy>`
or
`--policy <chain-name> <policy>`
	Set the default policy of the chain.
	`<chain-name>` is the name of the chain.
	`<policy>` is one of `DROP` or `ACCEPT`.

`-N <new-chain-name>`
or
`--new-chain <new-chain-name>`
	Create a new chain in the table.
	`<new-chain-name>` must be different from predefined or previously defined chains.

`-A <chain-name> <conditions> <action>`
or
`--append <chain-name> <conditions> <action>`
	Append the `<conditions>` and `<action>` pair as the last rule in the chain specified by `<chain-name>`.
	`<conditions>` specifies the condition(s) that a packet must satisfy and `<action>` specifies the action to perform on the packet if the condition(s) is true.
	`<conditions>` can have zero, one, or multiple *predicate clauses*. If no predicate clause is specified, then all packets satisfy the firewall condition. If more than one predicate clause is specified, then the firewall condition is satisfied if all predicates evaluate to true.
	The predicate clauses for firewall rules are described in Table 3-6.
	The action clauses for firewall rules are described in Table 3-7.
	Note: One and only one of the action clauses must be specified.

Syntax for Predicate Clauses

Table 3-6 describes the predicate clauses in firewall rules.

TABLE 3-6. Predicate Clauses in Firewall Rules

Command	Description
-p <protocol> or --protocol <protocol>	Match the protocol ID field of the IP header. The <protocol> can be an integer value, or one of the following strings: tcp: Match protocol ID = 6 (TCP protocol ID) udp: Match protocol ID = 17 (UDP protocol ID) icmp: Match protocol ID = 1 (ICMP protocol ID) all: Match all protocols This command has a negation form: -p ! <protocol> or --protocol ! <protocol> The clause will match only if the protocol in the packet header is not as indicated by <protocol>.
-s <address> or --src <address> or --source <address>	Match the source address field of the IP header. <address> can be one of the following: •Host address (e.g., 10.10.0.1) •Subnet address, in host bit notation(e.g., 10.10.0.0/24) •Subnet address, in subnet mask notation (e.g., 10.10.0.0/ 255.255.255.0) This command has a negation form: -s ! <address> or --src ! <address> or --source ! <address>
-d <address> or --dst <address> or --destination <address>	Match the destination address field of the IP header. <address> can be one of the following: •Host address (e.g., 10.10.0.1) •Subnet address, in host bit notation(e.g., 10.10.0.0/24) •Subnet address, in subnet mask notation (e.g., 10.10.0.0/ 255.255.255.0) This command has a negation form: -d ! <address> or --dst ! <address> or --destination ! <address>

TABLE 3-6. Predicate Clauses in Firewall Rules (Continued)

Command	Description

`-f`
or
`--fragment`
	Match the second or later fragments of a fragmented packet.
	
	This command has a negation form:
	! -f
	! --fragment
	This will match the first fragment or unfragmented packets.

`--sport <port>`
or
`--source-port <port>`
	Match the source port field of the TCP or UDP header.
	`<port>` can be one of the following:
	•An integer value (e.g., `--sport 80` will match packets with source port equal to 80)
	•A range (e.g., 80:100 will match ports in range 80 through 100, inclusive).
	Either the lower or the upper end of the range can be omitted.
	
	Examples:
	`--sport :80` will match all ports that are less than or equal to 80.
	`--sport 80:` will match all ports that are greater than or equal to 80.
	
	The command has a negation form:
	`--sport ! <port>`
	or
	`--source-port ! <port>`

`--dport <port>`
or
`--destination-port <port>`
	Match the destination port field of the TCP or UDP header.
	`<port>` can be one of the following:
	•An integer value (e.g., `--sport 80` will match packets with destination port equal to 80)
	•A range (e.g., 80:100 will match ports in range 80 through 100, inclusive).
	Either the lower or the upper end of the range can be omitted.
	
	Examples:
	`--sport :80` will match all ports that are less than or equal to 80.
	`--sport 80:` will match all ports that are greater than or equal to 80.
	
	The command has a negation form:
	`--dport ! <port>`
	or
	`--destination-port ! <port>`

TABLE 3-6. Predicate Clauses in Firewall Rules (Continued)

Command	Description

`--tcp-flags <mask> <check>`	Match when the TCP flags are set as specified.
	`<mask>` is the flags which are examined, written as a comma-separated list.
	`<check>` is a comma-separated list of flags which must be set.
	Flags are: `SYN, ACK, FIN, RST, URG, PSH, ALL, NONE`.
	
	Example:
	`tcp --tcp-flags SYN,ACK,FIN,RST SYN`
	This will only match packets with the SYN flag set, and the ACK, FIN and RST flags unset.
	
	This command has negation form:
	`--tcp-flags ! <mask> <check>`

`-i <index>`
or
`--in-interface <index>`
	Match packets that arrive from interface index equal to `<index>`.
	`<index>` is an integer.
	
	This command has a negation form:
	`-i ! <index>`
	or
	`--in-interface ! <index>`
	Note: This predicate cannot be used with the OUTPUT chain.

`-o <index>`
or
`--out-interface <index>`
	Match packets that will be transmitted from interface index equal to `<index>`.
	`<index>` is an integer.
	
	This command has a negation form:
	`-o ! <index>`
	or
	`--out-interface ! <index>`
	Note: This predicate cannot be used with the INPUT chain.

Syntax for Action Clauses

Table 3-7 describes the action clauses in firewall rules.

TABLE 3-7. Actions Clauses in Firewall Rules

Command	Description
-j <action> or --jump <action>	Action to be performed on the packet. <action> can be one of the following: ACCEPT: Accept the packet and allow it to proceed in the protocol stack. No further rules are checked for this packet. DROP: Drop the packet. No further rules are checked for this packet. <chain-name>: Jump immediately to the first rule in the chain <chain-name>, and continue matching rules from that chain. If that chain has not provided any response, continue with the next rule in the current chain. RETURN: Go back to the parent chain from which the current chain was called (via --jump or --goto command).
-g <chain-name> or --goto <chain-name>	Jump immediately to the first rule in the chain <chain-name>, and continue matching rules from that chain. Unlike the --jump option return will not continue processing in the current chain; instead in the parent chain that called the current chain via --jump.

Example

As an example, consider the following firewall command in the Firewall Configuration file:

```
-t FILTER -A INPUT -s 10.10.0.0/24 --tcp-flags SYN,ACK SYN
--in-interface 0  -j DROP
```

This command specifies that a rule must be appended to the INPUT chain of the FILTER table.

The firewall rule to be appended has three predicates:

- -s 10.10.0.0/24: This indicates that the source address in the IP header must belong to the subnet 10.10.0.0/24t.
- --tcp-flags SYN,ACK SYN: This indicates that the SYN flag of the TCP header must be set, and the ACK flag must not be set.
- --in-interface 0: This indicates that the packet must have arrived from interface index 0 on this host.

The action to perform if all three predicates are true is to drop the packet.

To summarize, the above rule instructs the firewall to drop all packets that arrive on interface index 0, from subnet 10.10.0.0/24, and have the SYN flag set and the ACK flag not set in the TCP header.

3.2.5 GUI Configuration

To configure the Firewall model in the GUI, do the following:

1. Go to **Scenario Property Editor > Cyber**.
2. Set the parameters listed in Table 3-8.

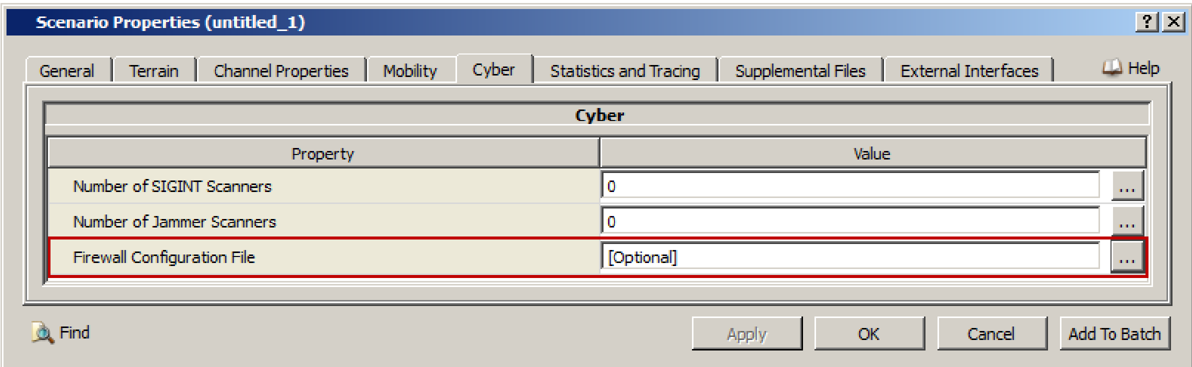


FIGURE 3-2. Setting Firewall Configuration File

TABLE 3-8. Command Line Equivalent of Firewall Configuration File Parameters

GUI Parameter	Scope of GUI Parameter	Command Line Parameter
Firewall Configuration File	Global	`FIREWALL-CONFIG-FILE`

Setting Parameters

- Set **Firewall Configuration File** to the name of the Firewall Configuration file. See Section 3.2.4.1 for the format of this file.

3.2.6 Runtime Commands for Firewall Model

Firewall commands can executed during the scenario execution Human-In-The-Loop (HITL) interface of the EXata GUI (see Section 6.1).

To interact with the firewall model at runtime, execute the following command from the HITL interface:

```
firewall <node-id> <table-desc> <chain-desc>
```

The syntax of this command is described in Section 3.2.4.

3.2.7 Statistics

Table 3-9 lists the Firewall model statistics that are output to the statistics (.stat) file at the end of simulation.

TABLE 3-9. Firewall Model Statistics

Statistic	Description
INPUT Chain Number of Packets Inspected	Number of packet that were inspected by the INPUT chain.
INPUT Chain Number of Packets Dropped	Number of packet that were denied access by the INPUT chain.
OUTPUT Chain Number of Packets Inspected	Number of packet that were inspected by the OUTPUT chain.
OUTPUT Chain Number of Packets Dropped	Number of packet that were denied access by the OUTPUT chain.
FORWARD Chain Number of Packets Inspected	Number of packet that were inspected by the FORWARD chain.
FORWARD Chain Number of Packets Dropped	Number of packet that were denied access by the FORWARD chain.

3.2.8 Scenarios Included in EXata

The EXata distribution includes several sample scenarios for the Firewall model. All scenarios are located in the directory EXATA_HOME/scenarios/cyber/firewall. Table 3-10 lists the sub-directory where each scenario is located.

TABLE 3-10. Firewall Scenarios Included in EXata

Scenario	Description
dmz-network	Represents a canonical enterprise network with private and DMZ networks. Firewall model is used to restrict access to internal network.
blacklist-node	A multihop wireless network where one node is blacklisted. Any node in the network will not receive packets from this node.

3.2.9 References

1. The netfilter.org project: http://www.netfilter.org.

3.3 Information Assurance Hierarchical Encryption Protocol (IAHEP)

3.3.1 Description

Information Assurance Hierarchical Encryption Protocol (IAHEP) is an encryption protocol that allows two or more secure enclaves to exchange data over an untrusted network.

In the IAHEP mechanism, nodes are classified as *black*, IAHEP, or *red*. An IAHEP module is formed by connecting an IAHEP node to a red node and a black node by wired links. In addition, a red node is connected to communicating nodes and a black node is connected to an untrusted network. The interface of an IAHEP node that communicates with a black node is classified as a black interface and the interface of the IAHEP node that communicates with a red node is classified as a red interface. All interfaces of a black node are classified as black interfaces. All interfaces of a red node are classified as red interfaces.

At the IAHEP node, packets are handled as follows:

1. When a packet from a black interface arrives, the packet's black header is removed and the packet is forwarded to the appropriate red interface. Routing control packets originating from the local black interface are ignored.
2. When an OSPF packet from a red interface arrives, a *black header* is added to it. The packet is then encrypted and authenticated with a red-network-wise key and forwarded to the black node.
3. When a non-OSPF packet from the red interface arrives, its processing depends upon whether an IP Security (IPsec) Security Association (SA) is established:
    a. If an IPsec SA towards the next red hop is not yet established, the packet is dropped and a request packet is sent to the next red hop to establish IPsec SA.
    b. If an IPsec SA is already available, a *black header* is added to it. The packet is then encrypted and authenticated with the SA key and forwarded to the black node.

3.3.2 Features and Assumptions

This section describes the implemented features, omitted features, assumptions and limitations of the IAHEP model.

3.3.2.1 Implemented Features

- IAHEP nodes with multiple red or black interfaces.
- Red and black multi-level security information is built into every node.
- Static Address Mapping Database (AMD) that provides a mapping between red and black IAHEP interface addresses.
- Fragmentation and reassembly.
- Bit-padding to last fragmentation unit.
- After Internet Security Association and Key Management Protocol (ISAKMP) establishes the SA for IPsec protocol Encapsulated Secure Payload (ESP) tunnel, this IPsec SA starts to work immediately.
- Processing delays and related statistics of IAHEP operations.

3.3.2.2 Omitted Features

- No actual crypto for encryption, decryption and authentication

3.3.2.3 Assumptions and Limitations

- The IAHEP model works only with IPv4 networks.

3.3.3 Supplemental Information

None.

3.3.4 Command Line Configuration

Configuration Requirements

- Each IAHEP node must be connected to one black node and one red node by point-to-point links. A black node can have interfaces to other black nodes or an IAHEP node. A red node can have interfaces to other red nodes or an IAHEP node.
- No routing protocol should be used on the interfaces of a IAHEP node and on the interface of a black node connecting to a IAHEP node(i.e., the parameter `ROUTING-PROTOCOL` should be set to `NONE` for those interfaces).

IAHEP Parameters

Table 3-11 lists the configuration parameters for the IAHEP model. See Section 1.2.1.3 for a description of the format used for the parameter table.

TABLE 3-11. IAHEP Parameters

Parameter	Value	Description
`IAHEP-NODE-TYPE` *Optional* *Scope:* Node	List: • `IAHEP` • `RED` • `BLACK`	Configures the node as a dedicated IAHEP node, red node, or black node. **Note:** For each node configured as an IAHEP node, another node must be configured as black node, and another node must be configured as a red node.
`IAHEP-INTERFACE-TYPE` *Optional* *Scope:* Interface, Subnet	List: • `RED` • `BLACK`	This parameter configures the interfaces between IAHEP and black nodes and between IAHEP and red nodes. This parameter is required for black-to-HAIPE, HAIPE-to-black, red-to-HAIPE, and HAIPE-to-red interfaces. This parameter must be set to `BLACK` for black-to-HAIPE and HAIPE-to-black interfaces, and to `RED` for red-to-HAIPE and HAIPE-to-red interfaces.
`IAHEP-ENCAPSULATION-OVERHEAD-SIZE` *Optional* *Scope:* Global, Node	Integer *Range:* ≥ `0` *Default :* `0` *Unit :* bytes	Size of overhead fields of an IAHEP encapsulation packet.
`IAHEP-ENCRYPTION-RATE` *Optional* *Scope:* Global, Node	Integer *Range:* `[≥ 0` *Default :* `0` *Unit :* bps	Encryption rate. The delay for encrypting a packet depends on the packet size and the encryption rate.
`IAHEP-AUTHENTICATION-RATE` *Optional* *Scope:* Global, Node	Integer *Range:* `[≥ 0` *Default :* `0` *Unit :* bps	Authentication rate. The delay for authenticating a packet depends on the packet size and the authentication rate.

TABLE 3-11. IAHEP Parameters (Continued)

Parameter	Value	Description
`IAHEP-AMD-FILE` *Optional* *Scope:* Global	Filename	Name of the IAHEP AMD file. The file specifies the mapping between a red node's IAHEP interface address and the corresponding IAHEP node's black interface address. This parameter is required if any node is configured as an IAHEP node. The format of the IAHEP AMD file is described in Section 3.3.4.1.
`MLS-STAR-PROPERTY` *Optional* *Scope:* Global	List: • `LIBERAL` • `STRONG` *Default :* `STRONG`	Specifies the communication security policy. `STRONG` A packet can be sent from a source to a destination only if the intermediate IAHEP nodes have the same MLS security level. `LIBERAL` A packet can be sent from a source to a destination if the intermediate IAHEP nodes have the same MLS security level. In addition, an IAHEP node with a lower security level can forward packets to an IAHEP node with a higher security level (but not vice-versa). The MLS security level of IAHEP nodes is specified in the IAHEP AMD file.

3.3.4.1 Format of the IAHEP AMD File

The IAHEP AMD file provides a mapping between a red node's IAHEP interface address and the corresponding IAHEP node's black interface address.

Each line in the IAHEP AMD file has the following format (all entries should be on the same line):

```
<Red Node's IAHEP-Interface> <IAHEP Node's Black-Interface>
<MLS Level>
```

where

`<Red Node's IAHEP-Interface>`	IP address of the IAHEP interface on the red IAHEP node.
`<IAHEP Node's Black Interface>`	IP address of the black interface on the corresponding IAHEP node.
`<MLS Level>`	Positive integer value specifying the MLS security level for this IAHEP node. The recommended value for this entry is 1.

Example

The following is an example of the IAHEP AMD file:

```
190.0.7.1  190.0.5.2  1
190.0.11.1 190.0.9.2  1
190.0.2.2  190.0.4.1  1
```

3.3.5 GUI Configuration

In the GUI, IAHEP properties are configured for nodes and for interfaces.

Configuration Requirements

- Each IAHEP node must be connected to one black node and one red node by point-to-point links. A black node can have interfaces to other black nodes or an IAHEP node. A red node can have interfaces to other red nodes or an IAHEP node.
- No routing protocol should be used on the interfaces of a IAHEP node and on the IAHEP interface of a black node (i.e., the parameter `ROUTING-PROTOCOL` should be set to `NONE` for those interfaces).

Configuring Node Properties

To configure IAHEP properties for a node, do the following:

1. Go to **Default Device Properties Editor > Node Configuration > Network Layer > Cyber**.
2. To enable IAHEP at the node, set **Enable IAHEP** to *Yes* and set the dependent parameters listed in Table 3-12.

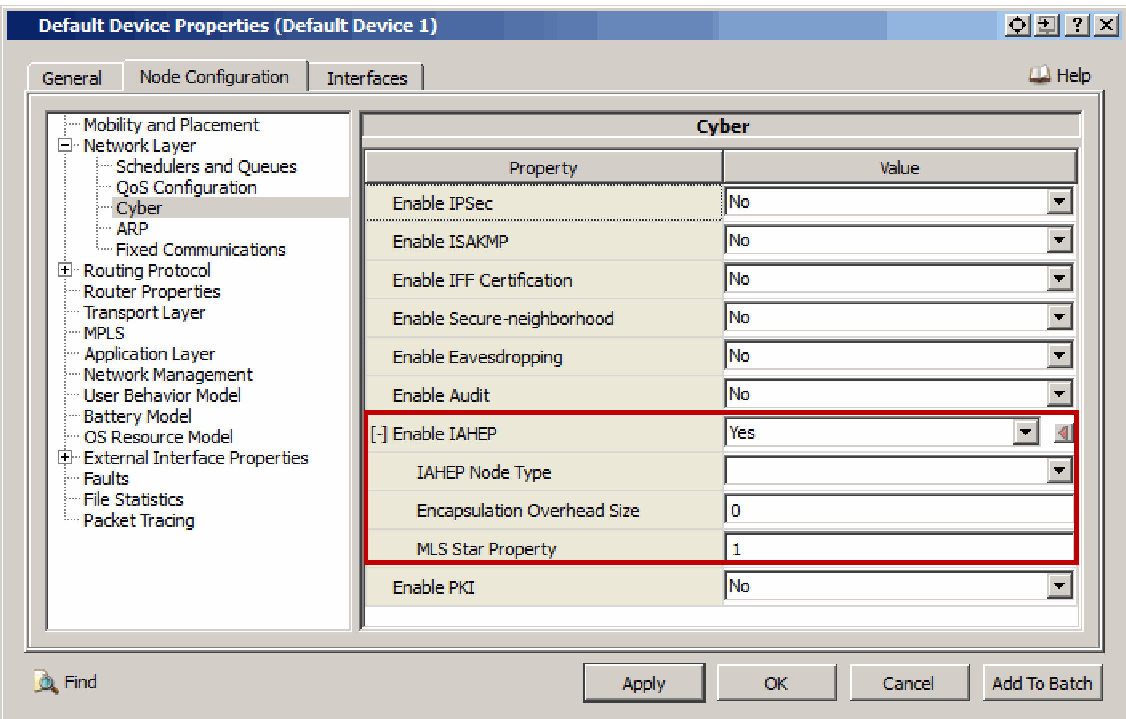


FIGURE 3-3. Setting IAHEP Node Parameters

TABLE 3-12. Command Line Equivalent of IAHEP Node Parameters

GUI Parameter	Scope of GUI Parameter	Command Line Parameter
IAHEP Node Type	Node	`IAHEP-NODE-TYPE`
Encapsulation Overhead Size	Node	`IAHEP-ENCAPSULATION-OVERHEAD-SIZE`
MLS Star Property	Node	`MLS-STAR-PROPERTY`

3. If IAHEP **Node Type** is set to *IAHEP*, then set the dependent parameters listed in Table 3-13.

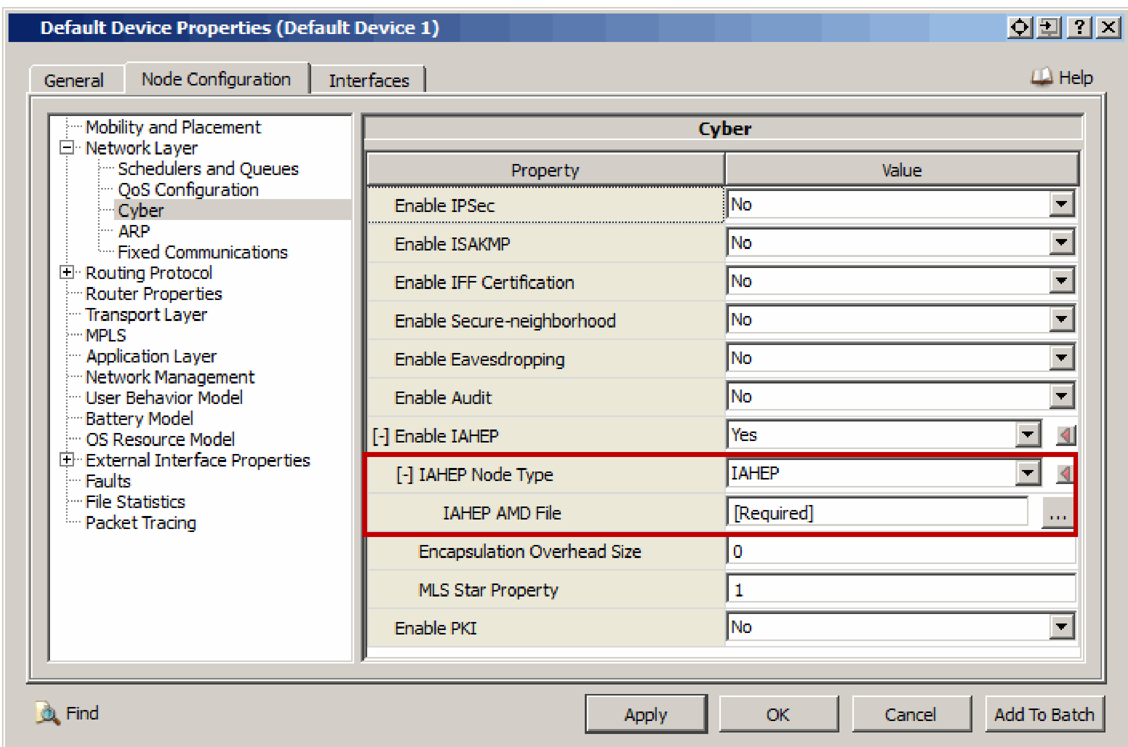


FIGURE 3-4. Specifying IAHEP AMD File

TABLE 3-13. Command Line Equivalent of IAHEP AMD File Parameters

GUI Parameter	Scope of GUI Parameter	Command Line Parameter
IAHEP AMD File	Node	`IAHEP-AMD-FILE`

Setting Parameters

- Set **IAHEP AMD File** to the name of the IAHEP AMD file. See Section 3.3.4.1 for the format of the IAHEP AMD file.

Configuring Interface Properties

To configure IAHEP properties for an interface, do the following:

1. Go to one of the following locations:

- **Interface Properties Editor > Interfaces > Interface # > Network Layer > Cyber**.
- **Default Device Properties Editor > Interfaces > Interface # > Network Layer > Cyber**.

In this section, we show how to configure IAHEP interface properties using the Interfaces tab of the Default Device Properties Editor. Parameters can be set in the other properties editors in a similar way.

2. Set the parameters listed in Table 3-14.

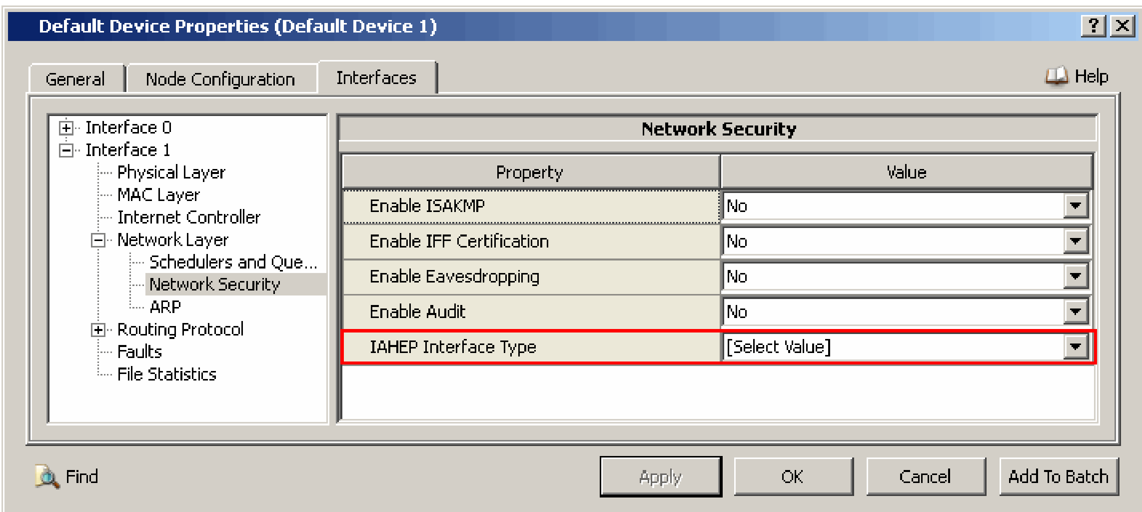


FIGURE 3-5. Setting IAHEP Interface Parameters

TABLE 3-14. Command Line Equivalent of IAHEP Interface Parameters

GUI Parameter	Scope of GUI Parameter	Command Line Parameter
IAHEP Interface Type	Interface	`IAHEP-INTERFACE-TYPE`

Setting Parameters

- Set **IAHEP Interface Type** to *Black* for the interface of a black node connecting to an IAHEP node and the interface of an IAHEP node connecting to a black node.
- Set **IAHEP Interface Type** to *Red* for the interface of a red node connecting to an IAHEP node and the interface of an IAHEP node connecting to a red node.

3.3.6 Statistics

The IAHEP model statistics that are output to the statistics (.stat) file at the end of simulation are listed in Table 3-15.

TABLE 3-15. IAHEP Statistics

Statistic	Description
MLS: Number Of Outgoing Unicast Packets Dropped Under STRONG PROPERTY	MLS: Number of outgoing unicast packets dropped under Strong security policy
MLS: Number Of Outgoing Unicast Packets Dropped Under LIBERAL PROPERTY	MLS: Number of outgoing unicast packets dropped under Liberal security policy
MLS: Number Of Incoming Unicast Packets Dropped Under SIMPLE PROPERTY	MLS: Number of incoming unicast packets dropped under Simple security policy
Number of IP Fragments Padded	Number of IP fragments padded
Number of Packets Received	Number of packets received
Number of Packets Sent	Number of packets sent
Number Of IGMP Report Messages Sent	Number of IGMP report messages sent
Number Of IGMP Leave Messages Sent	Number of IGMP leave messages sent

3.3.7 Scenarios Included in EXata

The EXata distribution includes several sample scenarios for the IAHEP model. All scenarios are located in the directory EXATA_HOME/scenarios/cyber/iahep. Table 3-16 lists the sub-directory where each scenario is located.

TABLE 3-16. IAHEP Model Scenarios

Scenario Sub-directory	Description
4nodes	Shows the functionality of a black node connected to multiple IAHEP nodes.
fragmentation	Shows the behavior of the IAHEP/IP node when an incoming IP packet size is less than or equal to `IAHEP-FRAGMENTATION-UNIT` and `IAHEP-FRAGMENTATION-UNIT` is less than `IP-FRAGMENTATION-UNIT`. After IAHEP processing, the packet size still remains less than or equal to `IAHEP-FRAGMENTATION-UNIT`.
ospfv2-2nodes	Show the functionality of IAHEP 2-node approach with OSPFv2 running on the entire network.

3.3.8 References

1. RFC 2408, “Internet Security Association and Key Management Protocol (ISAKMP).” D. Maughan, M. Schertler, M. Schneider, J. Turner. November 1998.
2. RFC 2407, “The Internet IP Security Domain of Interpretation for ISAKMP.” D. Piper. November 1998.
3. RFC 2412, “The OAKLEY Key Determination Protocol.” H. Orman. November 1998.
4. RFC 2401, “Security Architecture for the Internet Protocol.” S. Kent, R. Atkinson. November 1998.
5. RFC 2409, “The Internet Key Exchange (IKE).” D. Harkins and D. Carrel.

3.4 Internet Protocol Security (IPSec) Model

The EXata IPSec model is based on the RFC 2401, RFC 2403, RFC 2404, RFC 2405 and RFC 2406.

3.4.1 Description

IPSec is designed to provide cryptographically-based security for IPv4 and IPv6 that includes the following:

- Access Control
- Connectionless Integrity
- Data Origin Authentication
- Partial Sequential Integrity
- Confidentiality
- Traffic Flow Confidentiality

3.4.2 Features and Assumptions

This section describes the implemented features, omitted features, assumptions and limitations of the IPSec model.

3.4.2.1 Implemented Features

- TUNNEL mode and TRANSPORT mode
- Unicast packet transfer
- Fragmentation and reassembly of IPSec packets
- Encapsulation of IP and upper layer protocols
- IPSec service is provided using the Encapsulation Security Payload (ESP) protocol
- Delays for running encryption and authentication algorithms
- Encryption algorithms: DES-CBC, 3DES-CBC, AES-CBC, AES-CTR
- Authentication: HMAC-MD5-96, HMAC-SHA1-96

3.4.2.2 Omitted Features

- Authentication Header (AH)
- IPv6, Multicast, and NAT (Network Address Translation)
- Authentication and encryption algorithms other than the ones listed in Section 3.4.2.1
- Internet Key Exchange (IKE) algorithm
- IP payload compression
- Extended Sequence Number implementation

3.4.2.3 Assumptions and Limitations

- IPSec can only work with ESP (Encapsulating Security Protocol). Using NULL encryption in ESP can imitate the unsupported AH protocol.
- For certificate Authority (CA), the IPSec models uses a configuration file.
- Only basic SA between the transmitter and receiver is established.
- Only delays for running the authentication and encryption algorithms are considered.

3.4.3 Command Line Configuration

To enable IPSec, include the following parameter in the scenario configuration (.config) file:

```
[<Qualifier>] IPSEC-ENABLED       YES
```

The scope of this parameter declaration can be Global or Node. See Section 1.2.1.1 for a description of `<Qualifier>` for each scope.

Note: The default value of `IPSEC-ENABLED` is `NO`.

IPSec Parameters

Table 3-17 lists the IPSec parameters specified in the scenario configuration (.config) file. See Section 1.2.1.3 for a description of the format used for the parameter table.

TABLE 3-17. IPSec Parameters

Parameter	Value	Description
`IPSEC-CONFIG-FILE` *Required* *Scope:* Global, Node	Filename	Specifies the name of the IPSec configuration file. This file usually has the extension ".ipsec". The format of the IPSec configuration file is described in Section 3.4.3.1
`IPSEC-HMAC-MD5-PROCESSING-RATE` *Optional* *Scope:* Global, Node	Real *Range:* ≥ `0.0` *Default:* `800000000.0` *Unit:* bps	Processing rate for the 'HMAC-MD5' authentication algorithm.
`IPSEC-HMAC-SHA-1-PROCESSING-RATE` *Optional* *Scope:* Global, Node	Real *Range:* ≥ `0.0` *Default:* `800000000.0` *Unit:* bps	Processing rate for the 'HMAC-SHA-1' authentication algorithm.
`IPSEC-HMAC-MD5-96-PROCESSING-RATE` *Optional* *Scope:* Global, Node	Real *Range:* ≥ `0.0` *Default:* `800000000.0` *Unit:* bps	Processing rate for the 'HMAC-MD5-96' authentication algorithm.
`IPSEC-HMAC-SHA-1-96-PROCESSING-RATE` *Optional* *Scope:* Global, Node	Real *Range:* ≥ `0.0` *Default:* `800000000.0` *Unit:* bps	Processing rate for the 'HMAC-SHA-1-96' authentication algorithm

TABLE 3-17. IPSec Parameters (Continued)

Parameter	Value	Description
`IPSEC-DES-CBC-PROCESSING-RATE` *Optional* *Scope:* Global, Node	Real *Range:* ≥ `0.0` *Default:* `800000000.0` *Unit:* bps	Processing rate for the 'DES-CBC ' encryption algorithm.
`IPSEC-REAL-CRYPTO-ENABLED` *Optional* *Scope:* Global, Node	List: • `YES` • `NO` *Default:* `NO`	Specifies whether the payload is actually encrypted, decrypted, and authenticated. If this is disabled, a processing delay is introduced without actually encrypting/decrypting the payload. If this is enabled, the payload is actually encrypted, decrypted, and authenticated, in addition to adding the processing delay. In emulation mode, this must be enabled for communication with an external node.

3.4.3.1 Format for the IPSec Configuration File

An IPSec Configuration file specifies the IPSec configuration information (interface to be configured as IPSec, Security Association Database entry and Security Policy Database entry) for each IPSec enabled node.

For each IPSec enabled node, the IPSec Configuration file contains configuration information in the following format:

```
<Interface Section>
<SPD Section>
<SAD Section>
```

These sections are described below.

Interface Section

The Interface Section has the following format:

```
NODE <node ID> <interface index>
```

where

`<node ID>`	ID of the node.
`<interface index>`	Index of the IPSec enabled interface of the node.

SAD Section

An SAD (Security Association Database) section defines the parameters associated with each SA (Security Association). An SA is simply the bundle of algorithms and parameters (such as keys) that is used to encrypt and authenticate a particular flow in one direction.

The SAD section has one or more entries in the following format:

```
SA <name-tag> <mode> <dest> <security protocol> <spi>
[-E <encryption algorithm> <encryption key>]
[-A <authentication algorithm> <authentication key>]
```

where

`<name-tag>`	Name of the Security Association Entry.
`<mode>`	IPSec operation mode. It can take two possible values: `TUNNEL` or `TRANSPORT`.
`<dest>`	IPSec SA end point. It specifies the address of an interface of the other end of the IPSec SA. For `TRANSPORT` mode, this should be the address of the destination host. For `TUNNEL` mode, this should be the interface address of the end node of the IPSec tunnel.
`<security protocol>`	Currently EXata supports only `ESP` (Encapsulating Security Payload) mode.
`<spi>`	SPI (Security Parameter Index) is used to uniquely identify an SA.
`<encryption algorithm>`	Encryption algorithm for the SA entry. It can take one of the following values: `DES-CBC`, `3DES-CBC`, `AES-CBC`, or `AES-CTR`. **Note:** Specification of the encryption algorithm and the associated key is optional.
`<encryption key>`	Encryption key.
`<authentication algorithm>`	Authentication algorithm for the SA entry. It can take any of the following values: `HMAC-MD5-96` or `HMAC-SHA-1-96`. **Note:** Specification of the authentication algorithm and the associated key is optional.
`<authentication key>`	Authentication key.

Example

The following is an example of an SAD entry:

```
SA sa1 TUNNEL 6.2 ESP 12345
-E DES-CBC 0x12345678
-A HMAC-MD5-96 0x1234567890123456
```

SPD Section

A SPD (Security Policy Database) entry specifies what services are to be offered to IP datagrams and in what fashion. It contains an ordered list of policy entries and must be consulted during the processing of all traffic (inbound or outbound), including non-IPsec traffic.

The SPD section has one or more entries in the following format (all parameters should be entered on the same line):

```
SP <source range> <destination range> <upper layer protocol> -P
   <direction> <policy>
```

where

`<source range>`	Specifies the network address from where packets (to be processed by IPSec) originate. The format is: `Nx-y.0[<port-number>]`, where `<port-number>` is the source port number. Refer to *EXata User's Guide* for a description of the EXata "N" notation for network addresses. Specification of the port number is optional and if it is included, it is enclosed in square brackets.
`<destination range>`	Specifies the destination network address of the packets processed by IPSec. The format is: `Nx-y.0[port number]`, where `<port-number>` is the destination port number.
`<upper layer protocol>`	Specifies the transport protocol. It may be `TCP`, `UDP` or `ANY`. Packets delivered by these protocols will be processed by IPSec.
`<direction>`	Specifies the traffic mode (inbound or outbound) for IPSec processing through the IPSec enabled interface. It can either be `IN` for inbound packets or `OUT` for outbound packets.
`<policy>`	Policy to be applied to the packets. It can be one of the following: • `DISCARD` • `NONE` • `IPSEC <SA name>`, where `<SA name>` is the name of an SAD entry.

Example

The following is an example of an SPD entry:

```
SP N2-1.0[4001] N2-6.0[5001] TCP -P IN IPSEC sa1
```

Examples of IPSec Configuration File

Example 1: The following is an example of an IPSec Configuration file for tunnel mode:

```
NODE 2 1
SA sa1 TUNNEL 3.2 ESP 12346
-E DES-CBC "auth-key"
-A HMAC-MD5-96 "encryp_key"

SP N2-1.0 N2-7.0 UDP -P OUT IPSEC sa1

NODE 4 1
SA sa1 TUNNEL 3.2 ESP 12346
-E DES-CBC "auth-key"
-A HMAC-MD5-96 "encryp_key"

SP N2-2.0 N2-7.0 UDP -P OUT IPSEC sa1
```

Example 2: The following is an example of an IPSec Configuration file for transport mode:

```
NODE 1 0
SA sa1 TRANSPORT 5.2 ESP 12345
-E DES-CBC CCCCCCCC
-A HMAC-MD5-96 "encryp_key"

SA sa2 TRANSPORT 1.1 ESP 12346
-E DES-CBC CCCCCCCC
-A HMAC-SHA-1-96 "encryp_key"

SP N2-1.0 N2-5.0 UDP -P OUT IPSEC sa1

SP N2-5.0 N2-1.0 UDP -P IN IPSEC sa2


NODE 6 0
SA sa1 TRANSPORT 5.2 ESP 12345
-E DES-CBC CCCCCCCC
-A HMAC-MD5-96 "encryp_key"

SA sa2 TRANSPORT 1.1 ESP 12346
-E DES-CBC CCCCCCCC
-A HMAC-SHA-1-96 "encryp_key"

SP N2-1.0 N2-5.0 UDP -P IN IPSEC sa1

SP N2-5.0 N2-1.0 UDP -P OUT IPSEC sa2
```

3.4.4 GUI Configuration

This section describes how to configure IPSec in the GUI.

Configuring IPSec Parameters

To configure IPSec for a particular node, perform the following steps:

1. Go to **Default Device Properties Editor > Node Configuration > Network Layer > Cyber**.
2. Set **Enable IPSec** to *Yes,* and set the dependant parameters listed in Table 3-18.

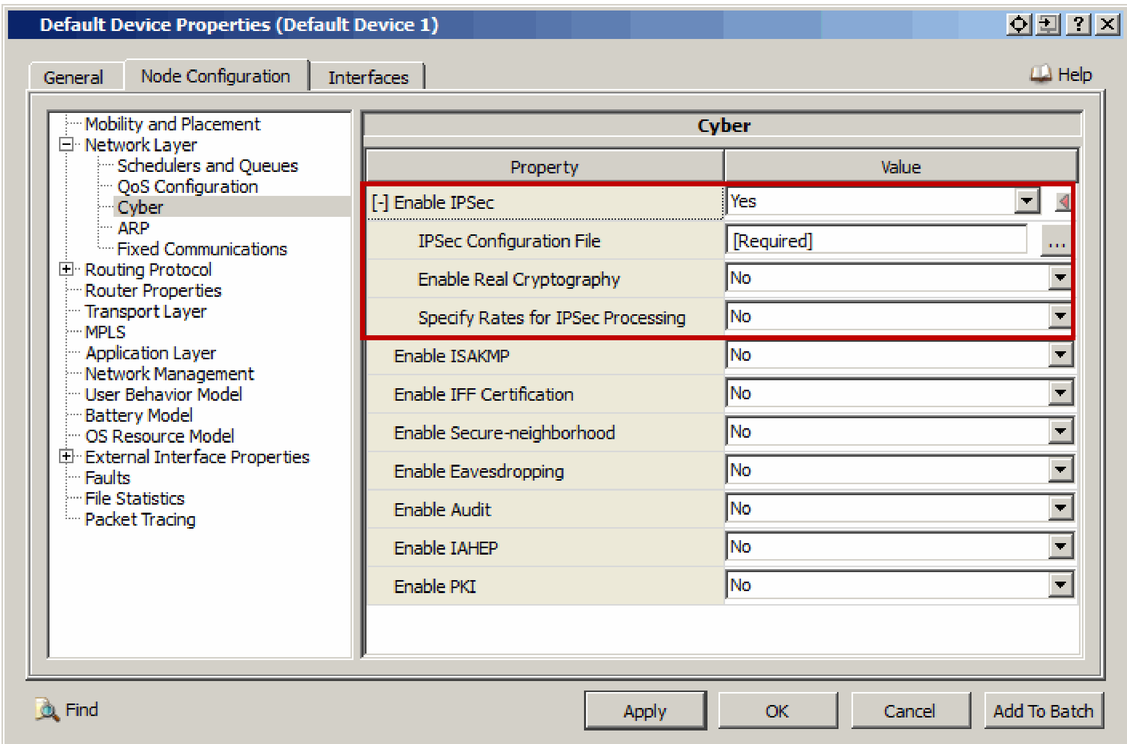


FIGURE 3-6. Configuring IPSec Parameters

TABLE 3-18. Command Line Equivalent of IPSec Parameters

GUI Parameter	Scope of GUI Parameter	Command Line Parameter
IPSec Configuration File	Node	`IPSEC-CONFIG-FILE`
Enable Real Cryptography	Node	`IPSEC-REAL-CRYPTO-ENABLED`
Specify Rates for IPSec Processing	Node	N/A

Setting Parameters

- Set **IPSec Configuration File** to the name of the IPSec configuration file. The format of the IPSec configuration file is described in Section 3.4.3.1.
- To specify rates for IPSec processing, set **Specify rates for IPSec Processing** to *Yes*; otherwise, set **Specify rates for IPSec Processing** to *No*.

3. If **Specify Rates for IPSec Processing** is set to *Yes*, then set the dependent parameters listed in Table 3-19.

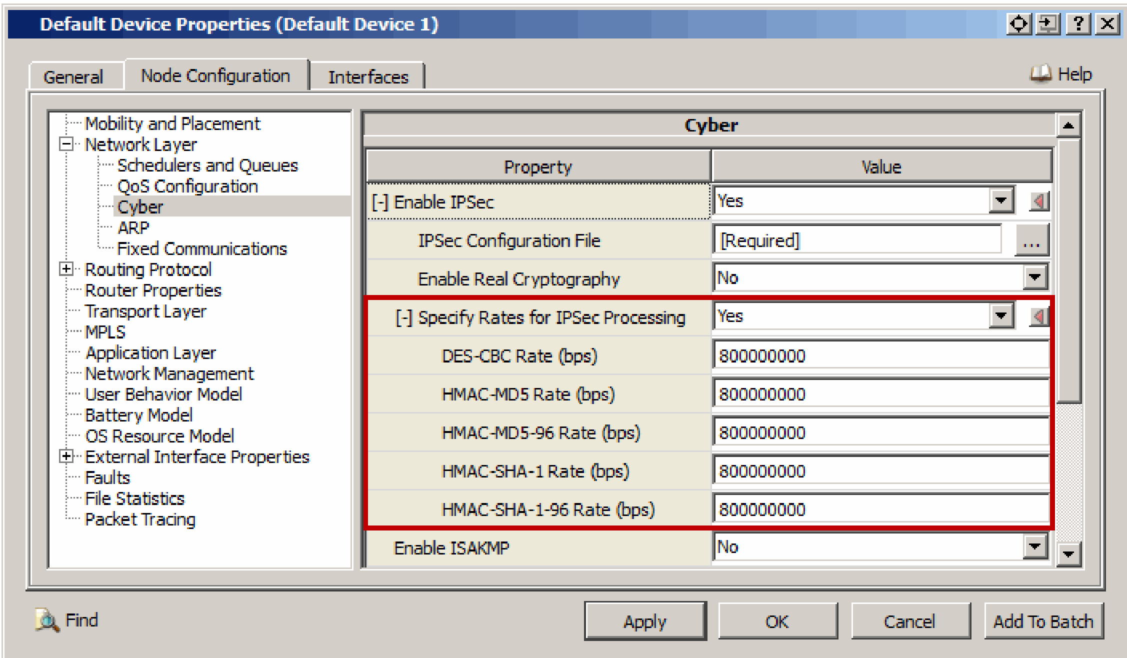


FIGURE 3-7. Specifying IPSec Rate Parameters

TABLE 3-19. Command Line Equivalent of IPSec Rate Parameters

GUI Parameter	Scope of GUI Parameter	Command Line Parameter
DES-CBC Rate	Node	`IPSEC-DES-CBC-PROCESSING-RATE`
HMAC-MD5 Rate	Node	`IPSEC-HMAC-MD5-PROCESSING-RATE`
HMAC-MD5-96 Rate	Node	`IPSEC-HMAC-MD5-96-PROCESSING-RATE`
HMAC-SHA-1 Rate	Node	`IPSEC-HMAC-SHA-1-PROCESSING-RATE`
HMAC-SHA-1-96 Rate	Node	`IPSEC-HMAC-SHA-1-96-PROCESSING-RATE`

Configuring Statistics Parameters

Statistics for IPSec can be collected at the global and node levels. See Section 4.2.9 of *EXata User's Guide* for details of configuring statistics parameters.

To enable statistics collection for IPSec, check the box labeled *Network* in the appropriate properties editor.

TABLE 3-20. Command Line Equivalent of Statistics Parameters

GUI Parameter	Scope of GUI Parameter	Command Line Parameter
Network	Global, Node	`NETWORK-LAYER-STATISTICS`

3.4.5 Statistics

Table 3-21 lists the IPSec statistics that are output to the statistics (.stat) file at the end of simulation.

TABLE 3-21. IPSec Statistics

Statistic	Description
Packet Processed	Number of IPSec packets processed at IPSec enabled inbound interface.
Packet Dropped	Number of IPSec packets dropped at IPSec enabled inbound interface.
Total Delay Overhead	Total delay for inbound IPSec packet processing.
Packet Processed	Number of IPSec packets processed at IPSec enabled outbound interface.
Packet Dropped	Number of IPSec packets dropped at IPSec enabled outbound interface.
Total Byte Overhead	Total overhead bytes for IPsec processing.
Total Delay Overhead	Total delay for outbound IPSec packet processing

3.4.6 Scenarios Included in EXata

The EXata distribution includes several sample scenarios for the IPSec model. All scenarios are located in the directories EXATA_HOME/scenarios/cyber/ipsec/transport-mode and EXATA_HOME/scenarios/cyber/ipsec/tunnel-mode. Table 3-22 and Table 3-23 list the sub-directory where each scenario is located.

TABLE 3-22. IPSec Transport Scenarios Included in EXata

Scenario	Description
BiDirTransportLargePacketAHAHandESP	Shows the behavior of bi-directional IPSec transport mode between a pair of IPSec-enabled nodes (using 2 SPD entries for each node) when one ESP and two AH protocols are used.
BiDirTransportSmallPacketAHandESP	Shows the behavior of bi-directional IPSec transport mode between a pair of IPSec-enabled nodes (using 2 SPD entries for each node) when both AH and ESP protocols are used.
BiDirTransportSmallPacketESPonly	Shows the behavior of bi-directional IPSec transport mode between a pair of IPSec-enabled nodes (using 2 SPD entries for each node) when only ESP protocol is used.
ConfigOptions	Show how different parameters can be configured. ESP protocol can use both -E and -A options when used in combination with the AH protocol. `SPI`: It can be a decimal or a hexadecimal number. `<Src Addr>` and `<Dest Addr>` in SP entry: They can take wild card values. `<"Encryption_key">`: It can be specified with or without quotes. In the latter case, the key should be a hex string beginning with “0x”. `<"Auth_key">`: It can be specified with or without quotes. In the latter case, the key should be a hex string beginning with “0x”.
MultireceiverTransportLargePacket	Shows multiple applications going through one IPSec (Transport mode) host to multiple other IPSec (Transport mode) hosts.
MultiSenderTransportLargePacketAHandESP	Shows that multiple IPSec enabled node can have multiple tunnels (Security Association) entry for a single SPD (Security Policy Database) entry when both AH and ESP protocols are used.

TABLE 3-22. IPSec Transport Scenarios Included in EXata (Continued)

Scenario	Description
SPPortNumberTest	Shows the trace of Super Application in TCP mode.
UniDirAHoverMulticastTraffic	Show the operation of PIM-DM in a string topology network.

.

TABLE 3-23. IPSec Tunnel Scenarios Included in EXata

Scenario	Description
BiDirTunnelLargePacket	Shows the behavior of bi-directional IPSec tunnels between a pair of IPSec-enabled nodes (using 2 SPD entries for each node) when only AH protocol is used.
BiDirTunnelSmallPacketAHandESP	Shows the behavior of bi-directional IPSec tunnels between a pair of IPSec-enabled nodes (using 2 SPD entries for each node) when both AH and ESP protocols are used.
MultiReceiverTunnelSmallPacket	Shows the behavior of multiple applications from one IPSec gateway to multiple IPSec gateways.
MultiSenderTunnelSmallPacket	Demonstrates multiple applications from multiple IPSec gateways to one IPSec gateway.
NestedTunnel	Demonstrates the IPSec functionality testing for nested tunneling.

3.4.7 References

1. [RFC4303] IP Encapsulating Security Payload.
2. [RFC4835] Cryptographic Algorithm Implementation Requirements for Encapsulating Security Payload (ESP) and Authentication Header (AH).
3. [RFC4301] Security Architecture for the Internet Protocol
4. [RFC1829] The ESP DES-CBC Transform
5. [RFC2403] The use of HMAC-MD5-96 within ESP and AH.
6. [RFC1321] The MD5 Message Digest Algorithm.
7. [RFC1851] The ESP Triple DES Transform
8. Software Requirement Specification (SRS) for IPSec

3.5 Internet Security Association and Key Management Protocol with Internet Key Exchange (ISAKMP-IKE)

The EXata Internet Security Association and Key Management Protocol with Internet Key Exchange (ISAKMP-IKE) model is based on RFC 2407, RFC 2408, and RFC 2409.

3.5.1 Description

Internet Security Association and Key Management Protocol (ISAKMP) provides a general framework to other security protocols for creating and maintaining Security Associations (SAs) in an Internet environment. The ISAKMP host negotiates SAs (ISAKMP SA) with other ISAKMP hosts and other security protocol and services use these ISAKMP SA to create their own SAs.

The SA feature coupled with authentication and key establishment allows users to choose their own security service, key exchange technique, encryption algorithm, and authentication mechanism based on their requirement with other users. For this, ISAKMP defines the general format and various payloads.

Internet Key Exchange (IKE) is a hybrid protocol to obtain authenticated keying material for use with ISAKMP and for other security associations, such as Authentication Header (AH) and Encapsulating Security Payload (ESP) for the IPsec Domain Of Interpretation (DOI).

3.5.2 Features and Assumptions

This section describes the implemented features, omitted features, assumptions and limitations of the ISAKMP model.

3.5.2.1 Implemented Features

- Payload and other header formats defined in RFC 2408 and RFC 2409.
- Exchange types defined in section 4.4 to 4.8 of RFC 2408.
- Exchange types defined in section 5.1 to 5.7 of RFC 2409.
- Processing of all payloads excepts that are mentioned in omitted section as defined in section 5 of RFC 2408.

3.5.2.2 Omitted Features

- Implementation of actual payloads, such as certificate, certificate request, and hash payloads.
- Re-establishment of Phase-1 SA.
- Implementation of Security Policy SIT_SECRECY and SIT_INTEGRITY type.
- Multiple SA negotiation.
- Cryptography related to OAKLEY in RFC 2409.

3.5.2.3 Assumptions and Limitations

- ISAKMP is implemented as a demon process in the actual world. Phase 1 is started after the initialization phase with user specified delay and phase 2 is started after phase 1 is completed.
- It is also possible to start phase 2 when some data packet comes at ISAKMP server and it doesn’t found any IPSec SA for that packet's source and destination networks.
- Algorithms for creating cookies, generating keys and nonce is been simulated by some simple stub functions.
- Established SAs are bi-directional, that is same SA is used for both inbound and outbound packets.

- Only one proposal is been sent during phase-1 establishment, however multiple transforms can be sent in a single proposal.
- IKE New Group mode is considered as a part of phase 1 only. After the ISAKMP SA establishment, New Group mode (if enabled) will start as Next Phase for phase 1.
- If certificate type exists, then certificate data payload is assumed to be size 1024 which will send as virtual payload in message.
- In public key exchange, it is assumed that the initiator is already having the responder's public key. Similar assumption is applied for pre-shared key.

3.5.3 Supplemental Information

None.

3.5.4 Command Line Configuration

To enable ISAKMP-IKE model, include the following parameter in the scenario configuration (.config) file:

```
[<Qualifier>]  ISAKMP-SERVER    YES
```

The scope of this parameter declaration can be Global, Node, Subnet, or Interface. See Section 1.2.1.1 for a description of `<Qualifier>` for each scope.

Note: The default value of this parameter is `NO`.

ISAKMP-IKE Parameters

Table 3-24 lists the ISAKMP-IKE parameters specified in the scenario configuration (.config) file. See Section 1.2.1.3 for a description of the format used for the parameter table.

TABLE 3-24. ISAKMP-IKE Parameters

Parameter	Value	Description
`ISAKMP-CONFIG-FILE` *Required* *Scope:* All	Filename	Specifies the name of the ISAKMP configuration file. This file usually has the extension ".isakmp" and is used to configure the ISAKMP parameters. The format of the ISAKMP configuration file is described in Section 3.5.4.1.
`ISAKMP-PHASE-1-START-TIME` *Optional* *Scope:* All	Time *Range:* ≥ `0S` *Default:* `30S`	Specifies the time after the initialization, when Phase 1 negotiation starts.
`ISAKMP-ENABLE-IPSEC` *Optional* *Scope:* All	List: • `YES` • `NO` *Default:* `NO`	Specifies whether the IPSec-SA negotiated by ISAKMP will be used as the parameter of IPSec-ESP.

3.5.4.1 Format of the ISAKMP Configuration File

ISAKMP parameters can be configured in the ISAKMP configuration (.isakmp) file for a node interface using the following steps.

Note: All the parameters are mandatory and need to be specified in the given order.

1. Specify all the peer servers with whom this interface will negotiate the ISAKMP exchanges, using the following syntax.

```
NODE <node-interface-ipv4-address>
PEER <peer-ipv4-address-1> <node-peer-configuration-1>
...
PEER <peer-ipv4-address-n> <node-peer-configuration-n>
```

where

`<node-interface-ipv4-address>`	Interface address of the node initiating the ISAKMP exchange. The wildcard character "*" can be used to represent any interface address.
`<peer-ipv4-address-i>`	Interface address of the *i*th peer server with whom this interface will negotiate the ISAKMP exchange. The wildcard character "*" can be used to represent any interface address.
`<node-peer-configuration-i>`	String identifier for the *i*th peer server configuration. Configuration parameters for each peer are grouped together. This identifier is used to associate a peer with its configuration parameters.

Example 1: Interface 192.168.3.1 of node 3 has interface 192.168.3.2 of node 4 as its peer server.

```
NODE 192.168.3.1
PEER 192.168.3.2 3-4-Config
```

Example 2: Interface 192.168.3.1 of node 3 has interface 192.168.3.2 of node 4 and interface 192.168.3.3 of node 5 as its peer servers.

```
NODE 192.168.3.1
PEER 192.168.3.2 3-4-Config
PEER 192.168.3.3 3-5-Config
```

2. Specify phase 1 and phase 2 configuration parameters using the following syntax:

```
PHASE 1
DOI                        <domain-of-interpretation>
EXCHANGE_TYPE              <exchange-type-1>
FLAGS                      <flags>
CERTIFICATE_ENABLED        <use-certificate>
NEW_GROUP_NODE_ENABLED     <use-new-group>
TRANSFORMS                 <transform-name-1> ... <transform-name-n>

PHASE 2
LOCAL-ID-TYPE              <local-id-type>
LOCAL-NETWORK              <local-network-address>
LOCAL-NETMASK              <local-network-mask>
REMOTE-ID-TYPE             <remote-id-type>
REMOTE-NETWORK             <remote-network-address>
REMOTE-NETMASK             <remote-network-mask>
UPPER-LAYER-PROTOCOL       <upper-layer-protocol>
DOI                        <domain-of-interpretation>
EXCHANGE_TYPE              <exchange-type-2>
FLAGS                      <flags>
PROPOSALS                  <proposal-name-1> ... <proposal-name-n>
```

where

`<domain-of-interpretation>`	Name of domain of interpretation. This should be set to `ISAKMP_DOI`.
`<exchange-type-1>`	ISAKMP exchange type for phase 1. The exchange type can be one of the following:

Exchange Type	Description
EXCH_BASE	ISAKMP Base Exchange
EXCH_IDENT	ISAKMP Identity Protection Exchange
EXCH_AUTH	ISAKMP Authentication Only Exchange
EXCH_INFO	ISAKMP Information Exchange
EXCH_AGGR	ISAKMP Aggressive Exchange
EXCH_MAIN_PRE_SHARED	ISAKMP IKE Main Pre-Shared Key Exchange
EXCH_MAIN_DIFG_SIG	ISAKMP IKE Main Digital Signature Exchange
EXCH_MAIN_PUB_KEY	ISAKMP IKE Main Public Key Exchange
EXCH_MAIN_REV_PUB_KEY	ISAKMP IKE Main Revised Public Key Exchange
EXCH_AGG_PRE_SHARED	ISAKMP IKE Aggressive Pre-Shared Key Exchange
EXCH_AGG_DIFG_SIG	ISAKMP IKE Aggressive Digital Signature Exchange
EXCH_AGG_PUB_KEY	ISAKMP IKE Aggressive Public Key Exchange
EXCH_AGG_REV_PUB_KEY	ISAKMP IKE Aggressive Revised Public Key Exchange

`<flags>`	Flag bits described in RFC2408. This field can have the following values: `ACE`, `CE`, `AE`, `AC`, `A`, `C`, `E`, `NONE`

where

A	Auth-only bit
C	Commit bit
E	Encryption bit
NONE	No flag bit is set

`<use-certificate>`	Indicates whether the certificate feature is enabled. This can be `YES` or `NO`. **Note:** This field is meaningful only if one of the transforms is `KEY_IKE`. **Note:** This parameter is optional. By default, the certificate feature is disabled.
`<use-new-group>`	Indicates whether the new group mode is enabled. This can be `YES` or `NO`. **Note:** This field is meaningful only if one of the transforms is `KEY_IKE`. **Note:** This field is optional. By default, the new group mode is disabled.
`<transform-name-`*i*`>`	User-defined name of the i^{th} ISAKMP transform. **Note:** At least one transform name should be specified.
`<local-id-type>`	Local network type. This can be `IPV4_ADDR_SUBNET` (indicating a subnet IP address) or `IP_ADDR` (indicating a host IP address).
`<local-network-address>`	Local network address.
`<local-network-mask>`	Local network mask.
`<remote-id-type>`	Remote network type. This can be `IPV4_ADDR_SUBNET` (indicating a subnet IP address) or `IP_ADDR` (indicating a host IP address).
`<remote-network-address>`	Remote network address.
`<remote-network-mask>`	Remote network mask.
`<upper-layer-protocol>`	Upper layer protocol. This can be `TCP` or `UDP`.
`<exchange-type-2>`	ISAKMP exchange type for phase 2. The exchange type can be one of the following:

Exchange Type	Description
`EXCH_BASE`	ISAKMP Base Exchange
`EXCH_IDENT`	ISAKMP Identity Protection Exchange
`EXCH_AUTH`	ISAKMP Authentication Only Exchange
`EXCH_INFO`	ISAKMP Information Exchange
`EXCH_AGGR`	ISAKMP Aggressive Exchange
`EXCH_QUICK`	ISAKMP IKE QUICK Exchange

`<proposal-name-i>`	User-defined name of the *i*th ISAKMP proposal. **Note:** At least one proposal name should be specified.

Example:

```
3-4-Config

PHASE 1
DOI ISAKMP_DOI
EXCHANGE_TYPE EXCH_MAIN_DIG_SIG
FLAGS ACE
CERTIFICATE_ENABLED  YES
NEW_GROUP_MODE_ENABLED  YES
TRANSFORMS 3DES-SHA

PHASE 2
LOCAL-ID-TYPE IPV4_ADDR_SUBNET
LOCAL-NETWORK 192.168.5.0

LOCAL-NETMASK 255.255.255.0
REMOTE-ID-TYPE IPV4_ADDR_SUBNET
REMOTE-NETWORK 192.168.1.0
REMOTE-NETMASK 255.255.255.0
UPPER-LAYER-PROTOCOL UDP
DOI IPSEC_DOI
EXCHANGE_TYPE EXCH_IDENT
FLAGS ACE
PROPOSALS ESP-DES-MD5-PFS AH-MD5-PFS
```

3. For each proposal used in the phase 2 configuration, specify the protocols using the following syntax:

```
<proposal-name>
PROTOCOLS <protocol-name-1> ... <protocol-name-n>
```

where

`<proposal-name>`	Name of the ISAKMP proposal. This should be the same as the name used in the phase 2 configuration.
`<protocol-name-i>`	User-defined name of the *i*th protocol. **Note:** At least one protocol name should be specified.

Example 1:

```
ESP-DES-MD5-PFS
PROTOCOLS ESP-DES-MD5
```

Example 2:

```
ESP-DES-MD5-PFS
PROTOCOLS ESP-DES-MD5
AH-MD5-PFS
PROTOCOLS AH-MD5
```

4. For each protocol used in a proposal, specify the protocol configuration using the following syntax:

```
<protocol-name>
PROTOCOL_ID <protocol-ID>
TRANSFORMS <transform-name-1> ... <transform-name-n>
```

where

`<protocol-name>`	Name of the protocol. This should be the same as the name used in the proposal specification.
`<protocol-ID>`	ID of the protocol. It can be one of the following: `AH` or `ESP`.
`<transform-name-i>`	User-defined name of the i^{th} transform.

Example 1:

```
ESP-DES-MD5
PROTOCOL_ID ESP
TRANSFORMS ESP-DES-MD5-PFS-XF
```

Example 2:

```
AH-MD5
PROTOCOL_ID AH
TRANSFORMS AH-MD5-PFS-XF
```

5. Specify phase 1 transforms used using the following syntax:

```
Phase-1-Transforms
<transform-specification-1>
...
<transform-specification-n>
```

Each phase 1 transform specification has the following format:

```
TRANSFORM_NAME          <transform-name>
TRANSFORM_ID            <transform-ID>
ENCRYPTION_ALGORITHM    <encryption-algorithm>
HASH_ALGORITHM          <hash-algorithm>
CERTIFICATION_TYPE      <certification-type>
AUTHENTICATION_METHOD   <auth-algorithm>
GROUP_DESCRIPTION       <group-description>
PROCESSING_DELAY        <processing-delay>
LIFE                    <life-time>
```

where

`<transform-name>`	Name of the transform. This should be the same as the name used in the phase 1 configuration.
`<transform-ID>`	Transform ID. It can be one of the following: `KEY_IKE`, `AH_MD5`, `AH_SHA`, `AH_DES`, `ESP_DES_IV64`, `ESP_DES`, `ESP_3DES`, `ESP_RC5`, `ESP_CAST`, `ESP_BLOWFISH`, `ESP_3IDEA`, `ESP_DES_IV32`, `ESP_RC4`, or `ESP_NULL`.

<encryption-algorithm>	Name of the encryption algorithm. It can be DEFAULT or one of the following: DES-CBC, 3DES-CBC, SIMPLE, BLOWFISH-CBC, or NULL.
<hash-algorithm>	Name of the hash algorithm. It can be DEFAULT or one of the following: MD5 or SHA.
<certification-type>	Name of the certificate type. It can be DEFAULT or one of the following:

Certificate Type	Description
NONE	Certificate Type NONE
PKCS7	PKCS #7 wrapped X,509 certificate
PGP	PGP Certificate
DNS_SIGNED	DNS Signed Key
X509_SIG	X.509 Certificate - Signature
X509_KEYEX	X.509 Certificate - KEY Exchange
KERBEROS	Kerberos Tokens
CRL	Certificate Revocation List
ARL	Authority Revocation List
SPKI	SPKI Certificate
X509_ATTRI	X.509 Certificate - Attribute

<auth-algorithm>	Name of the authentication method. It can be DEFAULT or one of the following: RSA_SIG or PRE_SHARED.
<group-description>	Name of the group description. It can be DEFAULT or one of the following: MODP_768 or MODP_1024.
<processing-delay>	Encryption delay. It can be one of the encryption delay (in EXata time format) or DEFAULT. **Note:** DEFAULT is equal to 10US.
<life-time>	Lifetime of this transform, in minutes.

Example 1:

```
TRANSFORM_NAME 3DES-SHA
TRANSFORM_ID KEY_IKE
ENCRYPTION_ALGORITHM 3DES-CBC
HASH_ALGORITHM SHA
CERTIFICATION_TYPE PKCS7
AUTHENTICATION_METHOD RSA_SIG
GROUP_DESCRIPTION MODP_1024
PROCESSING_DELAY 10US
LIFE 10
```

Example 2:

```
TRANSFORM_NAME DES-MD5
TRANSFORM_ID KEY_IKE
ENCRYPTION_ALGORITHM DES-CBC
HASH_ALGORITHM MD5
CERTIFICATION_TYPE PKCS7
AUTHENTICATION_METHOD PRE_SHARED
GROUP_DESCRIPTION MODP_1024
LIFE 7
```

6. Specify phase 2 transforms using the following syntax:

```
Phase-2-Transforms
<transform-specification-1>
...
<transform-specification-n>
```

Each phase 2 transform specification has the following format:

```
TRANSFORM_NAME              <transform-name>
TRANSFORM_ID                <transform-ID>
ENCAPSULATION_MODE          <encapsulation-mode>
GROUP_DESCRIPTION           <group-description>
AUTHENTICATION_ALGORITHM    <auth-algorithm>
LIFE                        <life-time>
```

where

`<transform-name>`	Name of the transform specified in the phase 2 protocol configuration.
`<transform-ID>`	Transform ID. It can be one of the following: `KEY_IKE`, `AH_MD5`, `AH_SHA`, `AH_DES`, `ESP_DES_IV64`, `ESP_DES`, `ESP_3DES`, `ESP_RC5`, `ESP_CAST`, `ESP_BLOWFISH`, `ESP_3IDEA`, `ESP_DES_IV32`, `ESP_RC4`, or `ESP_NULL`.
`<encapsulation-mode>`	Encapsulation mode to use. It can be `DEFAULT` or one of the following: `TUNNEL` or `TRANSPORT`.
`<group-description>`	Name of the group description. It can be `DEFAULT` or one of the following: `MODP_768` or `MODP_1024`.
`<auth-algorithm>`	Name of the authentication method. It can be `DEFAULT` or one of the following: `RSA_SIG` or `PRE_SHARED`.
`<life-time>`	Lifetime of this transform, in minutes.

Example 1:

```
TRANSFORM_NAME ESP-DES-MD5-PFS-XF
TRANSFORM_ID ESP_DES
ENCAPSULATION_MODE TUNNEL
GROUP_DESCRIPTION MODP_1024
AUTHENTICATION_ALGORITHM HMAC-MD5
LIFE 10
```

Example 2:

```
TRANSFORM_NAME AH-MD5-PFS-XF
TRANSFORM_ID AH_MD5
ENCAPSULATION_MODE TUNNEL
GROUP_DESCRIPTION MODP_1024
AUTHENTICATION_ALGORITHM HMAC-MD5
LIFE 15
```

3.5.5 GUI Configuration

To configure ISAKMP-IKE in the GUI, perform the following steps:

1. Go to one of the following locations:

- To set properties at the node level, go to **Default Device Properties Editor > Node Configuration > Network Layer > Cyber**.
- To set properties at the subnet level, go to **Wireless Subnet Properties Editor > Network Layer > Cyber**.
- To set properties at the interface level, go to one of the following locations:
  - **Interface Properties Editor > Interfaces > Interface # > Network Layer > Cyber**
  - **Default Device Properties Editor > Interfaces > Interface # > Network Layer > Cyber**.

In this section, we show how to configure ISAKMP parameters for a specific node using the Default Device Properties Editor. Parameters can be set in the other properties editors in a similar way.

2. Set **Enable ISAKMP** to *Yes* and set the dependent parameters listed in Table 3-25.

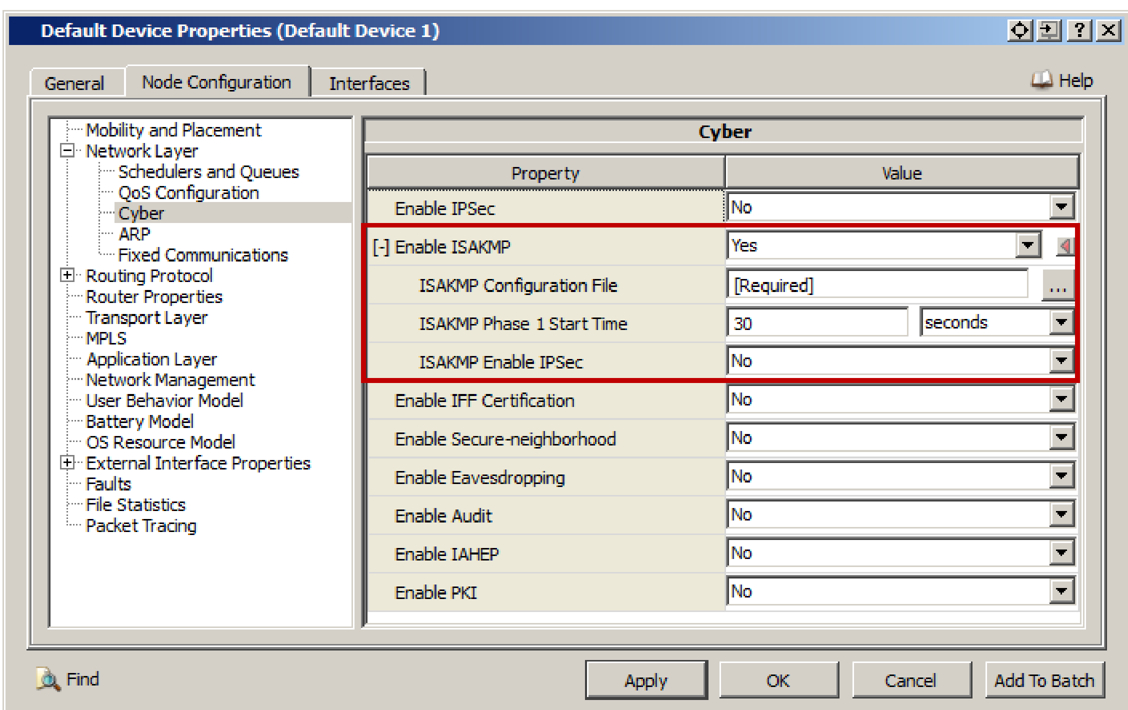


FIGURE 3-8. Setting ISAKMP Parameters

TABLE 3-25. Command Line Equivalent of ISAKMP Parameters

GUI Parameter	Scope of GUI Parameter	Command Line Parameter
ISAKMP Configuration File	Node, Subnet, Interface	`ISAKMP-CONFIG-FILE`
ISAKMP Phase 1 Start Time	Node, Subnet, Interface	`ISAKMP-PHASE-1-START-TIME`
ISAKMP Enable IPSec	Node, Subnet, Interface	`ISAKMP-ENABLE-IPSEC`

3.5.6 Statistics

Table 3-26 lists the statistics collected for the ISAKMP-IKE model that are output to the statistics (.stat) file at the end of simulation.

TABLE 3-26. ISAKMP-IKE Statistics

Statistic	Description
Total Number of Aggressive Exchange	Number of aggressive exchanges performed.
Total Number of Authentication Only Exchange	Number of authentication only exchanges performed.
Total Number of Base Exchange	Number of base exchanges performed.
Total Number of Identity Protection Exchange	Number of identity protection exchanges performed.
Total Number of IKE Main Pre-Shared Key Exchange	Number of IKE main pre-shared key exchanges performed.
Total Number of IKE Main Digital Signature Exchange	Number of IKE main digital signature exchanges performed.
Total Number of IKE Main Public Key Exchange	Number of IKE main public key exchanges performed.
Total Number of IKE Main Revised Public Key Exchange	Number of IKE main revised public key exchanges performed.
Total Number of IKE Aggressive Pre-Shared Key Exchange	Number of IKE aggressive pre-shared key exchanges performed.
Total Number of IKE Aggressive Digital Signature Exchange	Number of IKE aggressive digital signature exchanges performed.
Total Number of IKE Aggressive Public Key Exchange	Number of IKE aggressive public key exchanges performed.
Total Number of IKE Aggressive Revised Public Key Exchange	Number of IKE aggressive revised public key exchanges performed.
Total Number of IKE Quick Exchange	Number of IKE quick exchanges performed.
Total Number of IKE New Group Exchange	Number of IKE new group exchanges performed.
Total Number of Information Exchange Send	Number of informational exchanges sent.
Total Number of Information Exchange Receive	Number of informational exchanges received.
Total Number of Exchange Dropped	Number of exchanges dropped.
Total Number of SA Payload Send	Number of SA payload messages sent.
Total Number of SA Payload Rcv	Number of SA payload messages received.
Total Number of Nonce Payload Send	Number of nonce payload messages sent.
Total Number of Nonce Payload Rcv	Number of nonce payload messages received.
Total Number of Key Exchange Payload Send	Number of key exchange payload messages sent.

TABLE 3-26. ISAKMP-IKE Statistics

Statistic	Description
Total Number of Key Exchange Payload Rcv	Number of key exchange payload messages received.
Total Number of Identity Payload Send	Number of identity payload messages sent.
Total Number of Identity Payload Rcv	Number of identity payload messages received.
Total Number of Signature Payload Send	Number of authentic payload messages sent.
Total Number of Signature Payload Rcv	Number of authentic payload messages received.
Total Number of Hash Payload Send	Number of hash payload messages sent.
Total Number of Hash Payload Rcv	Number of hash payload messages received.
Total Number of Certificate Payload Send	Number of certificate payload messages sent.
Total Number of Certificate Payload Rcv	Number of certificate payload messages received.
Total Number of Notify Payload Send	Number of notify payload messages sent.
Total Number of Notify Payload Rcv	Number of notify payload messages received.
Total Number of Delete Payload Send	Number of delete payload messages sent.
Total Number of Delete Payload Rcv	Number of delete payload messages received.
Total Number of Retransmissions	Number of messages retransmitted.
Total Number of Reestablishments Initiated	Number of phase2 reestablishments initiated.

3.5.7 Sample Scenarios

3.5.7.1 ISAKMP Scenario

3.5.7.1.1 Scenario Description

This scenario tests the normal behavior of ISAKMP implementation for a Tunnel in which the same Security Policies (SP) are used for both inbound and outbound packets. It also illustrates the basic packet exchange during security association establishment.

Topology

Nodes 1 to 6 are connected by wired point-to-point links as shown above. Node 3 and node 4 are negotiating at the interfaces specified in their respective configuration file. See Figure 3-9.

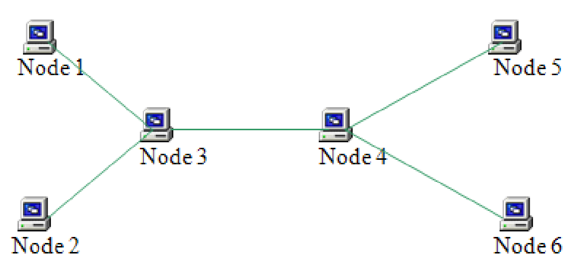


FIGURE 3-9. Topology of the ISAKMP Model

One CBR application is configured from node 1 to node 6.

3.5.7.1.2 Command Line Configuration

Include the following lines in the scenario configuration (.config) file:

```
# Nodes are connected through wired point to point link
#
LINK N8-192.0.0.0 { 1, 3 }
LINK N8-192.0.1.0 { 2, 3 }
LINK N8-192.0.2.0 { 3, 4 }
LINK N8-192.0.3.0 { 4, 5 }
LINK N8-192.0.4.0 { 4, 6 }

[3] ISAKMP-SERVER YES
[3] ISAKMP-CONFIG-FILE node3.isakmp

[4] ISAKMP-SERVER YES
[4] ISAKMP-CONFIG-FILE node4.isakmp

ISAKMP-PHASE-1-START-TIME 3S
ISAKMP-ENABLE-IPSEC NO
```

To configure the application configuration file for sample scenario from the command line, include the following lines in the application configuration (.app) file:

```
CBR 1 6 100 512 1S 60S 0S PRECEDENCE 0
```

Include the following lines in the file “node3.isakmp”:

```
NODE 192.0.2.1
PEER 192.0.2.2 3-4-Config

3-4-Config
PHASE 1
DOI ISAKMP_DOI
EXCHANGE_TYPE EXCH_BASE
FLAGS ACE
CERTIFICATE_ENABLED NO
NEW_GROUP_MODE_ENABLED NO
TRANSFORMS 3DES-SHA-XF

PHASE 2
LOCAL-ID-TYPE IPV4_ADDR_SUBNET
LOCAL-NETWORK 192.0.0.0
LOCAL-NETMASK 255.255.255.0
REMOTE-ID-TYPE IPV4_ADDR_SUBNET
REMOTE-NETWORK 192.0.3.0
REMOTE-NETMASK 255.255.255.0
UPPER-LAYER-PROTOCOL   UDP
DOI IPSEC_DOI
EXCHANGE_TYPE EXCH_IDENT
FLAGS ACE
PROPOSALS ESP-DES-MD5-PFS AH-MD5-PFS
```

```
# Phase 2 proposals
ESP-DES-MD5-PFS
PROTOCOLS ESP-DES-MD5-PFS

AH-MD5-PFS
PROTOCOLS AH-MD5-PFS

# Phase 2 protocols
ESP-DES-MD5-PFS
PROTOCOL_ID ESP
TRANSFORMS ESP-DES-MD5-PFS-XF

AH-MD5-PFS
PROTOCOL_ID AH
TRANSFORMS AH-MD5-PFS-XF

Phase-1-Transforms

TRANSFORM_NAME 3DES-SHA-XF
TRANSFORM_ID KEY_IKE
ENCRYPTION_ALGORITHM DEFAULT
HASH_ALGORITHM SHA
AUTHENTICATION_METHOD RSA_SIG
GROUP_DESCRIPTION MODP_1024
LIFE 60

Phase-2-Transforms

TRANSFORM_NAME ESP-DES-MD5-PFS-XF
TRANSFORM_ID ESP_DES
ENCAPSULATION_MODE TUNNEL
GROUP_DESCRIPTION MODP_1024
AUTHENTICATION_ALGORITHM HMAC-MD5-96
LIFE 30

TRANSFORM_NAME AH-MD5-PFS-XF
TRANSFORM_ID AH_MD5
ENCAPSULATION_MODE TUNNEL
GROUP_DESCRIPTION MODP_1024
AUTHENTICATION_ALGORITHM HMAC-MD5
LIFE 30
```

Include the following lines in the file “node4.isakmp”:

```
NODE 192.0.2.2
PEER 192.0.2.1 4-3-Config

4-3-Config

PHASE 1
DOI ISAKMP_DOI
EXCHANGE_TYPE EXCH_AGGR
FLAGS ACE
CERTIFICATE_ENABLED NO
NEW_GROUP_MODE_ENABLED NO
TRANSFORMS 3DES-SHA-XF

PHASE 2
LOCAL-ID-TYPE IPV4_ADDR_SUBNET
LOCAL-NETWORK 192.0.3.0
LOCAL-NETMASK 255.255.255.0
REMOTE-ID-TYPE IPV4_ADDR_SUBNET
REMOTE-NETWORK 192.0.0.0
REMOTE-NETMASK 255.255.255.0
UPPER-LAYER-PROTOCOL  UDP
DOI IPSEC_DOI
EXCHANGE_TYPE EXCH_IDENT
FLAGS ACE
PROPOSALS ESP-DES-MD5-PFS AH-MD5-PFS

# Phase 2 proposals
ESP-DES-MD5-PFS
PROTOCOLS ESP-DES-MD5-PFS

AH-MD5-PFS
PROTOCOLS AH-MD5-PFS

# Phase 2 protocols
ESP-DES-MD5-PFS
PROTOCOL_ID ESP
TRANSFORMS ESP-DES-MD5-PFS-XF

AH-MD5-PFS
PROTOCOL_ID AH
TRANSFORMS AH-MD5-PFS-XF
```

```
Phase-1-Transforms

TRANSFORM_NAME 3DES-SHA-XF
TRANSFORM_ID KEY_IKE
ENCRYPTION_ALGORITHM DEFAULT
HASH_ALGORITHM SHA
AUTHENTICATION_METHOD RSA_SIG
GROUP_DESCRIPTION MODP_1024
LIFE 60

TRANSFORM_NAME DES-MD5
TRANSFORM_ID KEY_IKE
ENCRYPTION_ALGORITHM DES-CBC
HASH_ALGORITHM MD5
AUTHENTICATION_METHOD PRE_SHARED
GROUP_DESCRIPTION MODP_1024
LIFE 60

Phase-2-Transforms

TRANSFORM_NAME ESP-DES-MD5-PFS-XF
TRANSFORM_ID ESP_DES
ENCAPSULATION_MODE TUNNEL
GROUP_DESCRIPTION MODP_1024
AUTHENTICATION_ALGORITHM HMAC-MD5-96
LIFE 30

TRANSFORM_NAME AH-MD5-PFS-XF
TRANSFORM_ID AH_MD5
ENCAPSULATION_MODE TUNNEL
GROUP_DESCRIPTION MODP_1024
AUTHENTICATION_ALGORITHM HMAC-MD5
LIFE 30
```

3.5.7.1.3 GUI Configuration

To configure the sample scenario in EXata GUI, perform the following steps:

1. Place six nodes of the Default device type. Connect all the nodes with each other as shown in the Figure 3-9.
2. Create ISAKMP configuration files for nodes 3 and 4, as described in Section 3.5.7.1.2**.**
3. For node 3, go to **Default Device Properties Editor > Node Configuration > Network Layer > Cyber**. Set **Enable ISAKMP** to *Yes* and set the dependent parameters as follows:
   a. Set **ISAKMP Configuration File** to the location of the ISAKMP configuration file for node 3.
   b. Set **ISAKMP-PHASE1-START-TIME** to *3S*.
   c. Set **ISAKMP Enable IPSec** to *No.*
4. Configure node 4 in the similar way.
5. Create a CBR application between node 1 and node 6 with the parameters described in Section 3.5.7.1.2**.**

3.5.7.2 IKE Scenario

3.5.7.2.1 Scenario Description

This scenario comprises two ISAKMP-IKE enabled nodes communicating over a wireless channel. Each nodes uses its own ISAKMP configuration file.

3.5.7.2.2 Command Line Configuration

Include the following lines in the scenario configuration (.config) file:

```
# Nodes are connected through a wireless subnet
SUBNET N8-192.0.0.0 { 1, 2 }

# ISAKMP configuration
[ 1 2 ] ISAKMP-SERVER YES
[ 1 ] ISAKMP-CONFIG-FILE node1.isakmp
[ 2 ] ISAKMP-CONFIG-FILE node2.isakmp
```

To configure the application configuration file for sample scenario from the command line, include the following lines in the application configuration (.app) file:

```
CBR 1 2 10 512 1S 5M 0S PRECEDENCE 0
```

Include the following lines in the file “node1.isakmp”:

```
NODE 192.0.0.1
PEER 192.0.0.2 1-2-Config

1-2-Config

PHASE 1
DOI ISAKMP_DOI
EXCHANGE_TYPE EXCH_AGG_DIG_SIG
FLAGS NONE
CERTIFICATE_ENABLED YES
NEW_GROUP_MODE_ENABLED YES
TRANSFORMS DES-SHA-XF

PHASE 2
LOCAL-ID-TYPE IPV4_ADDR_SUBNET
LOCAL-NETWORK 192.0.0.0
LOCAL-NETMASK 255.255.255.0
REMOTE-ID-TYPE IPV4_ADDR_SUBNET

REMOTE-NETWORK 192.0.0.0
REMOTE-NETMASK 255.255.255.0
UPPER-LAYER-PROTOCOL  UDP
DOI IPSEC_DOI
EXCHANGE_TYPE EXCH_QUICK
FLAGS NONE
PROPOSALS ESP-DES-MD5-PFS AH-MD5-PFS

# Phase 2 proposals
ESP-DES-MD5-PFS
PROTOCOLS ESP-DES-MD5-PFS

AH-MD5-PFS
PROTOCOLS AH-MD5-PFS

# Phase 2 protocols
ESP-DES-MD5-PFS
PROTOCOL_ID ESP
TRANSFORMS ESP-DES-MD5-PFS-XF

AH-MD5-PFS
PROTOCOL_ID AH
TRANSFORMS AH-MD5-PFS-XF
```

```
Phase-1-Transforms

TRANSFORM_NAME DES-SHA-XF
TRANSFORM_ID KEY_IKE
ENCRYPTION_ALGORITHM DES-CBC
HASH_ALGORITHM SHA
CERTIFICATION_TYPE PKCS7
AUTHENTICATION_METHOD RSA_SIG
GROUP_DESCRIPTION MODP_1024
LIFE 60

Phase-2-Transforms

TRANSFORM_NAME ESP-DES-MD5-PFS-XF
TRANSFORM_ID ESP_DES
ENCAPSULATION_MODE TRANSPORT
GROUP_DESCRIPTION MODP_1024
AUTHENTICATION_ALGORITHM HMAC-MD5-96
LIFE 30

TRANSFORM_NAME AH-MD5-PFS-XF
TRANSFORM_ID AH_MD5
ENCAPSULATION_MODE TRANSPORT
GROUP_DESCRIPTION MODP_1024
AUTHENTICATION_ALGORITHM HMAC-MD5
LIFE 30
```

Include the following lines in the file “node2.isakmp”:

```
NODE 192.0.0.2
PEER 192.0.0.1 2-1-Config

2-1-Config

PHASE 1
DOI ISAKMP_DOI
EXCHANGE_TYPE EXCH_AGG_DIG_SIG
FLAGS NONE
CERTIFICATE_ENABLED YES
NEW_GROUP_MODE_ENABLED YES
TRANSFORMS DES-MD5-XF

PHASE 2
LOCAL-ID-TYPE IPV4_ADDR_SUBNET
LOCAL-NETWORK 192.0.0.0
LOCAL-NETMASK 255.255.255.0
REMOTE-ID-TYPE IPV4_ADDR_SUBNET
REMOTE-NETWORK 192.0.0.0
REMOTE-NETMASK 255.255.255.0
UPPER-LAYER-PROTOCOL  UDP
DOI IPSEC_DOI
EXCHANGE_TYPE EXCH_QUICK
FLAGS NONE
PROPOSALS ESP-DES-MD5-PFS AH-MD5-PFS
#phase 2 proposals
ESP-DES-MD5-PFS
PROTOCOLS ESP-DES-MD5-PFS

AH-MD5-PFS
PROTOCOLS AH-MD5-PFS

# Phase 2 protocols
ESP-DES-MD5-PFS
PROTOCOL_ID ESP
TRANSFORMS ESP-DES-MD5-PFS-XF

AH-MD5-PFS
PROTOCOL_ID AH
TRANSFORMS AH-MD5-PFS-XF
```

```
Phase-1-Transforms

TRANSFORM_NAME DES-MD5-XF
TRANSFORM_ID KEY_IKE
ENCRYPTION_ALGORITHM DES-CBC
HASH_ALGORITHM MD5
CERTIFICATION_TYPE PKCS7
AUTHENTICATION_METHOD PRE_SHARED
GROUP_DESCRIPTION MODP_1024
LIFE 60

Phase-2-Transforms

TRANSFORM_NAME ESP-DES-MD5-PFS-XF
TRANSFORM_ID ESP_DES
ENCAPSULATION_MODE TRANSPORT
GROUP_DESCRIPTION MODP_1024
AUTHENTICATION_ALGORITHM HMAC-MD5-96
LIFE 30

TRANSFORM_NAME AH-MD5-PFS-XF
TRANSFORM_ID AH_MD5
ENCAPSULATION_MODE TRANSPORT
GROUP_DESCRIPTION MODP_1024
AUTHENTICATION_ALGORITHM HMAC-MD5
LIFE 30
```

3.5.7.2.3 GUI Configuration

To configure the sample scenario in EXata GUI, perform the following steps:

1. Place two nodes of the Default device type. Place a wireless subnet and connect the two nodes to the wireless subnet.
2. Create ISAKMP configuration files for nodes 1 and 2, as described in Section 3.5.7.2.2.
3. For node 1, go to **Default Device Properties Editor > Node Configuration > Network Layer > Cyber**. Set **Enable ISAKMP** to *Yes* and set the dependent parameters as follows:
   a. Set **ISAKMP Configuration File** to the location of the ISAKMP configuration file for node 1.
   b. Set **ISAKMP-PHASE1-START-TIME** to *30S*.
   c. Set **ISAKMP Enable IPSec** to *No.*
4. Configure node 2 in the similar way.
5. Create a CBR application between node 1 and node 2 with the parameters described in Section 3.5.7.2.2.

3.5.8 Scenarios Included in EXata

The EXata distribution includes several sample scenarios for the ISAKMP-IKE model. All scenarios are located in the directory EXATA_HOME/scenarios/cyber/isakmp. Table 3-27 lists the sub-directory where each scenario is located.

TABLE 3-27. ISAKMP-IKE Model Scenarios

Scenario Sub-directory	Description
DifferentApplication	Shows the wildcard ISAKMP settings (in .isakmp file) under multiple applications (in .app file)
Manet	Shows the wildcard ISAKMP settings (in .isakmp file) in a mobile ad hoc network
Wired/many-to-one	Shows the multiple applications from multiple ISAKMP server to one ISAKMP server
Wired/mixed	Shows the ISAKMP functionalities for a mixed network
Wired/multiple-sp	Shows the ISAKMP functionalities for multiple ISAKMP tunnels between two gateways nodes. Such type of tunnels can be established by establishing different IPsec SA at each of the tunnel end nodes
Wired/nested-tunnel	Shows the ISAKMP functionality for nested tunneling
Wired/one-to-many	Shows the multiple applications go through one ISAKMP server to multiple other ISAKMP servers
Wired/simple-tunnel	Shows the simple tunnel between a pair of ISAKMP enabled nodes
Wired-wildcard/many-to-one	Shows the multiple applications from multiple ISAKMP server to one ISAKMP server
Wired-wildcard/mixed	Shows the functionalities of ISAKMP implementation for wild card in wired subnet and link network
Wired-wildcard/multiple-sp	Shows the ISAKMP functionalities for multiple ISAKMP tunnels between two gateways nodes as well as the functionary of multiple security association proposals between two nodes
Wired-wildcard/nested-tunnel	Shows the ISAKMP functionalities for nested tunneling using wildcard setting
Wired-wildcard/one-to-many	Shows the functionalities of ISAKMP implementation for wild card ISAKMP configuration between one and multiple ISAKMP servers

TABLE 3-27. ISAKMP-IKE Model Scenarios (Continued)

Scenario Sub-directory	Description
Wired-wildcard/simple-tunnel	Shows the functionalities of ISAKMP implementation with a wild card in a simple tunnel
Wireless	Shows the functionalities of ISAKMP implementation in a simple wireless subnet
Wrd-Wrls-Wrd	Shows the functionalities of ISAKMP implementation in a simple wired wireless Combination Network
Wrls-Wrd-Wrls-WldCd	Shows the functionalities of ISAKMP implementation in a simple ISAKMP Wireless-Wired-Wireless Combination Network

3.5.9 References

1. RFC 2407, The Internet IP Security Domain of Interpretation for ISAKMP. D. Piper. November 1998. http://www.ietf.org/rfc/rfc2407.txt.
2. RFC 2408, Internet Security Association and Key Management Protocol (ISAKMP). D. Maughan, M. Schertler, M. Schneider, J. Turner. November 1998. http://www.ietf.org/rfc/rfc2408.txt.
3. RFC2409, The Internet Key Exchange (IKE). D. Harkins, D. Carrel. November 1998. http://www.ietf.org/rfc/rfc2409.txt.
4. RFC 2412, The OAKLEY Key Determination Protocol. H. Orman. November 1998. http://www.ietf.org/rfc/rfc2412.txt.
5. RFC 2401, Security Architecture for the Internet Protocol. S. Kent, R. Atkinson. November 1998. http://www.ietf.org/rfc/rfc2401.txt

3.6 Public Key Infrastructure (PKI) Model

3.6.1 Description

A PKI is an infrastructure that uses digital certificates as an authentication mechanism and is built to better manage certificates and their associated keys. A digital certificate is itself a way to reliably identify the user or computer claiming to be the owner of a specific public key.

A PKI can be implemented within an organization, for the use of the users on its network, or it can be a commercial entity that issues certificates to Internet users, such as VeriSign. The PKI consists of the following components:

- At least one certification authority (CA) to issue certificates.
- Policies that govern the operation of the PKI.
- The digital certificates.

The PKI model supports the basic features of security including encryption-decryption, signing, verification, and certificate reading.

3.6.2 Features and Assumptions

This section describes the implemented features, omitted features, assumptions and limitations of the PKI model.

3.6.2.1 Implemented Features

- Assigning certificates to a node through configuration parameter.
- Generation of X.509 certificate.
- Generation of X.509 private key file.
- Extracting information from certificate.
- Encrypt a packet/message.
- Decrypt the packet/message.
- Sign packet.
- Verify the signature.

3.6.2.2 Omitted Features

None.

3.6.2.3 Assumptions and Limitations

- Currently PKI uses des_ede3_cbc cipher and SHA1 hash algorithm.
- Certificate Authority (CA) is not supported.
- The certificates are available to all nodes at the start of the simulation and are read during the initialization phase.
- Certificate management protocols are not supported.
- The certificate revocation list is not supported.

3.6.3 Supplemental Information

None.

3.6.4 Command Line Configuration

To specify PKI model, include the following parameter in the scenario configuration (.config) file:

```
[<Qualifier>]   PKI-ENABLED    YES
```

The scope of this parameter declaration can be Global or Node. See Section 1.2.1.1 for a description of `<Qualifier>` for each scope.

Note: The default value of the parameter **`PKI-ENABLED`** is **`NO`**.

PKI Model Parameters

Table 3-28 lists the configuration parameters for the PKI model. See Section 1.2.1.3 for a description of the format used for the parameter table.

TABLE 3-28. IAHEP Parameters

Parameter	Value	Description
`PKI-CONFIGURATION-FILE` *Required* *Scope:* Global, Node	Filename	Specifies the name of the PKI Configuration file. See Section 3.6.4.1 for the format of the PKI Configuration file.

3.6.4.1 Format of PKI Configuration File

The PKI Configuration file has the same format as the scenario configuration (.config) file. Table 3-29 describes the parameters that can be specified in the PKI Configuration file. See Section 1.2.1.3 for a description of the format used for the parameter table.

TABLE 3-29. PKI Configuration File Parameters

Parameter	Value	Description
GENERATED-CERTIFICATE-PRIVATEKEY-PAIR-COUNT *Optional* *Scope:* Global, Node	Integer *Range:* ≥ 0 *Default:* 0	Number of certificates allocated to a node that are generated by the PKI model.
CERTIFICATE-FILENAME *Optional* *Scope:* Global, Node *Instances:* Index of the certificate	String *Default:* See description	Name of the generated certificate file. By default, the generated certificate file is called certificate.<nodeID>.<instance>.pem where <nodeID>: ID of the node <index>: Index of the generated certificate **Note:** This parameter is applicable only if GENERATED-CERTIFICATE-PRIVATEKEY-PAIR-COUNT > 0.
CERTIFICATE-OWNER-COUNTRYNAME *Optional* *Scope:* Global, Node *Instances:* Index of the certificate	String *Default:* US	Name of the owner's country for the generated certificate. **Note:** This parameter is applicable only if GENERATED-CERTIFICATE-PRIVATEKEY-PAIR-COUNT > 0.
CERTIFICATE-OWNER-STATENAME *Optional* *Scope:* Global, Node *Instances:* Index of the certificate	String *Default:* See description	Name of the owner's state for the generated certificate. By default, the state name is CA_<nodeID> where <nodeID>: ID of the node **Note:** This parameter is applicable only if GENERATED-CERTIFICATE-PRIVATEKEY-PAIR-COUNT > 0.
CERTIFICATE-OWNER-LOCATION *Optional* *Scope:* Global, Node *Instances:* Index of the certificate	String *Default:* See description	Owner's location for the generated certificate. By default, the location is LA_<nodeID> where <nodeID>: ID of the node **Note:** This parameter is applicable only if GENERATED-CERTIFICATE-PRIVATEKEY-PAIR-COUNT > 0.

TABLE 3-29. PKI Configuration File Parameters (Continued)

Parameter	Value	Description
`CERTIFICATE-OWNER-ORGNAME` *Optional* *Scope:* Global, Node *Instances:* Index of the certificate	String *Default:* See description	Name of the owner's organization for the generated certificate. By default, the organization name is `SNT_<nodeID>` where `<nodeID>`: ID of the node **Note:** This parameter is applicable only if `GENERATED-CERTIFICATE-PRIVATEKEY-PAIR-COUNT` > 0.
`CERTIFICATE-OWNER-ORGUNIT` *Optional* *Scope:* Global, Node *Instances:* Index of the certificate	String *Default:* See description	Owner's organization unit for the generated certificate. By default, the organization unit is `EXata_<nodeID>` where `<nodeID>`: ID of the node **Note:** This parameter is applicable only if `GENERATED-CERTIFICATE-PRIVATEKEY-PAIR-COUNT` > 0.
`CERTIFICATE-OWNER-COMMONNAME` *Optional* *Scope:* Global, Node *Instances:* Index of the certificate	String *Default:* See description	Name of the owner for the generated certificate. By default, the common name is `QualNet_<nodeID>_<instance>` where `<nodeID>`: ID of the node `<index>`: Index of the generated certificate **Note:** This parameter is applicable only if `GENERATED-CERTIFICATE-PRIVATEKEY-PAIR-COUNT` > 0. **Note:** `CERTIFICATE-OWNER-COMMONNAME` should be different for each certificate.
`CERTIFICATE-OWNER-EMAIL` *Optional* *Scope:* Global, Node *Instances:* Index of the certificate	String *Default:* `support@scalable-networks.com`	Owner's e-mail ID for the generated certificate. **Note:** This parameter is applicable only if `GENERATED-CERTIFICATE-PRIVATEKEY-PAIR-COUNT` > 0.
`CERTIFICATE-ISSUER-COUNTRYNAME` *Optional* *Scope:* Global, Node *Instances:* Index of the certificate	String *Default:* `US`	Name of the issuer's country for the generated certificate. **Note:** This parameter is applicable only if `GENERATED-CERTIFICATE-PRIVATEKEY-PAIR-COUNT` > 0.

TABLE 3-29. PKI Configuration File Parameters (Continued)

Parameter	Value	Description
CERTIFICATE-ISSUER-STATENAME *Optional* *Scope:* Global, Node *Instances:* Index of the certificate	String *Default:* See description	Name of the issuer's state for the generated certificate. By default, the state name is CA_<nodeID> where <nodeID>: ID of the node **Note:** This parameter is applicable only if GENERATED-CERTIFICATE-PRIVATEKEY-PAIR-COUNT > 0.
CERTIFICATE-ISSUER-LOCATION *Optional* *Scope:* Global, Node *Instances:* Index of the certificate	String *Default:* See description	Issuer's location for the generated certificate. By default, the location is LA_<nodeID> where <nodeID>: ID of the node **Note:** This parameter is applicable only if GENERATED-CERTIFICATE-PRIVATEKEY-PAIR-COUNT > 0.
CERTIFICATE-ISSUER-ORGNAME *Optional* *Scope:* Global, Node *Instances:* Index of the certificate	String *Default:* See description	Name of the issuer's organization for the generated certificate. By default, the organization name is SNT_<nodeID> where <nodeID>: ID of the node **Note:** This parameter is applicable only if GENERATED-CERTIFICATE-PRIVATEKEY-PAIR-COUNT > 0.
CERTIFICATE-ISSUER-ORGUNIT *Optional* *Scope:* Global, Node *Instances:* Index of the certificate	String *Default:* See description	Issuer's organization unit for the generated certificate. By default, the organization unit is EXata_<nodeID> where <nodeID>: ID of the node **Note:** This parameter is applicable only if GENERATED-CERTIFICATE-PRIVATEKEY-PAIR-COUNT > 0.
CERTIFICATE-ISSUER-COMMONNAME *Optional* *Scope:* Global, Node *Instances:* Index of the certificate	String *Default:* See description	Issuer's name for the generated certificate. By default, the common name is QualNet_<nodeID>_<instance> where <nodeID>: ID of the node <index>: Index of the generated certificate **Note:** This parameter is applicable only if GENERATED-CERTIFICATE-PRIVATEKEY-PAIR-COUNT > 0.

TABLE 3-29. PKI Configuration File Parameters (Continued)

Parameter	Value	Description
`CERTIFICATE-ISSUER-EMAIL` *Optional* *Scope:* Global, Node *Instances:* Index of the certificate	String *Default:* `support@scalable-networks.com`	Issuer's E-mail ID for the generated certificate. **Note:** This parameter is applicable only if `GENERATED-CERTIFICATE-PRIVATEKEY-PAIR-COUNT` > 0.
`PRIVATE-KEY-TYPE` *Optional* *Scope:* Global, Node *Instances:* Index of the certificate	List: • `DSA` • `RSA` *Default:* `RSA`	Private key type for the generated certificate. **Note:** This parameter is applicable only if `GENERATED-CERTIFICATE-PRIVATEKEY-PAIR-COUNT` > 0.
`PRECONFIGURED-CERTIFICATE-PRIVATEKEY-PAIR-COUNT` *Optional* *Scope:* Global, Node	Integer *Range:* ≥ `0` *Default:* `0`	Number of pre-configured <private key, certificate> pairs assigned to a node.
`PRECONFIGURED-CERTIFICATE-FILE-TYPE` *Required* *Scope:* Global, Node *Instances:* Index of the certificate	List: • `PEM` • `P12`	Type of the pre-configured certificate file. There should be `PRECONFIGURED-CERTIFICATE-PRIVATEKEY-PAIR-COUNT` instances of this parameter. **Note:** This parameter is applicable only if `PRECONFIGURED-CERTIFICATE-PRIVATEKEY-PAIR-COUNT` > 0.
`PRECONFIGURED-CERTIFICATE-FILE` *Required* *Scope:* Global, Node *Instances:* Index of the certificate	String	Name of the pre-configured certificate file. There should be `PRECONFIGURED-CERTIFICATE-PRIVATEKEY-PAIR-COUNT` instances of this parameter. **Note:** This parameter is applicable only if `PRECONFIGURED-CERTIFICATE-PRIVATEKEY-PAIR-COUNT` > 0.
`PRECONFIGURED-PRIVATE-KEY-FILE` *Required* *Scope:* Global, Node *Instances:* Index of the certificate	String	Name of the pre-configured private key file. There should be `PRECONFIGURED-CERTIFICATE-PRIVATEKEY-PAIR-COUNT` instances of this parameter. **Note:** This parameter is applicable only if `PRECONFIGURED-CERTIFICATE-PRIVATEKEY-PAIR-COUNT` > 0.
`PRECONFIGURED-SHARED-CERTIFICATE-COUNT` *Optional* *Scope:* Global, Node	Integer *Range:* ≥ `0` *Default:* `0`	Number of pre-configured shared certificates allocated to a node.

TABLE 3-29. PKI Configuration File Parameters (Continued)

Parameter	Value	Description
PRECONFIGURED-SHARED-CERTIFICATE-FILE-TYPE *Required* *Scope:* Global, Node *Instances:* Index of the certificate	List: • PEM • P7B	Type of the pre-configured shared certificate file which will be loaded at the node. There should be PRECONFIGURED-SHARED-CERTIFICATE-COUNT instances of this parameter. **Note:** This parameter is applicable only if PRECONFIGURED-SHARED-CERTIFICATE-COUNT > 0.
PRECONFIGURED-SHARED-CERTIFICATE-FILE *Required* *Scope:* Global, Node *Instances:* Index of the certificate	String	Name of the pre-configured shared certificate file. There should be PRECONFIGURED-SHARED-CERTIFICATE-COUNT instances of this parameter. **Note:** This parameter is applicable only if PRECONFIGURED-SHARED-CERTIFICATE-COUNT > 0.

3.6.5 GUI Configuration

To configure the PKI model in the GUI, perform the following steps:

1. Go to **Default Device Properties Editor > Node Configuration > Network Layer > Cyber**.
2. Set **Enable PKI** to *Yes* and set the dependent parameters listed in Table 3-30.

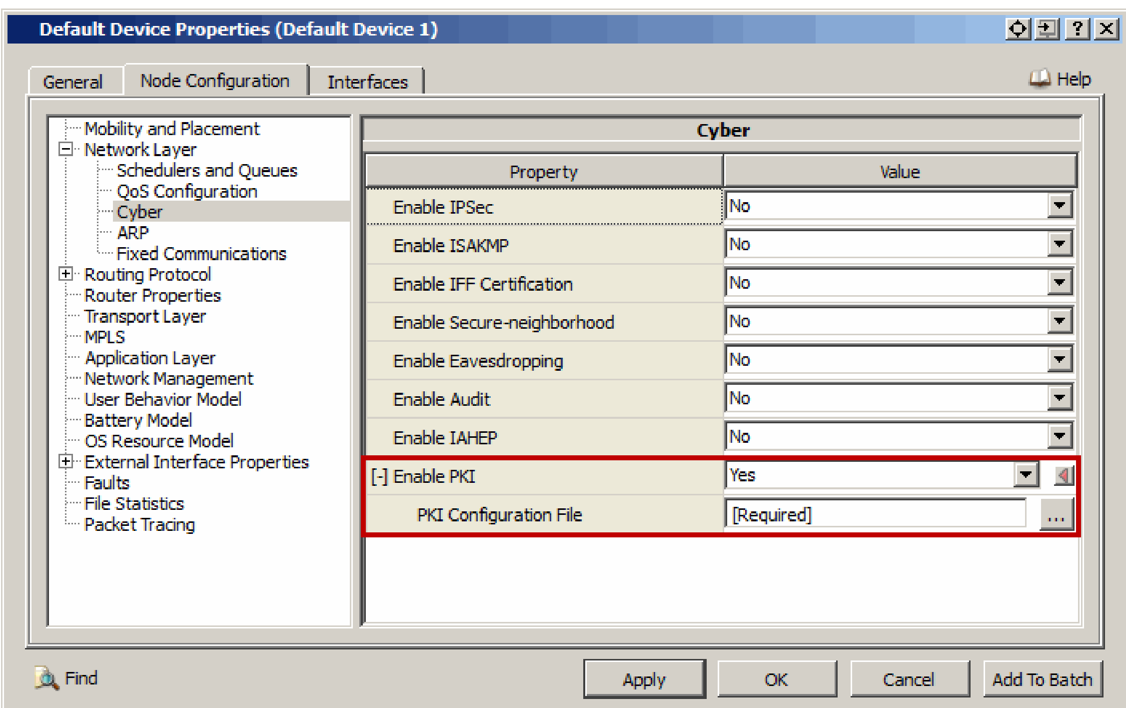


FIGURE 3-10. Enabling PKI

TABLE 3-30. Command Line Equivalent of PKI Parameters

GUI Parameter	Scope of GUI Parameter	Command Line Parameter
Enable PKI	Node	PKI-ENABLED
PKI Configuration File	Node	PKI-CONFIGURATION-FILE

Setting Parameters

- Set **PKI Configuration File** to the name of the PKI configuration file. The format of the PKI configuration file is described in Section 3.6.4.1.

3.6.6 Statistics

No statistics are collected for the PKI model.

3.6.7 Sample Scenario

3.6.7.1 Scenario Description

There are two nodes connected by a point-to-point link which generate certificates and private key files for communication.

3.6.7.2 Command Line Configuration

The scenario configuration (.config) file for the sample scenario should include the following parameters:

```
EXPERIMENT-NAME cert-privateKey-generate
SIMULATION-TIME   10M
SEED   1

COORDINATE-SYSTEM    CARTESIAN
TERRAIN-DIMENSIONS   (1500, 1500)
NODE-PLACEMENT FILE
NODE-POSITION-FILE cert-privateKey-generate.nodes
MOBILITY   FILE
LINK N2-190.0.1.0  { 1, 2 }
LINK-BANDWIDTH            112000
LINK-PROPAGATION-DELAY   50MS
MAC-PROTOCOL MAC802.3
SUBNET-DATA-RATE              10000000
SUBNET-PROPAGATION-DELAY     1US
PROMISCUOUS-MODE   NO
NETWORK-PROTOCOL   IP
IP-QUEUE-NUM-PRIORITIES   3
IP-QUEUE-PRIORITY-QUEUE-SIZE   50000
IP-QUEUE-TYPE   FIFO
IP-QUEUE-SCHEDULER   STRICT-PRIORITY
IP-FORWARDING YES
ROUTING-PROTOCOL   OSPFv2

PKI-ENABLED YES
PKI-CONFIGURATION-FILE signEncrypt.pki
```

The PKI configuration file, signEncrypt.pki, is:

```
[1] PRECONFIGURED-CERTIFICATE-PRIVATEKEY-PAIR-COUNT 2
[1] PRECONFIGURED-CERTIFICATE-FILE-TYPE[0] PEM
[1] PRECONFIGURED-CERTIFICATE-FILE[0] cert4.pem
[1] PRECONFIGURED-PRIVATE-KEY-FILE[0] pkey4.pem
[1] PRECONFIGURED-CERTIFICATE-FILE-TYPE[1] P12
[1] PRECONFIGURED-CERTIFICATE-FILE[1] CSS-000000011-SecretComm.p12

[1] PRECONFIGURED-SHARED-CERTIFICATE-COUNTS 1
[1] PRECONFIGURED-SHARED-CERTIFICATE-FILE-TYPE[0] PEM
[1] PRECONFIGURED-SHARED-CERTIFICATE-FILE[0] sharedcert5.pem

[2] PRECONFIGURED-CERTIFICATE-PRIVATEKEY-PAIR-COUNT  1
[2] PRECONFIGURED-CERTIFICATE-FILE-TYPE[0] PEM
[2] PRECONFIGURED-CERTIFICATE-FILE[0] cert5.pem
[2] PRECONFIGURED-PRIVATE-KEY-FILE[0] pkey5.pem

[2] PRECONFIGURED-SHARED-CERTIFICATE-COUNTS 2
[2] PRECONFIGURED-SHARED-CERTIFICATE-FILE-TYPE[0] PEM
[2] PRECONFIGURED-SHARED-CERTIFICATE-FILE[0] sharedcert4.pem
[2] PRECONFIGURED-SHARED-CERTIFICATE-FILE-TYPE[1] P7B
[2] PRECONFIGURED-SHARED-CERTIFICATE-FILE[1] CSS-000000011-SecretComm.p7b
```

3.6.7.3 GUI Configuration

To configure the sample scenario in GUI, perform the following steps:

1. Place two default nodes.
2. Create a file, signEncrypt.pki, as described in Section Section 3.6.7.2
3. For each of the nodes, go to **Default Device Properties Editor > Node Configuration > Network Layer > Cyber** (see Figure 3-10).
    a. Set **Enable PKI** to **Yes**
    b. Set **PKI Configuration File** to signEncrypt.pki.

3.6.8 Scenarios Included in EXata

The EXata distribution includes several sample scenarios for the PKI model. All scenarios are located in the directory EXATA_HOME/scenarios/cyber/pki. Table 3-31 lists the sub-directory where each scenario is located.

TABLE 3-31. PKI Scenarios Included in EXata

Scenario	Description
SignEncrypt	Shows PKI configuration for pem/ p12/P7B certificate files which can be used by any application to sign/verify or encrypt/decrypt data.
pki-certificate-generation	Shows certificates and private key generation.
pki-signencrypt-single	Shows PKI configuration for single certificate file which can be used by any application to sign/verify or encrypt/decrypt data.

TABLE 3-31. PKI Scenarios Included in EXata (Continued)

Scenario	Description
Passphrase\pki-certificate-generation	Shows certiface and private key (both RSA and DSA) generation of PKI.
Passphrase\pki-signencrypt-single	Shows sign/verification and encryption/decryption of PKI using single certificate file.

3.6.9 References

1. RFC 5280: Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile
2. RFC 2510: Internet X.509 Public Key Infrastructure Certificate Management Protocols
3. RFC 2459: Internet X.509 Public Key Infrastructure Certificate and CRL Profile
4. Article on Public Key Infrastructure (http://en.wikipedia.org/wiki/Public_key_infrastructure)
5. Article on Public Key Cryptography (http://en.wikipedia.org/wiki/Public-key_cryptography)
6. Microsoft's Technical Article on PKI (http://technet.microsoft.com/en-us/library/cc779826(WS.10).aspx)

3.7 Secure Neighbor

The EXata Secure Neighbor model is based on the publications referred to in the Reference section.

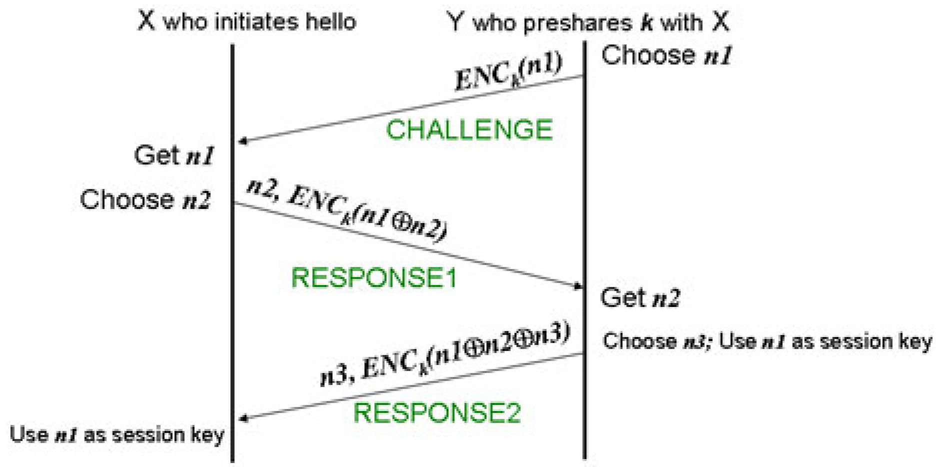
3.7.1 Description

The secure neighbor authentication has two variants. The first variant is based on *pair-wise shared secrets*, and the second variant is based on *certification*.

In secure neighbor authentication (SNAuth), every mobile node establishes an authenticated neighborhood on the move. Periodically, every mobile node X broadcasts its identity packet <SNAuth-HELLO, X> to its neighborhood.

1. In the pair-wise shared secret variant of SNAuth, Y, a neighboring receiver of the identity broadcast initiates a 3-way challenge-response handshake to authenticate X, the sender of the identity broadcast.
   - **a.** Suppose X and Y share a pair-wise secret k. Now Y selects a random nonce n1, encrypts n1 with k, sends the encrypted result ENC_k (n1) to X by a message <CHALLENGE, Y, ENC_k (n1)>.
   - **b.** If the receiver of the challenge message is indeed X, then it can decrypt ENC_k (n1) and sees n1. X selects another random nonce n2, encrypts ENC_k (n1 XOR n2), and sends back <RESPONSE1, X, n2, ENC_k (n1 XOR n2)> as the response to the challenger Y.
   - **c.** When Y receives the response, Y decrypts ENC_k (n1 XOR n2) and obtains n1 XOR n2. If Y can get the same result from XORing n2 in the response and its own challenge n1, then X passes the test with success. Otherwise, Y does not send any packet to X and does not receive packets from X except the response packets, until a correct <RESPONSE1> packet from X can pass the test. Upon detecting a success, Y puts X in its secure neighbor list. Y selects a random nonce n3 and sends out a confirmation response <RESPONSE2, Y, n3, ENC_k (n1 XOR n2 XOR n3)> to X.
   - **d.** Upon receiving the RESPONSE2 message, X decrypts ENC_k (n1 XOR n2 XOR n3) and obtains n1 XOR n2 XOR n3. If this matches the result of XORing n1 that is previously decrypted, its own n2 and n3 in the RESPONSE2 packet, then X inserts Y into its secure neighbor list. (This three-way handshake is required because X needs to verify that Y actually knows k)
   - **e.** End of the challenge-response protocol.

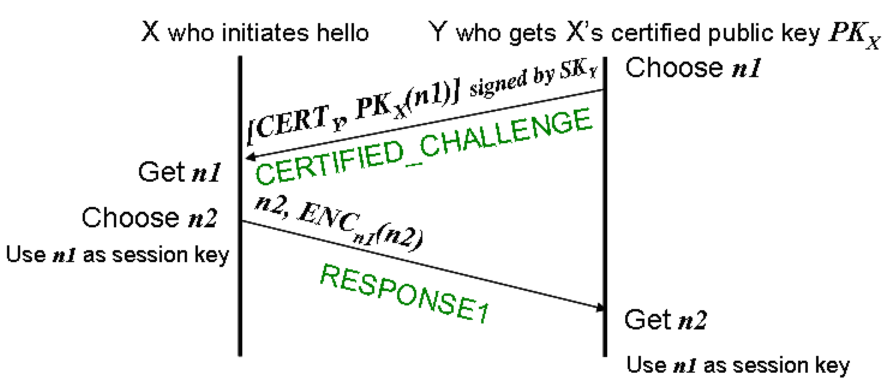
Note: The cryptographic term “nonce” is used above to mean a value that is used only once.



In the above description, all nonce length is currently set to 128-bit long. Encryption block length is 128-bit. Key k can be 128-bit, 192-bit, or 256-bit. Session key means that the key n1 is used until the time when the next HELLO received by Y from X successfully passes the test again.

2. A slightly different challenge-response scheme is used if Y does not pre-share a master secret k with X. Here X must broadcast its certificate $CERT_X$ = [X, certified public key PK_X, certificate valid time] in a CERTIFIED_HELLO message. For Y's CHALLENGE, Y uses PK_X to encrypt n1 and obtains ciphertext PK_X (n1). Y must also add its own certificate $CERT_Y$ = [Y, certified public key PK_Y, certificate valid time] and sign the entire message with its own private key SK_Y. We recommend the public key cryptosystem in use be an Elliptic Curve Cryptosystem (ECC), because ECC features shorter certificate length and ciphertext length, thus incurring less communication overhead.

As depicted below, there are a number of computational changes, and RESPONSE2 is spared, but the RESPONSE message format is unchanged.



When every neighboring receiver of X finishes the authentication and key-agreement process, node X obtains a secure snapshot of its neighborhood. In the neighborhood, every other node is authenticated and shares an IPsec security association with the node X. As the SNAuth protocol runs on every mobile node, the statement is true if node X is replaced with any node X'.

Caveats

- All the above secure neighbor authentication variants may fail to reach the session key establishment final phase due to jamming, packet loss, etc. In other words, the adversary can deny the protocol execution. However, the adversary cannot forge (uncompromised) neighboring nodes' identities.
- Brute-force jamming and wormhole attacks are feasible attacks to foil secure neighbor authentication. Brute-force jamming can be thwarted by countermeasures such as spread spectrum and forward error correction. Wormhole attack can be thwarted by countermeasures such as distance-bounding protocols. These attacks are not studied here.

SNAuth is a building block for other advanced network security services. For example, in secure routing, you can enforce a rule that the current node only forwards packets for those nodes detected by SNAuth. Packets from other nodes not detected by SNAuth are dropped. This way, packets from unauthenticated nodes are limited in their immediate neighborhoods. The danger of denial-of-service is hence limited in unauthenticated nodes' immediate neighborhoods.

3.7.2 Features and Assumptions

This section describes the implemented features, omitted features, assumptions and limitations of the Secure Neighbor model.

3.7.2.1 Implemented Features

- Periodic announcing of certified credentials per node
- Two-way and three-way Challenge-Response scheme
- Interface with certification

3.7.2.2 Omitted Features

- Actual crypto-processing
- Interface with ISAKMP

3.7.2.3 Assumptions and Limitations

None.

3.7.3 Supplemental Information

None.

3.7.4 Command Line Configuration

To enable Secure Neighbor model, include the following parameter in the scenario configuration (.config) file:

```
[<Qualifier>]   SECURE-NEIGHBOR-ENABLED    YES
```

The scope of this parameter declaration can be Global or Node. See Section 1.2.1.1 for a description of `<Qualifier>` for each scope.

Note: The default value of this parameter is `NO`.

Secure Neighbor-specific Parameters

Table 3-32 lists the Secure Neighbor parameters specified in the scenario configuration (.config) file. See Section 1.2.1.3 for a description of the format used for the parameter table.

TABLE 3-32. Secure Neighbor-specific Parameters

Parameter	Value	Description
SECURE-NEIGHBOR-TIMEOUT Optional *Scope:* Global, Node	Time *Range:* [1 to 1000000000000000] *Default:* 5S	Specifies the time interval for which a node waits to do next neighbor detection handshake. **Note:** For fast mobile scenarios, reduce the value to get fresher snapshots. For slow mobile scenarios, enlarge the value to reduce overhead.
SECURE-NEIGHBOR-CERTIFIED-HELLO Optional *Scope:* Global, Node	List: • YES • NO *Default:* NO	Specifies whether or not the network will assume that a pair-wise secret is pre-shared between two nodes. YES: If set to YES, secure neighbor uses the Certificate Variant, which is a two way challenge response scheme which bears sender's certificate in the hello message NO: If set to No, secure neighbor uses the pair-wise shared secret variant of secure neighborhood, which is a three way challenge response scheme

Examples of Parameter Usage

The following configurations enables secure neighbor in node 1:

```
[ 1 ] SECURE-NEIGHBOR-ENABLED YES
[ 1 ] SECURE-NEIGHBOR-TIMEOUT 5S
[ 1 ] SECURE-NEIGHBOR-CERTIFIED-HELLO NO
```

3.7.5 GUI Configuration

To configure the general Secure Neighbor parameters, perform the following steps:

1. Go to **Default Device Properties Editor > Node Configuration > Cyber**.
2. Set **Enable Secure-neighborhood** to *Yes* and set the dependent parameters listed in Table 3-33.

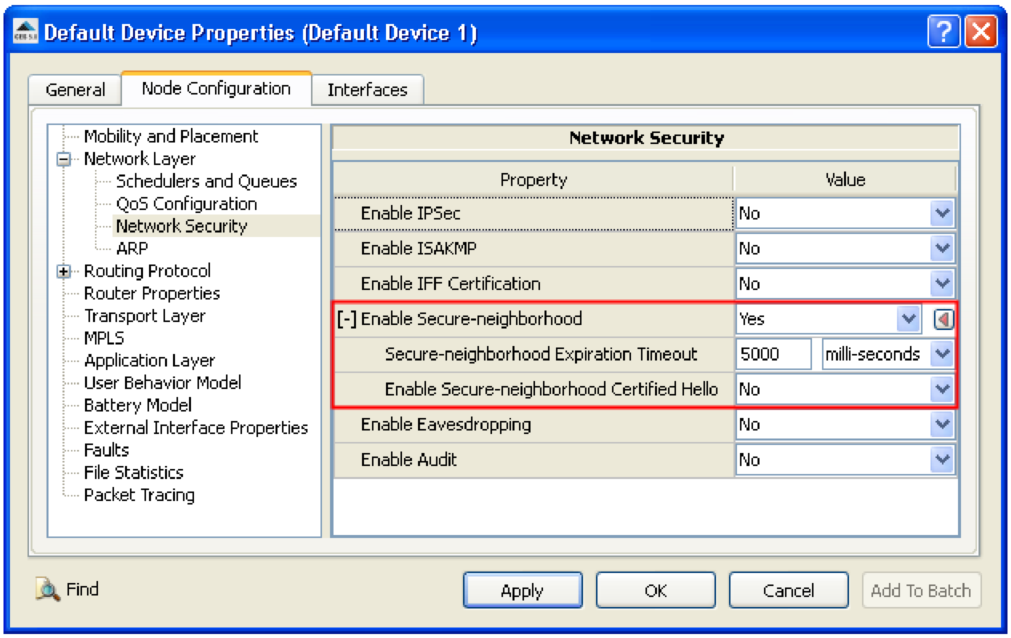


FIGURE 3-11. Setting Secure Neighbor Protocol

TABLE 3-33. Command Line Equivalent of Secure Neighbor Parameters

GUI Parameter	Scope of GUI Parameter	Command Line Parameter
Secure-neighborhood Expiration Timeout	Node	`SECURE-NEIGHBOR-TIMEOUT`
Enable Secure-neighborhood Certified Hello	Node	`SECURE-NEIGHBOR-CERTIFIED-HELLO`

Setting Parameters

- To enable certificate variant mode, set **Enable Secure-neighborhood Certified Hello** to *Yes.* Otherwise set **Enable Secure-neighborhood Certified Hello** to *No.*

3.7.6 Statistics

Table 3-34 lists the statistics collected for the Secure Neighbor model that are output to the statistics (.stat) file at the end of simulation.

TABLE 3-34. Secure Neighbor Statistics

Statistic	Description
Number of HELLO packets Initiated	Total number of Hello messages sent.
Number of bytes of HELLO packets Initiated	Total number of bytes of Hello messages sent.
Number of HELLO packets Received	Total number of Hello messages received.
Number of bytes of HELLO packets Received	Total number of bytes of Hello messages received.
Number of CHALLENGE packets Initiated	Total number of Challenge messages sent.
Number of bytes of CHALLENGE packets Initiated	Total number of bytes of Challenge messages sent.
Number of CHALLENGE packets Received	Total number of Challenge messages received.
Number of bytes of CHALLENGE packets Received	Total number of bytes of Challenge messages received.
Number of RESPONSE1 packets Initiated	Total number of Response1 messages sent.
Number of bytes of RESPONSE1 packets Initiated	Total number of bytes of Response1 messages sent.
Number of RESPONSE1 packets Received	Total number of Response1 messages received.
Number of bytes of RESPONSE1 packets Received	Total number of bytes of Response1 messages received.
Number of RESPONSE2 packets Initiated	Total number of Response2 messages sent.
Number of bytes of RESPONSE2 packets Initiated	Total number of bytes of Response2 messages sent.
Number of RESPONSE2 packets Received	Total number of Response2 messages received.
Number of bytes of RESPONSE2 packets Received	Total number of bytes of Response2 messages received.

3.7.7 Scenarios Included in EXata

The EXata distribution includes several sample scenarios for the Secure Neighbor model. All scenarios are located in the directory EXATA_HOME/scenarios/cyber/secure-neighbor. Table 3-35 lists the sub-directory where each scenario is located.

TABLE 3-35. Secure Neighbor Model Scenarios

Scenario Sub-directory	Description
snd-certified-hello	Shows the certificate variant of secure neighbor.
snd-handshaking	Shows the pair-wise shared secret and certificate variant of secure neighbor in one network.
snd-hello	Shows the pair-wise shared secret variant of secure neighbor.
snd-link-fault	Shows the secure neighbor table updation in case of link fault and link re-establishment.
snd-mixed	Shows the secure neighborhood behavior in a mixed network.
snd-mobility	Shows the overall behavior of secure neighborhood where mobility is encountered.

3.7.8 References

1. [HuPJ02] Yih-Chun Hu, Adrian Perrig, David B. Johnson, “Ariadne: A Secure On-demand Routing Protocol for Ad Hoc Networks”, pp.12-23, in Proceedings of The Eighth Annual International

Conference on Mobile Computing and Networking (MOBICOM), September 23-28, 2002. Atlanta, Georgia, USA.

2. [HuPJ03b] Yi-Chun Hu, Adrian Perrig, David B. Johnson, “Rushing Attacks and Defense in Wireless Ad Hoc Network Routing Protocols”, pp.30-40, ACM Wireless Security (WiSe'03), September 19, 2003. San Diego, California, USA, in conjunction with MobiCom, 2003.

4 Routing Protocol Models

This chapter describes features, configuration requirements and parameters, statistics, and scenarios for Routing Protocol Models in the Cyber Model Library, and consists of the following section:

- ANODR Model

4.1 Anonymous On-Demand Routing (ANODR) Protocol

The EXata ANODR model is based on publication of AODV and ANODR in the References section.

4.1.1 Description

Anonymous On-Demand Routing (ANODR) is designed to provide an a network-centric anonymous and untraceable routing scheme for mobile ad-hoc networks. It is based on table-driven AODV, and therefore any EXata simulation scenario using AODV can also use ANODR, instead to implement anonymous routing.

Privacy in mobile wireless networks uses different terminology than that traditionally used for banking systems and the wired Internet. In addition to traditional ideas of privacy, mobile privacy has concerns for the mobile node's identity, location, and motion patterns.

Anonymity issues are critical for ANODR scenarios, since allowing adversaries to trace network routes and infer the motion pattern of nodes at the end of those routes may pose serious threats to covert operations. This heightened privacy demand poses challenging constraints on routing and data forwarding. ANODR allows you to protect your mobile wireless communication from being traced, and without the necessity of removing your device's battery. ANODR provides the following security services:

1. Negligibility-based anti-tracing such that signal interceptors cannot trace signal transmitters mobility pattern via wireless signal tracing (with non-negligible probability defined on the victim network's size).
2. Confidentiality and anonymity.
3. Traffic flow confidentiality.
4. Identity-free routing.
5. One-time packet contents such that any two wireless transmissions are indistinguishable with each other in regard to a cryptanalyst.

These services are provided at the Network Layer and Link Layer to protect the IP and link layer protocols.

4.1.2 Features and Assumptions

This section describes the implemented features, omitted features, assumptions and limitations of the ANODR model.

4.1.2.1 Implemented Features

- On-demand routing
- Network layer identity-free control flow and data flow
- Anonymous virtual circuit establishment and maintenance (in routing table)

4.1.2.2 Omitted Features

- Pseudo-random route pseudonym update
- Link-layer identity-free control flow and data flow (because this requires modification of every link layer MAC protocol)
- Uniform packet size and neighborhood traffic mixing
- Support for IPv6

4.1.2.3 Assumptions and Limitations

- Network is using on-demand routing schemes

4.1.3 Supplemental Information

ANODR is a network layer protocol. Link layer must have its own anonymity support.

4.1.4 Command Line Configuration

To specify ANODR as the routing protocol, include the following parameter in the scenario configuration (.config) file:

```
[<Qualifier>]  ROUTING-PROTOCOL   ANODR
```

The scope of this parameter declaration can be Global, Node, Subnet, or Interface. See Section 1.2.1.1 for a description of `<Qualifier>` for each scope.

ANODR-specific Parameters

Table 4-1 lists the ANODR parameters specified in the scenario configuration (.config) file. See Section 1.2.1.3 for a description of the format used for the parameter table.

Note: All parameters in Table 4-1 are optional and can be configured at the global, node, subnet, and interface levels.

TABLE 4-1. ANODR-specific Parameters

Parameter	Value	Description
`ANODR-ACTIVE-ROUTE-TIMEOUT` Optional *Scope:* All	Time *Range:* `> 0` *Default:* `5000MS`	Specifies the expiry time for an active route; each time the route is used, the lifetime of that route is updated to this value.
`ANODR-NET-DIAMETER` Optional *Scope:* All	Integer *Range:* `> 0` *Default:* `35`	Specifies the maximum possible number of hops between two nodes in the network.
`ANODR-NODE-TRAVERSAL-TIME` Optional *Scope:* All	Time *Range:* `> 0` *Default:* `40MS`	Specifies the conservative estimate of the average one hop traversal time for packets and should include queuing, transmission, propagation and other delays.

TABLE 4-1. ANODR-specific Parameters

Parameter	Value	Description
`ANODR-BUFFER-MAX-PACKET` Optional *Scope:* All	Integer *Range:* `> 0` *Default:* `100`	Specifies the maximum number of packets the message buffer of ANODR can hold at any given time irrespective of packet size. If the buffer fills up, incoming packets for the buffer will be dropped.
`ANODR-BUFFER-MAX-BYTE` Optional *Scope:* All	Integer *Range:* `≥ 0` *Unit:* bytes *Default:* `0`	Specifies the maximum size of ANODR buffer in bytes. If zero is specified to this parameter, ANODR-BUFFER-MAX-PACKET will be used to determine the size of the buffer.

Examples of Parameter Usage

The following configurations enable ANODR in a subnet:

```
[ N8-192.0.0.0 ] ROUTING-PROTOCOL ANODR
[ N8-192.0.0.0 ] ANODR-NET-DIAMETER 35
[ N8-192.0.0.0 ] ANODR-NODE-TRAVERSAL-TIME 40MS
[ N8-192.0.0.0 ] ANODR-ACTIVE-ROUTE-TIMEOUT 5000MS
[ N8-192.0.0.0 ] ANODR-BUFFER-MAX-PACKET 100
[ N8-192.0.0.0 ] ANODR-BUFFER-MAX-BYTE 0
```

4.1.5 GUI Configuration

This section describes how to configure ANODR in the GUI.

4.1.5.1 General Configuration

To configure the ANODR parameters, perform the following steps:

1. Go to one of the following locations:
   - To set properties at the subnet level, go to **Wireless Subnet Properties Editor > Routing Protocol > General**.
   - To set properties at the point-to-point link, go to **Point-to-point Link Properties Editor > Point-to-point Link Properties > Routing Protocol**.
   - To set properties at the node level, go to **Default Device Properties Editor > Node Configuration > Routing Protocol**.
   - To set properties at the interface level, go to one of the following locations:
     - **Interface Properties Editor > Interfaces > Interface # > Routing Protocol**.
     - **Default Device Properties Editor > Interfaces > Interface # > Routing Protocol**.

   In this section, we show how to configure ANODR parameters for a specific node using the Default Device Properties Editor. Parameters can be set in the other properties editors in a similar way.

2. Set **Routing Protocol IPv4** to *ANODR* and set the dependent parameters listed in Table 4-2.

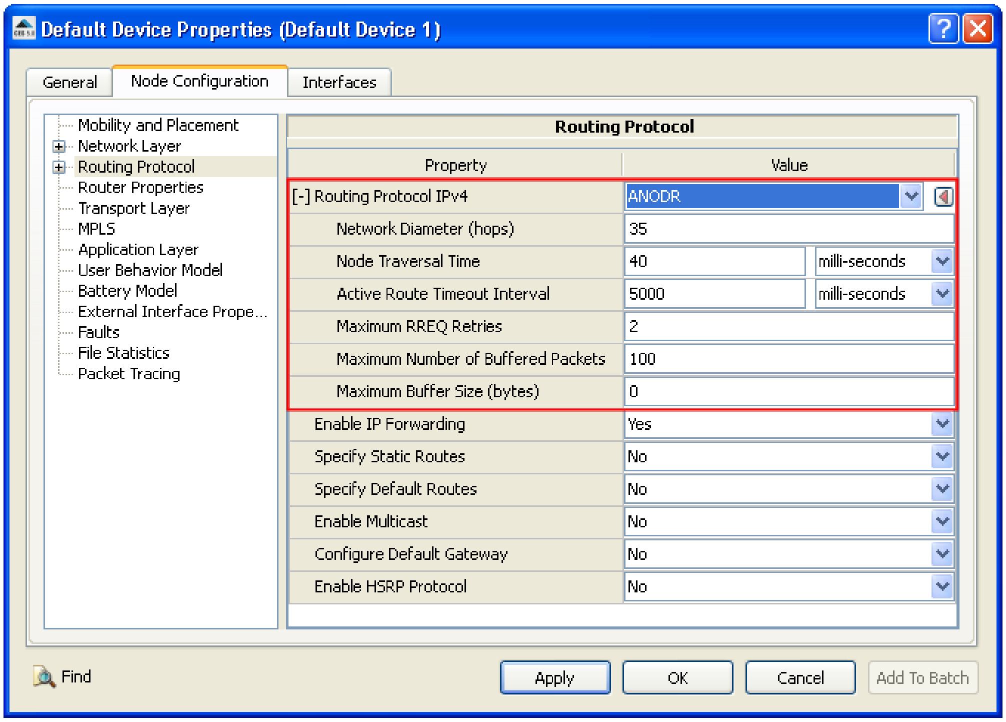


FIGURE 4-1. Setting ANODR as Routing Protocol

TABLE 4-2. Command Line Equivalent of ANODR General Parameters

GUI Parameter	Scope of GUI Parameter	Command Line Parameter
Network Diameter	Node, Subnet, Interface	ANODR-NET-DIAMETER
Node Traversal Time	Node, Subnet, Interface	ANODR-NODE-TRAVERSAL-TIME
Active Route Timeout Interval	Node, Subnet, Interface	ANODR-ACTIVE-ROUTE-TIMEOUT
Maximum RREQ Retries	Node, Subnet, Interface	ANODR-RREQ-RETRIES
Maximum Number of Buffered Packets	Node, Subnet, Interface	ANODR-BUFFER-MAX-PACKET
Maximum Buffer Size	Node, Subnet, Interface	ANODR-BUFFER-MAX-BYTE

4.1.6 Statistics

Table 4-3 lists the statistics collected for the ANODR model that are output to the statistics (.stat) file at the end of simulation.

TABLE 4-3. ANODR Statistics

Statistic	Description
Number of RREQ Initiated	Number of RREQ initiated for new connections.
Number of RREQ Retried	Number of RREQ re-initiated for existing connections.
Number of RREQ Forwarded	Number of RREQ forwarded as intermediate forwarder.
Number of RREQ Received	Number of any RREQ received.
Number of Duplicate RREQ Received	Number of duplicated RREQ received.
Number of RREQ Received by Dest	Number of RREQ received as destination.
Number of RREQ received by Dest with global trapdoor in symmetric key encryption	Number of RREQ received as destination and the RREQ is using efficient symmetric-key based global trapdoor.
Number of RREP Initiated as Dest	Number of RREP initiated.
Number of RREP Forwarded	Number of RREP forwarded as intermediate forwarder.
Number of RREP ACKed	Number of AACK initiated to ack RREP.
Number of RREP Received	Total Number of RREP received.
Number of RREP Received as Source	Number of RREP received as source.
Number of RREP-AACK Received	Total Number of RREP AACK received.
Number of RERR Initiated	Number of RERR initiated.
Number of RERR Forwarded	Number of RERR forwarded.
Number of RERR ACKed	Number of AACK initiated to ack RERR.
Number of RERR Received	Number of RERR received.
Number of RERR-AACK Received	Number of RERR AACK received.
Number of Data packets sent as Source	Number of data packets initiated.
Number of Data Packets Forwarded	Number of data packets forwarded.
Number of Data Packets Received	Number of data packets received.
Number of DATA-AACK Received	Number of DATA AACK received.
Number of Data Packets Dropped for no route	Number of data packets dropped because of having no route.
Number of Data Packets Dropped for buffer overflow	Number of data packet dropped because of being over the cache limit.
Number of times link broke	Number of link breakage detected.

4.1.7 Scenarios Included in EXata

The EXata distribution includes several sample scenarios for the ANODR protocol. All scenarios are located in the directory EXATA_HOME/scenarios/cyber/secure_routing/anodr. Table 4-4 lists the sub-directory where each scenario is located.

TABLE 4-4. ANODR Model Scenarios

Scenario Sub-directory	Description
anodr_buffer_test	Shows the functionality of buffer (ANODR-BUFFER-MAX-PACKET) in ANODR routing protocol when configured in a subnet
anodr_mi_test	Shows the functionality of ANODR routing protocol when configured in a subnet and used across multiple interfaces
anodr_mob_test	Shows the working of ANODR routing protocol when configured in a subnet with a mobile intermediate node
anodr_rerr_test	Shows the functionality of WTLS certification implementation in a wired scenario
anodr_route_timeout_test	Shows the functionality of route time out in ANODR routing protocol when configured in a subnet
anodr_rreq_rrep_test	Shows the functionality of RREQ and RREP in ANODR routing protocol when configured in a subnet

4.1.8 References

1. Jiejun Kong, Xiaoyan Hong, ANODR: ANonymous On Demand Routing with Untraceable Routes for Mobile Ad-hoc Networks, pp.291-302, The Fourth ACM International Symposium on Mobile Ad Hoc Networking and Computing (MobiHoc), Annapolis, Maryland, USA. June 1-3, 2003.
2. Jiejun Kong, Xiaoyan Hong, Mario Gerla. An Identity-free and On Demand Routing Scheme against Anonymity Threats in Mobile Ad-hoc Networks, Vol.6, No.8, pp.888-902, IEEE Transactions on Mobile Computing, August 2007.

5 Multi-layer Models

This chapter describes features, configuration requirements and parameters, statistics, and scenarios for Multi-layer Models in the Cyber Model Library, and consists of the following section:

- Adversary Model

5.1 Adversary Model

5.1.1 Description

The Adversary Model (also known as Threat Model, Attack Model, and Penetration Model) comprises an active adversary model “wormhole attacker” and a passive adversary model “eavesdropper”.

Active Threat (Wormhole attack)

Compared to jamming, wormhole attack is more covert in nature and harder to detect. The term “wormhole” refers to an adversary carrying information and traveling faster than anyone else, thus the adversary is capable of launching unusual timing attacks. While physical wormholes do not exist, communication wormholes do exist, because adversaries can forward packets faster than regular nodes that require a queuing delay, transmission delay, and MAC contention delay.

A wormhole attacker tunnels messages received in one location in the network over a low-latency high-bandwidth link and replays them in a different location. This typically requires at least two adversarial devices colluding to relay packets along a fast channel available only to the attackers, so that it can disrupt multi-hop ad hoc routing. In the presence of wormholes, the attacking nodes can selectively let routing control messages get through. Then, the wormhole link has a higher probability of being chosen as part of multi-hop routes due to its excellent packet delivery capability. Once the attacking nodes know they are en route, they can launch a *black hole* attack to drop all data packets, or a *gray hole* attack to selectively drop some critical packets.

In practice, single-hop wormholes (i.e., wormholes with both ends in the one-hop transmission range of the victim network), are typically ineffective because the wormholes cannot gain any timing advantage because of the science of physics. Recommended physical length of a wormhole link is between 1.2R and 2R where R is the nominal one-hop transmission range of the victim network. Such a wormhole link can gain significant timing advantage over a multi-hop forwarding path in the victim network. Moreover, victim network's turnaround time at the physical layer and the link layer must be properly estimated. EXata provides two configuration parameters, `WORMHOLE-VICTIM-COUNT-TURNAROUND-TIME` and `WORMHOLE-VICTIM-TURNAROUND-TIME`, for the user to specify such delay. In IEEE 802.11 standard, this turnaround time includes all delays between the time an 802.11 receiver receives RF signals and the time the same 802.11 device finishes transmitting the corresponding response. Typically, the turnaround time includes RxRFDelay (receiving radio signals and analog-digital conversion), RxPHYDelay (decoding, de-interleaving, descrambling), MAC processing delay, TxPHYDelay (scrambling, interleaving, encoding) and TxRFDelay (digital-analog conversion and transmitting radio signals). A secure version of any network protocol must also count cryptographic delays to implement message's data origin authentication.

A wormhole link may work in different modes of operation:

- *Transparent Mode* as external adversary: Wormhole devices are not regular network members. However, to make wormhole attack work, the adversary must be able to intercept legitimate wireless messages (assuming the wormhole attackers can thwart low-probability-interception mechanisms). Messages are *covertly* intercepted at one location and replayed at other locations while regular network members do not know the existence of wormhole devices. In other words, the existence of the wormhole devices is transparent to regular network nodes. A corresponding implementation uses layer-1 devices in the victim network and layer-2 devices in the attacking network to implement the wormhole devices.
- *Participant Mode* as internal adversary: Wormhole devices are regular network members. They are compromised nodes with legitimate network addresses like IP addresses and MAC addresses. A corresponding implementation uses layer-3 devices to implement the wormhole devices. Because

wormholes working in the transparent mode already significantly thwart victim network's routing functions, the participant mode is currently not implemented due to implementation redundancy.

Passive Threat (Eavesdrop)

Wireless traffic can be intercepted by any eavesdropping entity in the network, particularly, as mobile wireless nodes of the adversary. Each eavesdropper has an IP protocol stack. If needed, it can be an internal adversary/compromised node to participate in network functions. The eavesdropped packets are output to a file, the format of which is described in Section 5.1.6.

5.1.2 Features and Assumptions

This section describes the implemented features, omitted features, assumptions and limitations of the Adversary model.

5.1.2.1 Implemented Features

- (Multi-end) Wormhole network protocols including wormhole tunneling MAC as a contending bus, wormhole replaying MAC in an aggressive CSMA, queuing delays, transmission delays, propagation delays, prevention of infinite tunneling (i.e., do not tunnel wormhole-replayed packets, which have already been tunneled for at least once)
- Eavesdropping records (output as file contents)

5.1.2.2 Omitted Features

- Tunneling MAC in other forms
- Replay MAC in other forms
- Traffic analysis

5.1.2.3 Assumptions and Limitations

- Wormhole nodes can monitor victim nodes' RF signals and intercept victim's packets.

5.1.3 Supplemental Information

None.

5.1.4 Command Line Configuration

To enable the Wormhole model, include the following parameter in the scenario configuration (.config) file:

```
[<Qualifier>] MAC-PROTOCOL     MAC-WORMHOLE
```

To enable the Eavesdrop model, include the following parameter in the scenario configuration (.config) file:

```
[<Qualifier>] EAVESDROP-ENABLED    YES
```

The scope of these parameter declarations can be Global, Node, Subnet, or Interface. See Section 1.2.1.1 for a description of `<Qualifier>` for each scope.

Note: The default value of the parameter `EAVESDROP-ENABLED` is `NO`.

Configuration Requirements

There should be at least two nodes in a wormhole subnet.

Wormhole- Parameters

Table 5-1 lists the Wormhole parameters specified in the scenario configuration (.config) file. See Section 1.2.1.3 for a description of the format used for the parameter table.

TABLE 5-1. Wormhole Parameters

Parameter	Value	Description
WORMHOLE-MODE *Required* *Scope:* All	List: • THRESHOLD • ALLPASS • ALLDROP	Specifies the mode for the wormhole. THRESHOLD : Wormhole drops any packet with size greater than or equal to the threshold value. ALLPASS : Wormhole passes all packets irrespective of their size. ALLDROP : Wormhole drops all packets irrespective of their size.
WORMHOLE-THRESHOLD *Optional* *Scope:* All	Integer *Range:* ≥ 0 *Default:* 150 *Unit:* bytes	Specifies the threshold value for Wormhole. **Note:** This parameter is applicable only if WORMHOLE-MODE is set to THRESHOLD.
WORMHOLE-REPLAY-MAC-PROTOCOL *Required* *Scope:* All	String	Specifies the replay medium access protocol for the wormhole subnet. Currently only WORMHOLE-CSMA is supported.
WORMHOLE-LINK-BANDWIDTH *Required* *Scope:* All	Integer *Range:* > 0 *Unit:* bps	Specifies the wormhole link bandwidth for wormhole subnet.
WORMHOLE-PROPAGATION-DELAY *Optional* *Scope:* All	Time *Range:* > 0S *Default:* SIMULATION-TIME	Specifies the wormhole propagation delay for the wormhole subnet.

TABLE 5-1. Wormhole Parameters

Parameter	Value	Description
`WORMHOLE-VICTIM-COUNT-TURNAROUND-TIME` *Optional* *Scope:* All	List: • `YES` • `NO` *Default:* `NO`	The victim network may have actual physical layer delay and link layer delay that is not counted. For example, to resist forgery of RTS/CTS packets in 802.11 network, full packet authentication on every packet must be implemented. This incurs extra cryptographic latency that should be counted in turnaround time.
`WORMHOLE-VICTIM-TURNAROUND-TIME` *Optional* *Scope:* All	Time *Range:* `> 0S` *Default:* `0S`	Specifies the turnaround time for the victim subnets. This value has critical impact on the network's behavior under wormhole attacks. **Note:** This parameter is applicable only if `WORMHOLE-VICTIM-COUNT-TURNAROUND-TIME` is set to `YES`.

5.1.5 GUI Configuration

This section describes how to configure the Wormhole and Eavesdrop models in the GUI.

5.1.5.1 Configuring Wormhole Parameters

To configure the Wormhole parameters, perform the following steps:

1. Go to one of the following locations:

- To set properties at subnet level, go to the **Wireless Subnet Properties Editor > MAC Layer**.
- To set properties at interface level, go to one of the following locations:
  - **Interface Properties Editor > Interfaces > Interface # > MAC Layer**.
  - **Default Device Properties Editor > Interfaces > Interface # > MAC Layer**.

In this section, we show how to configure the general Wormhole parameters in the Wireless Subnet Properties editor. Parameters can be set in the other properties editors in a similar way.

2. Set **MAC Protocol** to *Wormhole* and set the dependent parameters listed in Table 5-2.

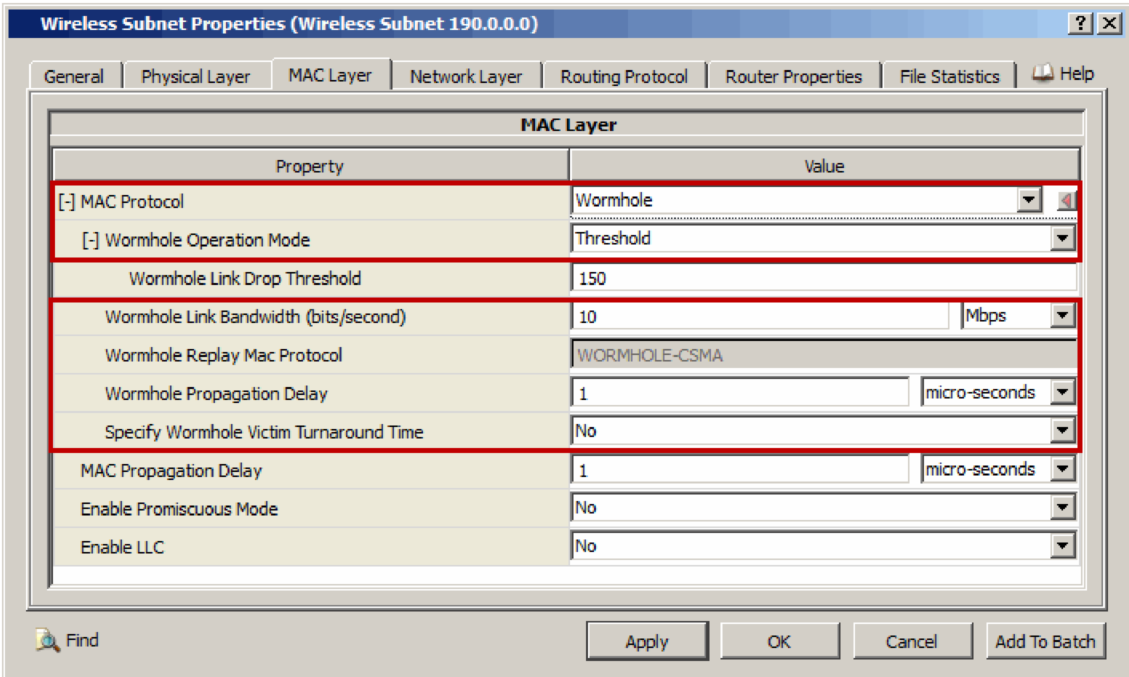


FIGURE 5-1. Setting Wormhole Parameters

TABLE 5-2. Command Line Equivalent of Wormhole Parameters

GUI Parameter	Scope of GUI Parameter	Command Line Parameter
Wormhole Operation Mode	Subnet, Interface	`WORMHOLE-MODE`
Wormhole Link Bandwidth	Subnet, Interface	`WORMHOLE-LINK-BANDWIDTH`
Wormhole Replay Mac Protocol	Subnet, Interface	`WORMHOLE-REPLAY-MAC-PROTOCOL`
Wormhole Propagation Delay	Subnet, Interface	`WORMHOLE-PROPAGATION-DELAY`
Specify Wormhole Victim Turnaround Time	Subnet, Interface	`WORMHOLE-VICTIM-COUNT-TURNAROUND-TIME`

Setting Parameters

- To enable the THRESHOLD mode, set **Wormhole Operation Mode** to *Threshold.*
- To enable the ALLPASS mode, set **Wormhole Operation Mode** to *All Pass.*
- To enable the ALLDROP mode, set **Wormhole Operation Mode** to *All Drop.*

3. If **Wormhole Operation Mode** is set to *Threshold,* set the dependent parameters listed in Table 5-3.

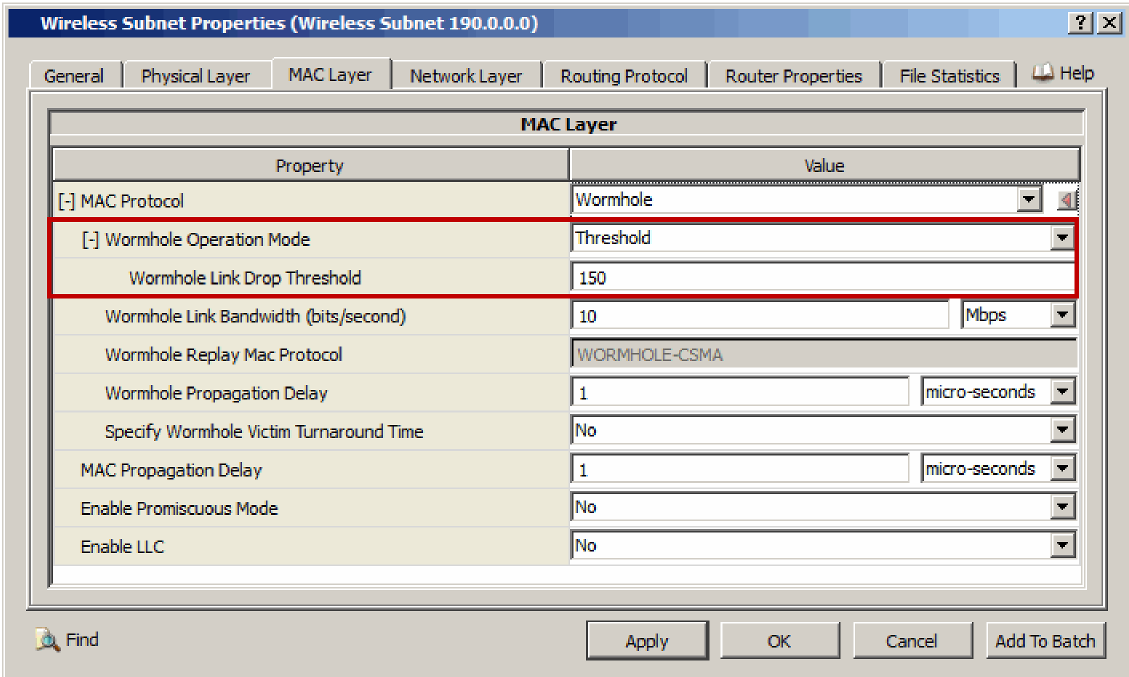


FIGURE 5-2. Setting Wormhole Threshold Operation Mode Parameters

TABLE 5-3. Command Line Equivalent of Wormhole Threshold Mode Parameters

GUI Parameter	Scope of GUI Parameter	Command Line Parameter
Wormhole Link Drop Threshold	Subnet, Interface	`WORMHOLE-THRESHOLD`

4. If **Specify Wormhole Victim Turnaround Time** is set to *Yes,* set the dependent parameters listed in Table 5-4.

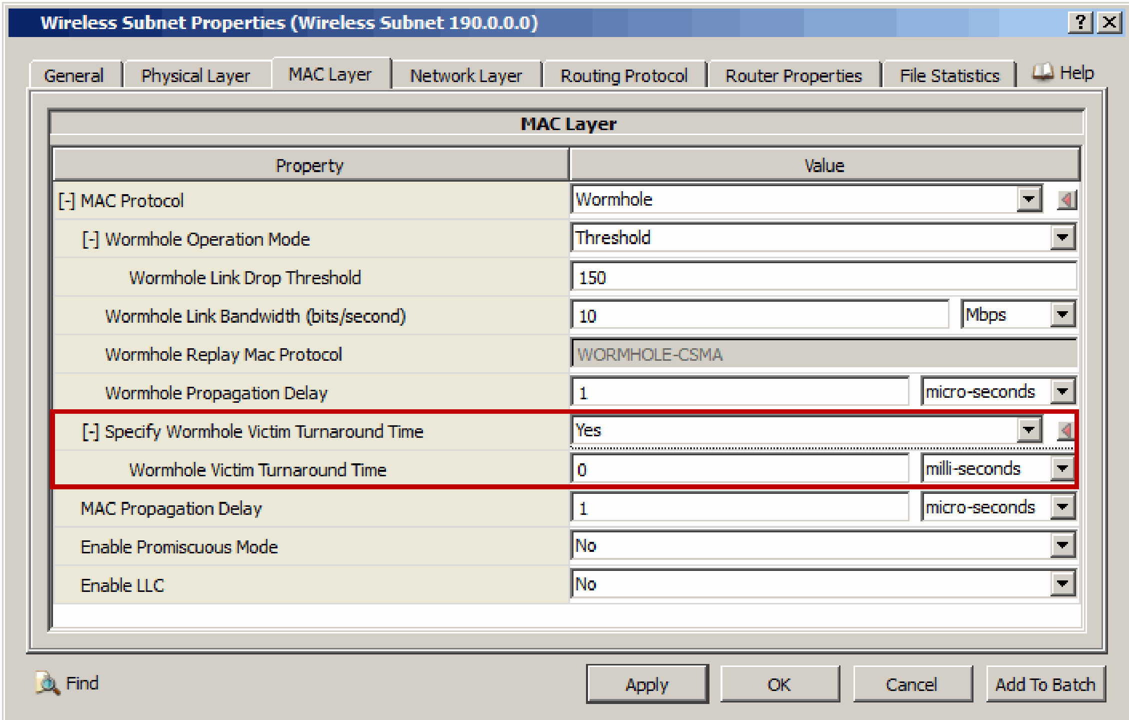


FIGURE 5-3. Setting Victim Turnaround Time

TABLE 5-4. Command Line Equivalent of Wormhole Victim Turnaround Time Parameters

GUI Parameter	Scope of GUI Parameter	Command Line Parameter
Wormhole Victim Turnaround Time	Subnet, Interface	`WORMHOLE-VICTIM-TURNAROUND-TIME`

5.1.5.2 Configuring Eavesdrop Parameters

To configure the general Eavesdrop parameters, perform the following steps:

1. Go to one of the following locations:

- To set properties at node level, go to **Default Device Properties Editor > Node Configuration > Network Layer > Cyber**.
- To set properties at wireless subnet level, go to **Wireless Subnet Properties Editor > Network Layer > Cyber**.
- To set properties at interface level, go to one of the following locations:
  - **Interface Properties Editor > Interfaces > Interface # > Network Layer > Cyber**.
  - **Default Device Properties Editor > Interfaces > Interface # > Network Layer > Cyber**.

In this section, we show how to configure the Eavesdrop parameters in the Wireless Subnet Properties Editor. Parameters can be set in the other properties editors in a similar way.

2. Set **Enable Eavesdropping** to *Yes.*

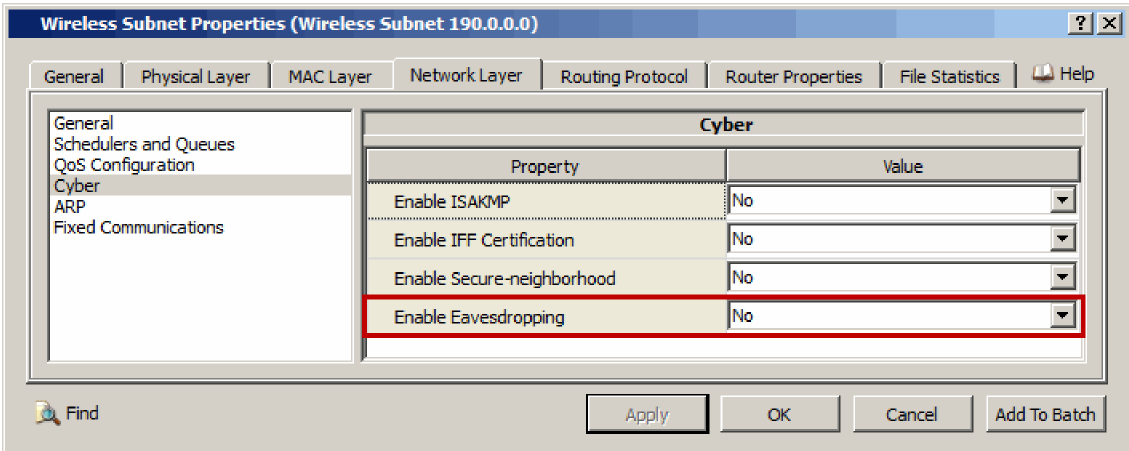


FIGURE 5-4. Enabling Eavesdropping

TABLE 5-5. Command Line Equivalent of Eavesdropping Parameters

GUI Parameter	Scope of GUI Parameter	Command Line Parameter
Enable Eavesdropping	Subnet, Interface	`EAVESDROP-ENABLED`

5.1.6 Statistics and Output

Wormhole Statistics

Table 5-6 lists the statistics collected for the Wormhole that are output to the statistics (.stat) file at the end of simulation.

TABLE 5-6. Wormhole Statistics

Statistic	Description
Frames intercepted all	Number of frames intercepted by the wormhole node.
Frames dropped by wormhole	Number of frames dropped by the wormhole link (since the frames are classified as data packets, for example, with packet size greater than a threshold).
Frames tunneled	Number of frames tunneled by the wormhole node (frames intercepted multiple times due to repetitive replay will not be tunneled.)
Frames replayed	Number of frames replayed by the wormhole node
Frames dropped by queue	Number of frames dropped by the queue in the wormhole node

Eavesdrop Output

Eavesdrop does not print any statistics to the statistics (.stat) file. Instead a file is generated for each interface that records the eavesdropped packets. The file for an interface is named "default.eavesdrop.<interface-address>". The output file contains the following information, which is explained in Table 5-7:

```
time: ip_v ip_hl ip_tos ip_len ip_id
flags ip_reserved ip_dont_fragment ip_more_fragments
ip_fragment_offset ip_ttl ip_p ip_sum ip_src ip_dst
```

TABLE 5-7. Eavesdrop Output

Output Field	Description
Time	
ip_v	IP Version 4
ip_hl	IP Header
ip_tos	IP type of services
ip_len	Total length of the IP header
Ip_id	IP identification
Flags	
ip_reserved	To distinguish SDR control packets
ip_dont_fragment	To handle fragmentation/offset whenever needed
ip_more_fragments	To handle fragmentation/offset whenever needed
ip_fragment_offset	To handle fragmentation/offset whenever needed
ip_ttl	IP time to live
ip_p	Transport protocol
ip_sum	Checksum
ip_src	Source IP
ip_dst	Destination IP

5.1.7 Sample Scenarios

5.1.7.1 Wormhole Sample Scenario

5.1.7.1.1 Scenario Description

In the sample scenario shown in Figure 5-5, nodes 1 and 3 are connected to a wireless subnet. Nodes 5 and 6 are connected through another wireless subnet. Nodes 2 and 4 are wormhole nodes connected to a subnet. Wormhole is enabled on the subnet. One CBR application is configured from node 1 to node 6. 100 packets are sent from node 1 to node 6.

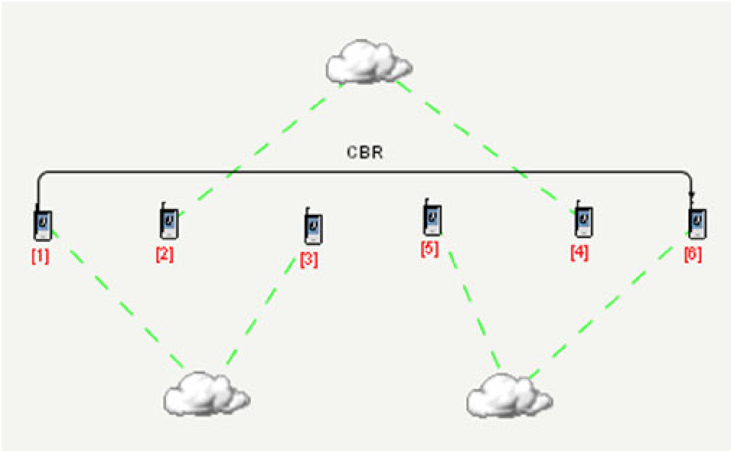


FIGURE 5-5. Wormhole Sample Scenario

5.1.7.1.2 Command Line Configuration

Include the following lines in the scenario configuration (.config) file:

```
# Nodes are placed and connected through these wireless subnets
SUBNET N8-192.0.0.0 {2 4}
SUBNET N8-192.0.1.0 {5 6}
SUBNET N8-192.0.2.0 {1 3}

# At Subnet level: Wormhole is configured as follows:
[N8-192.0.0.0] MAC-PROTOCOL MAC-WORMHOLE
[N8-192.0.0.0] WORMHOLE-MODE THRESHOLD
[N8-192.0.0.0] WORMHOLE-THRESHOLD 100
[N8-192.0.0.0] WORMHOLE-REPLAY-MAC-PROTOCOL WORMHOLE-CSMA
[N8-192.0.0.0] WORMHOLE-LINK-BANDWIDTH 100000000
[N8-192.0.0.0] WORMHOLE-PROPAGATION-DELAY 2US
WORMHOLE-VICTIM-COUNT-TURNAROUND-TIME YES
WORMHOLE-VICTIM-TURNAROUND-TIME 1MS
```

Include the following line in the application configuration (.app) file.

```
CBR 1 6 100 512 1S 1S 0 PRECEDENCE 0
```

5.1.7.1.3 GUI Configuration

Perform the following steps to create this sample scenario using the GUI:

1. Place six nodes of the Default device type and three wireless subnets on the canvas. Connect all the nodes to the corresponding wireless subnet as shown in the FIGURE 2-4.
2. To set MAC PROTOCOL for second Subnet (nodes 2 and 5), go to **MAC Layer** tab of Wireless Subnet Properties Editor and set **MAC Protocol to Wormhole** as shown in the Figure 5-1, “Setting Wormhole Parameters,” on page 108.
3. Create CBR application between node 1 and node 6.

5.1.7.2 Eavesdrop Sample Scenario

5.1.7.2.1 Scenario Description

In the sample scenario shown in Figure 5-6, nodes 1, 3 and 5 are connected to a wireless subnet. Nodes 2 and 4 are eavesdrop enabled nodes connected to a different subnet. One CBR application is configured from node 1 to node 5.

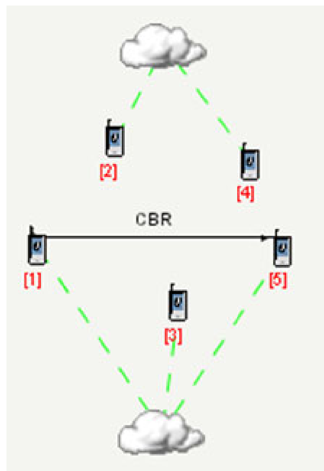


FIGURE 5-6. Eavesdrop Sample Scenario

5.1.7.2.2 Command Line Configuration

Include the following lines in the scenario configuration (.config) file:

```
# Nodes are placed and connected through these wireless subnets
SUBNET N8-192.0.0.0 {1 3 5}
SUBNET N8-192.0.1.0 {2 4}

# At Node level: Eavesdrop is enabled as follows:
[2 4] EAVESDROP-ENABLED YES
```

5.1.7.2.3 GUI Configuration

Perform the following steps to create this sample scenario using the GUI:

1. Place five nodes of the Default device type and two wireless subnets on the canvas. Connect all the nodes to the corresponding wireless subnet as shown in the Figure 5-6.
2. To set Eavesdrop for the second subnet, go to **Wireless Subnet Properties Editor > Network Layer > Cyber** and set **Enable Eavesdropping** to *Yes* as shown in the Figure 5-1.
3. Create CBR application between node 1 and node 5.

5.1.8 Scenarios Included in EXata

The EXata distribution includes several sample scenarios for the Adversary Model. All scenarios are located in the directory EXATA_HOME/scenarios/cyber/adversary. Table 5-8 lists the sub-directory where each scenario is located.

TABLE 5-8. Adversary Model Scenarios

Scenario Sub-directory	Description
prevent_infinite_tunneling	Shows the prevention of infinite tunneling of packets by the wormhole nodes.
wormhole_alldrop	Shows how wormhole drops all packets including both Control packets and Data packets.
wormhole_allpass	Shows how wormhole passes all packets including both Control packets and Data packets.
wormhole_propagation_delays	Shows the impact of a longer propagation delay on the wormhole link.
wormhole_replay	Shows the wormhole replay function with all packets going through the wormhole link.
wormhole_threshold	Shows the wormhole tunneling function with a user-defined threshold value (72 bytes in this case).
wormhole_tunneling	Shows the wormhole tunneling function with all packets tunneled through the wormhole link.

5.1.9 References

1. [RFC3561] C. Perkins, E. Belding-Royer, S. Das. “Ad hoc On-Demand Distance Vector (AODV) Routing.” July 2003.
2. [JohnsonM03] David B. Johnson, David A. Maltz, “The Dynamic Source Routing Protocol for Mobile Ad Hoc Networks (DSR)”, Internet-Draft, draft-ietf-manet-dsr-09.txt, April, 2003.
3. [SanzgiriDLSR02] Kimaya Sanzgiri, Bridget Dahill, Brian Neil Levine, Clay Shields, Elizabeth Royer, “A Secure Routing Protocol for Ad Hoc Networks”, pp.78-89, in Proceedings of The Tenth IEEE International Conference on Network Protocols (ICNP), 2002. November 12-15. Paris, France.
4. [HuPJ02] Yih-Chun Hu, Adrian Perrig, David B. Johnson, “Ariadne: A Secure On-demand Routing Protocol for Ad Hoc Networks”, pp.12-23, in Proceedings of The Eighth Annual International Conference on Mobile Computing and Networking (MOBICOM), September 23-28, 2002. Atlanta, Georgia, USA.
5. [HuPJ03a] Yih-Chun Hu, Adrian Perrig, David B. Johnson, “Packet Leashes: A Defense against Wormhole Attacks in Wireless Networks”, in Proceedings of The 22nd IEEE INFOCOM, March 30-April 3, 2003. San Francisco, California, USA.
6. [HuPJ03b] Yi-Chun Hu, Adrian Perrig, David B. Johnson, “Rushing Attacks and Defense in Wireless Ad Hoc Network Routing Protocols”, pp.30-40, ACM Wireless Security (WiSe'03), September 19, 2003. San Diego, California, USA, in conjunction with MobiCom 2003.

6 Attack Models

This chapter describes features, configuration requirements and parameters, statistics, and scenarios for Attack Models in the Cyber Model Library, and consists of the following sections:

- Denial of Service (DOS) Attack Model
- Signal Intelligence (SIGINT) Attack Model
- Virus Attack Model
- Wireless Eavesdropping Attack Model
- Wireless Jamming Attack Model

6.1 Initiating Attacks from EXata GUI

Attacks can be launched from the Human-In-The-Loop (HITL) interface of the EXata GUI. For details of using the EXata GUI, refer to *EXata/Cyber User's Guide*.

To launch an attack, do the following:

1. Open a scenario in EXata Architect. (The scenario can be created using the command line interface or the Design mode of Architect.)
2. Press the **Run Simulation**  button to initialize the scenario.

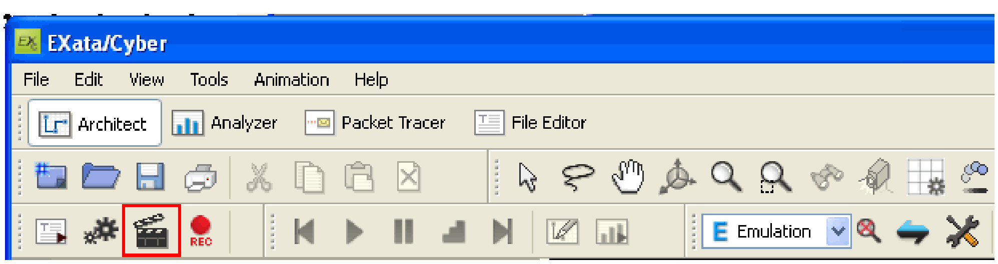


FIGURE 6-1. Run Simulation Button

This changes the Architect mode from Design to Visualize mode.

3. Click on the **Human in the Loop** button at the bottom of the **Visualization Controls** panel.

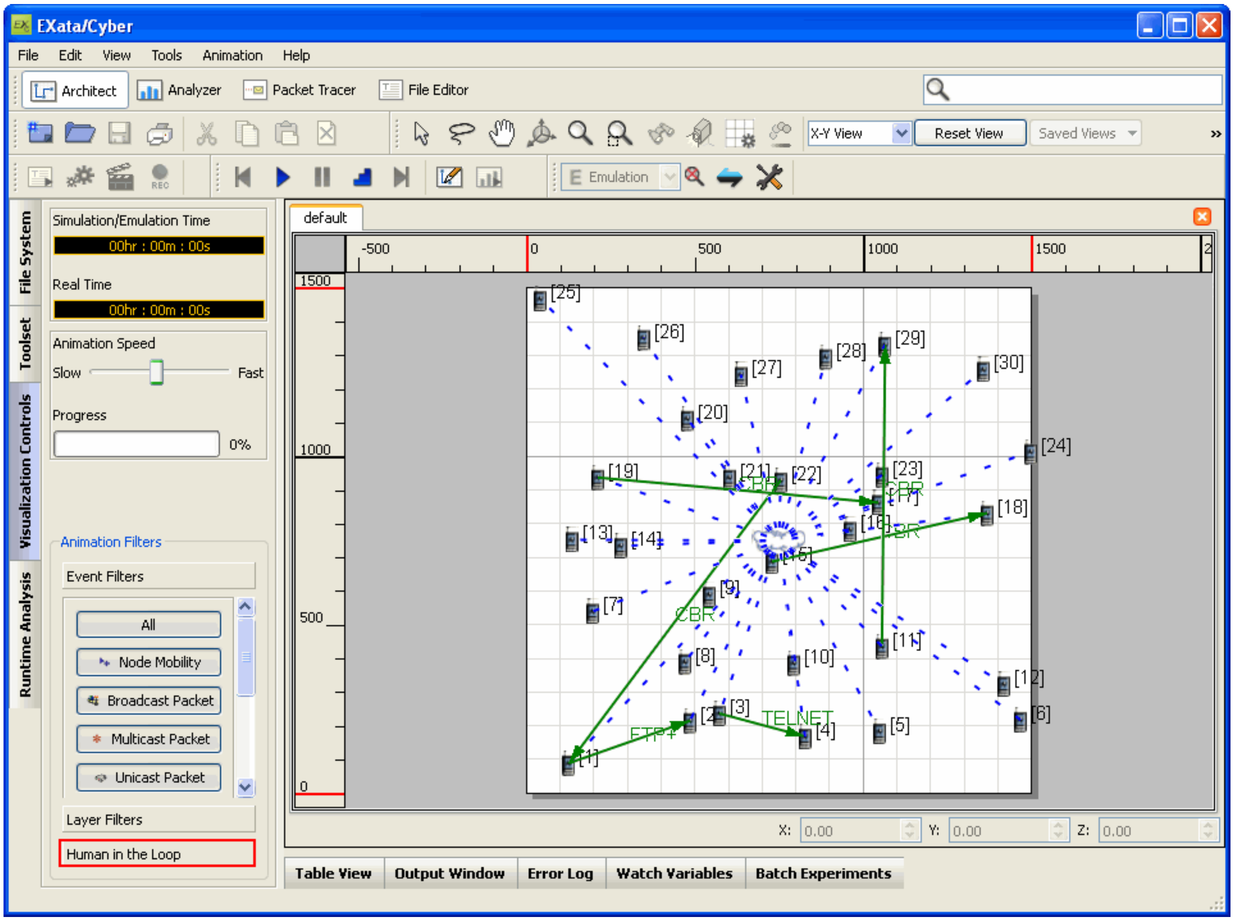
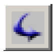


FIGURE 6-2. Visualization Controls Panel

4. The Human-In-The-Loop (HITL) interface is used to send commands to the simulator over the socket while the scenario is running. To send a command to the simulator, enter it in the text box and press the  button.

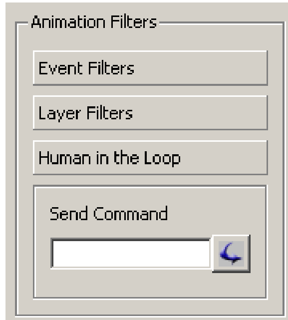


FIGURE 6-3. Human in the Loop Interface

HITL command for EXata attack models are described in the following sections. For additional HITL commands, refer to Chapter 6 of *EXata/Cyber User's Guide.*

6.2 Denial of Service (DOS) Attack Model

6.2.1 Description

A Denial-of-Service (DOS) attack is the act of overwhelming the resources of a victim computer or network so that the victim cannot service requests from other clients. The clients, therefore, are denied service from the victim computer or network. The DOS attack typically targets the memory and/or computational resources of the victim computer by sending a large volume of traffic.

The DOS Attack model in EXata supports three kinds of attacks:

- Basic: This is where the attacker(s) send large volume of UDP traffic to the victim host or network. This traffic consumes network buffer memory as well as CPU resources.
- TCP SYN: This is where the attacker(s) send TCP SYN packets to the victim computer. Each TCP SYN packet opens a new TCP connection at the victim computer, thus consuming the transport layer buffer memory.
- IP Fragmentation: This is where the attacker(s) send partially fragmented IP packets to the victim computer. The victim computer buffers these fragmented packets and wait for remaining segments, thus consuming the network layer buffer memory.

6.2.2 Features and Assumptions\

This section describes the implemented features, omitted features, assumptions and limitations of the DOS Attack model.

6.2.2.1 Implemented Features

- Denial of service attacks.
- Attack traffic that can be UDP data stream, TCP SYN packets, or fragmented IP packets.

6.2.2.2 Omitted Features

None.

6.2.2.3 Assumptions and Limitations

None.

6.2.3 Command Line Configuration

To configure a DOS attack, include the following statement in the application configuration (.app) file:

```
DOS <victim> <num-of-attacker> <attacker-1> .... <attacker-N>
    <attack-type> <victim port> <item-count> <item-size> <interval>
    <start-time> <end-time>
```

Note: All parameters should be entered on the same line.

Table 6-1 lists the configuration parameters for the DOS Attack model. See Section 1.6.1.3 for a description of the format used for the parameter table.

TABLE 6-1. DOS Model Parameters

Parameters	Value	Description
`<victim>` *Required*	Integer or IP Address	Victim node's ID or IP address.
`<num-of-attackers>` *Required*	Integer *Range:* `>0`	Number of attackers.
`<attacker1> <attacker2>…….<attackerN>` *Required*	List of integers	Space-separated list of node IDs of attackers. Example: 1 4 10 25
`<attack-type>` *Required*	List • `BASIC` • `SYN` • `FRAG`	Type of DOS attack traffic. `BASIC`: Sends UDP traffic and consumes the network buffer memory and CPU resources. `SYN`: Sends TCP SYN packets and consumes the Transport layer memory. `FRAG`: Sends IP fragments and consumes the Network layer buffer memory
`<victim-port>` *Required*	Integer *Range:* `[0, 65535]`	The port number at victim node to which the DOS traffic is sent. **Note:** This parameter is ignored if `<attack-type>` is set as `FRAG`.
`<items-count>` *Required*	*Integer :* *Range:* ≥ `0`	Number of packets to send. If this is set to 0, items will be sent continually until `<end-time>` or until the end of the simulation, whichever comes first. **Note:** If `<items-count>` and `<end-time>` are both greater than 0, packets are transmitted until `<items-to-send>` packets have been sent, `<end-time>` is reached, or the simulation ends, whichever comes first.
`<item-size>` *Required*	*Integer* *Range:* `[32, 65023]`	Size of each item. **Note:** This parameter is ignored if `<attack-type>` is set as `SYN`.
`<interval>` *Required*	Time *Range:* `> 0S`	Time between transmissions of successive packets (inter-departure time).
`<start-time>` *Required*	Time *Range:* ≥ `0S`	Time when the transmission of packets should begin.

TABLE 6-1. DOS Model Parameters (Continued)

Parameters	Value	Description
`<end-time>` *Required*	Time *Range:* ≥ `0S`	Time when the transmission of packets should end. If this is set to 0, transmission ends after `<items-to-send>` packets have been sent or until the end of simulation, whichever comes first. **Note:** `<end-time>` should be 0 or greater than `<start-time>`. If `<items-count>` and `<end-time>` are both greater than 0, packets are transmitted until `<items-to-send>` packets have been sent, `<end-time>` is reached, or the simulation ends, whichever comes first.

Examples of Parameter Usage

The following are examples of DOS attack configuration:

1. One attacker (node 15) attacks victim node (node 10) using SYN DOS attack mode.

```
DOS 10 1 15 SYN 80 10000 512 10MS 10S 20S
```

2. Five attackers (nodes 11 through 15) attack victim node (node 10) using BASIC DOS attack mode.

```
DOS 10 5 11 12 13 14 15 BASIC 1234 0 512 10MS 10S 20S
```

6.2.4 GUI Configuration

To configure a DOS attack, perform the following steps:

1. Click the **DOS** button in the **Cyber Attacks** tab of the Standard Toolset.
2. On the canvas, click on the node that is to be the victim of the attack.
3. Open the DOS Properties Editor by doing one of the following:
   a. Right-click on the symbol next to the victim node on the canvas and select **Properties** from the menu.
   b. In the **Applications** tab of Table View either double-click on the DOS application row or right-click on the application row and select **Properties** from the menu.

4. Set the parameters listed in Table 6-2.

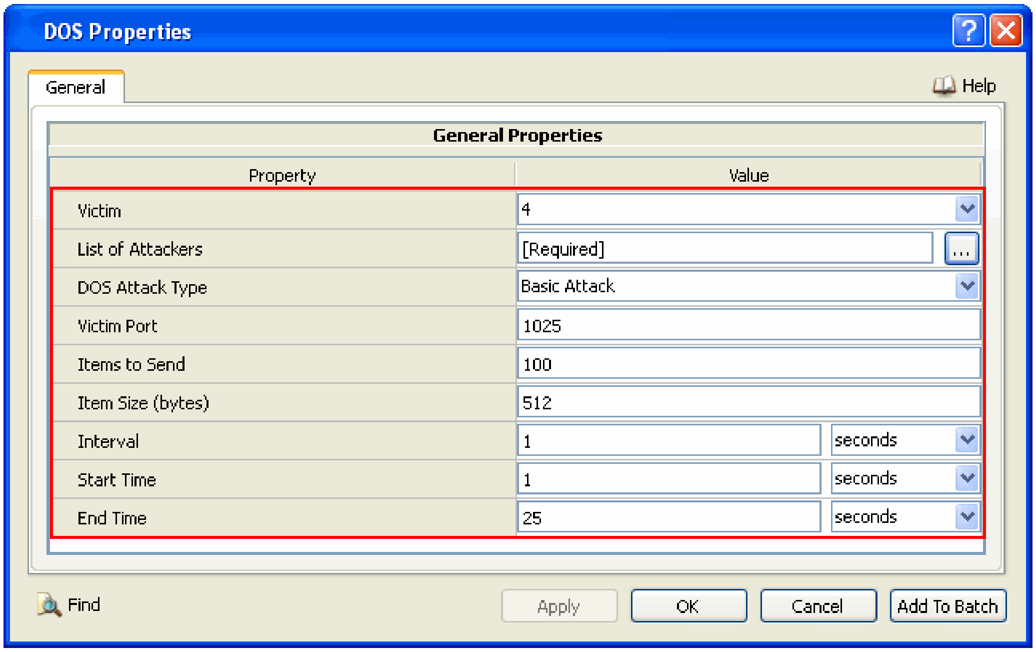


FIGURE 6-4. Setting the DOS Parameters

TABLE 6-2. Command Line Equivalent of DOS Parameters

GUI Parameter	Command Line Parameter
Victim	`<victim>`
List of Attackers	`<attacker1> <attacker2>.......<attackerN>`
DOS Attack Type	`<attack-type>`
Victim Port	`<victim-port>`
Items to Send	`<items-count>`
Item Size	`<item-size>`
Interval	`<interval>`
Start Time	`<start-time>`
End Time	`<end-time>`

5. To specify the attacker nodes, do the following:

   a. Click the **Select Nodes** ... button in the **Value** column of **List of Attackers**. This opens the dialog to enter node IDs (Figure 6-5).

   b. Enter the node IDs of the attacker nodes.

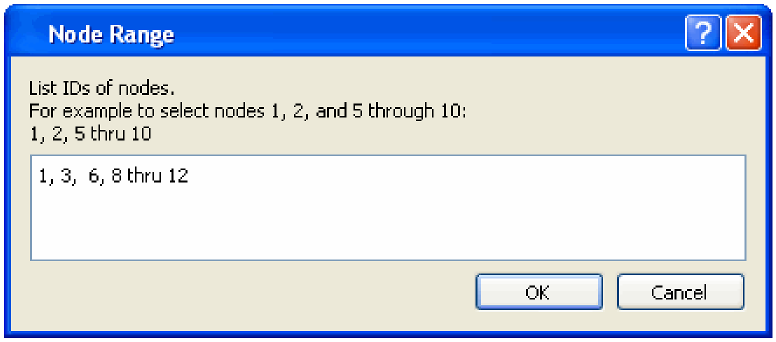


FIGURE 6-5. Selecting Attacker Nodes

6.2.5 Runtime Commands for DOS Model

This section describes how to launch and terminate DOS attacks from the Human-in-the-loop interface of EXata GUI (see Section 6.1).

Launching Denial of Service Attacks at Runtime

To launch a DOS attack, execute the following command from the HITL interface:

```
dos <victim> <num-of-attackers> [<attacker-1> .... <attacker-N>]
    <attack-type> <victim port> <interval> <duration>
```

where

`<victim>`	Victim node's ID or IP address.
`<num-of-attackers>`	Number of attackers.
`<attacker-1> .... <attacker-N>`	Space-separated list of node IDs of attackers. Example: 1 4 10 25 **Note:** The specification of attacker Node IDs may be omitted; in which case, the model randomly selects `<num-of-attackers>` count of attackers.
`<attack-type>`	Type of DOS attack traffic. `BASIC`: Sends UDP traffic and consumes the network buffer memory and CPU resources. `SYN`: Sends TCP SYN packets and consumes the Transport layer memory. `FRAG`: Sends IP fragments and consumes the Network layer buffer memory
`<victim port>`	The port number at victim node to which the DOS traffic is sent. **Note:** This parameter is ignored if `<attack-type>` is set as `FRAG`.

`<interval>`	Time between transmissions of successive packets (inter-departure time).
`<duration>`	Duration of the DOS attack. (The attack starts as soon as the command is sent.)

Terminating Denial of Service Attacks at Runtime

To terminate all DOS attacks on a victim node, execute the following command from the HITL interface:

```
stop dos <victim>
```

where

`<victim>`	Node ID or IP address of the victim node.

6.2.6 Statistics

Table 6-3 lists the DOS model statistics that are output to the statistics (.stat) file at the end of simulation.

TABLE 6-3. DOS Model Statistics

Statistic	Description
First packet sent at (sec)	The time instance when the first DOS packet was sent.
Last packet sent at (sec)	The time instance when the last DOS packet was sent.
Number of UDP packets sent	Number of UDP packets sent by the attacker. This statistic is reported for the BASIC attack type only.
Number of TCP SYN packets sent	Number of TCP SYN packets sent by the attacker. This statistic is reported for the SYN attack type only.
Number of IP Fragment packets sent	Number of IP Fragment packets sent by the attacker. This statistics is reported for the FRAG attack type only.

6.2.7 Scenarios Included in QualNet

The EXata distribution includes several sample scenarios for the DOS model. All scenarios are located in the directory EXATA_HOME/scenarios/cyber/dos. Table 6-4 lists the sub-directory where each scenario is located.

TABLE 6-4. DOS Scenarios Included in EXata

Scenario	Description
dos_basic_attack	Shows the DOS basic attack capability.
dos_frag_attack	Shows the DOS frag attack capability.
dos_syn_attack	Shows the DOS syn attack capability.

6.2.8 References

None.

6.3 Signals Intelligence (SIGINT) Attack Model

6.3.1 Description

Signals Intelligence (SIGINT) is an act of gathering information by intercepting and analyzing the signals. No attempt is made to decode the signal (the form of intelligence that gathers information by decoding signals is called Communications Intelligence). Only the characteristics of signals, such as frequency range, power of transmission, RF signatures, etc., are determined.

The SIGINT model in EXata provides a basic framework and API upon which advanced intelligence gathering algorithms may be developed. The SIGINT model itself reports the following information for each signal it detects:

- Channel frequency
- Received signal power
- Direction of arrival (azimuth and elevation angles)
- Time of transmission

The SIGINT model in EXata supports the following strategies of frequency scanning for SIGINT:

- Wideband SIGINT: The scanner scans all transmissions in a specified frequency range.
- Sweep SIGINT: The scanner divides the frequency range in contiguous blocks of sweep bandwidth each. The scanner scans each frequency block at a time for the sweep slot duration before moving to next frequency block. The sweep order can be either sequential or random.
- Custom SIGINT: This mode offers greatest configuration control over the frequency ranges and scanning patterns that must be scanned. The scanner scans frequency ranges according to the scanning patterns specified in a file.

6.3.2 Features and Assumptions

This section describes the implemented features, omitted features, assumptions and limitations of the SIGINT model.

6.3.2.1 Implemented Features

- Wideband scanning
- Sequential and Random Sweep scanning
- Custom frequency scanning.
- Commands to launch and terminate the SIGINT model during runtime

6.3.2.2 Omitted Features

- Cross-channel and partial-channel overlap scanning.

6.3.2.3 Assumptions and Limitations

- At any time, only one SIGINT instance can be active on a given interface of a node.

6.3.3 Command Line Configuration

Table 6-5 lists the SIGINT model parameters that are specified in the scenario configuration (.config) file. See Section 1.2.1.1 for a description of the format used for the parameter table.

TABLE 6-5. SIGINT Model Scenario Configuration File Parameters

Parameter	Value	Description
SIGINT-SCANNER-TYPE *Optional* *Scope:* Global *Instances:* scanner index	List • BASIC • SWEEP • FILE *Default:* BASIC	Specifies how the frequencies are selected for scanning as follows: BASIC: Selects a wideband range of frequency for scanning. SWEEP: Sweeps across a frequency range in small blocks of bandwidth at a time. FILE: Uses a custom strategy of frequency selection for scanning.
SIGINT-START-FREQUENCY *Optional* *Scope:* Global *Instances:* scanner index	Integer *Range:* > 0 *Unit:* Hz	Specifies the lower end of frequency range selected for scanning. **Note:** This parameter is required if SIGINT-SCANNER-TYPE is set to BASIC or SWEEP.
SIGINT-END-FREQUENCY *Optional* *Scope:* Global *Instances:* scanner index	Integer *Range:* > 0 *Unit:* Hz	Specifies the upper end of the frequency range selected forscanning. **Note:** This is a “half-right-open” bound. For example, if SIGINT-START-FREQUENCY is set to 1e9 and SIGINT-END-FREQUENCY is set to 2e9, frequencies greater than or equal to 1 GHz and strictly less than 2 GHz are included. **Note:** This parameter is required if SIGINT-SCANNER-TYPE is set to BASIC or SWEEP.
SIGINT-SWEEP-BANDWIDTH *Optional* *Scope:* Global *Instances:* scanner index	Integer *Range:* > 0 *Unit:* Hz	Specifies the bandwidth of frequency that is scanned in a single sweep. **Note:** This parameter is required if SIGINT-SCANNER-TYPE is set to SWEEP.
SIGINT-SWEEP-SLOT-DURATION *Optional* *Scope:* Global *Instances:* scanner index	Time *Range:* > 0S	Specifies the duration of a single sweep. **Note:** This parameter is required if SIGINT-SCANNER-TYPE is set to SWEEP.

TABLE 6-5. SIGINT Model Scenario Configuration File Parameters (Continued)

Parameter	Value	Description
SIGINT-SWEEP-PATTERN *Optional* *Scope:* Global *Instances:* scanner index	List • SEQ • RANDOM	Specifies the pattern for sweeping the frequency blocks within the frequency range as follows: SEQ: Sweeps the frequency blocks sequentially. RANDOM: Sweeps the frequency blocks in random order. **Note:** This parameter is required if SIGINT-SCANNER-TYPE is set to SWEEP.
SIGINT-SCANNER-FILE *Optional* *Scope:* Global *Instances:* scanner index	Filename	Specifies a filename that contains the custom frequency range and scanning strategy for scanning. The format of this file is described in Section 6.3.3.1. **Note:** This parameter is required if SIGINT-SCANNER-TYPE is set to FILE.

6.3.3.1 Format of SIGINT Scanner File

The SIGINT Scanner file defines a custom frequency range and scanning strategy for the SIGINT model. Each line of the file defines the frequency range and the time interval, as follows:

```
<start-frequency> <end-frequency> <start-time> <end-time>
```

where

<start-frequency> Lower bound of frequency range to be scanned.

<end-frequency> Upper bound of frequency range to be scanned.

Note: This is “half right open” bound, which means the <start-frequency> value is included in the range, but <end-frequency> value is not.

<start-time> Time when scanning should start.

Note: This is not the absolute time, but is relative to the time when the SIGINT model is activated.

<end-time> Time when scanning should end.

Note: This is not the absolute time, but is relative to the time when the SIGINT model is activated.

Note: The scanning pattern repeats after the largest <end-time> specified in the file.

Example of SIGINT Scanner File

The following is an example of a SIGINT Scanner File:

```
1e9  2e9 0S 10S
5e9  6e9 0S 10S
1e9 10e9 10S 20S
```

The above configures a scanner that scans frequency range of [1 GHz, 2 GHz) and [5 GHz, 6 GHz) for 10 seconds, and then a range of [1 GHz, 10 GHz) for the next 10 seconds. At the end of 20 seconds, the scanner will repeat this pattern.

6.3.4 GUI Configuration

To configure SIGINT scanners perform the following steps:

1. Go to **Scenario Configuration Editor > Cyber**.
2. Set **Number of SIGINT Scanners** to the desired value as shown in Figure 6-6.

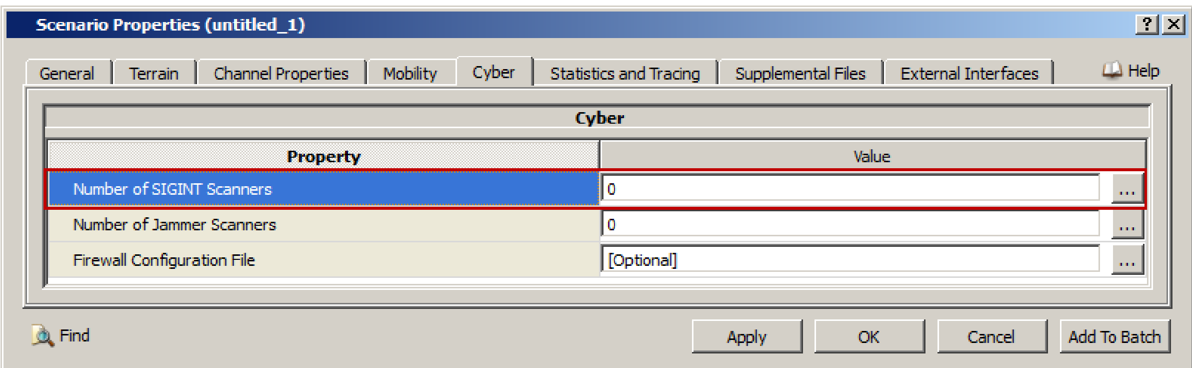


FIGURE 6-6. Setting Number of SIGINT Scanners

3. To configure the SIGINT scanner properties, do the following:
   a. Click the **Open Array Editor**  button in the **Value** column. This opens the Array Editor (see Figure 6-7).
   b. Set the parameters listed in Table 6-6 for each scanner index.

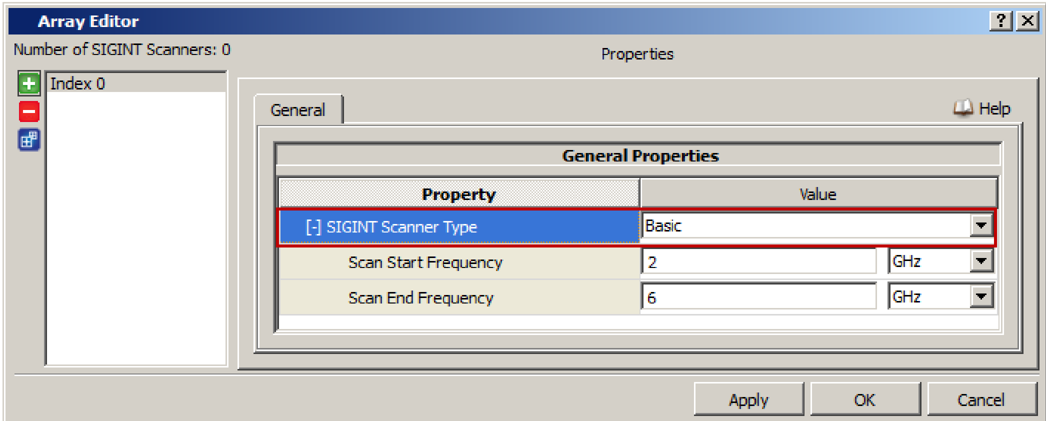


FIGURE 6-7. Setting SIGINT Scanner Type

TABLE 6-6. Command Line Equivalent of SIGINT Scanner Type Parameters

GUI Parameter	Scope of GUI Parameter	Command Line Parameter
SIGINT Scanner Type	Global	`SIGINT-SCANNER-TYPE`

4. If **SIGINT Scanner Type** is set to *Basic* then set the dependent parameters listed in Table 6-7.

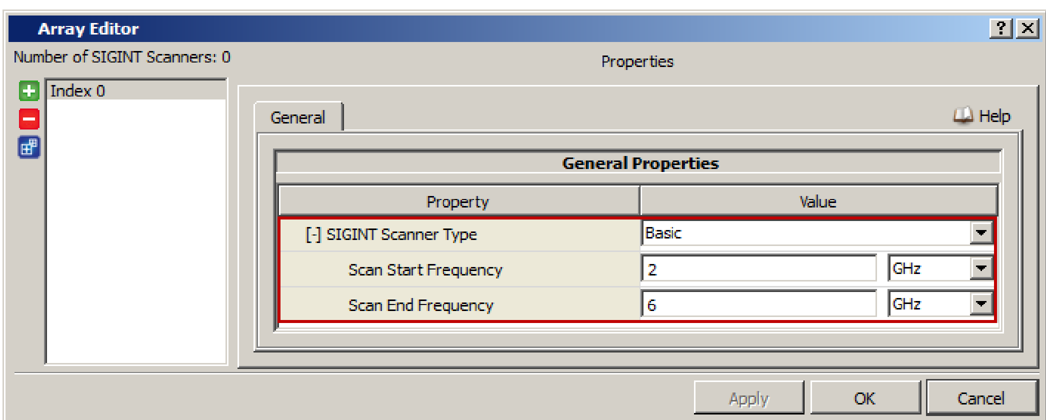


FIGURE 6-8. Setting Basic Scanner Parameters

TABLE 6-7. Command Line Equivalent of Basic Scanner Parameters

GUI Parameter	Scope of GUI Parameter	Command Line Parameter
Scan Start Frequency	Global	`SIGINT-START-FREQUENCY`
Scan End Frequency	Global	`SIGINT-END-FREQUENCY`

5. If **SIGINT Scanner Type** is set to *Sweeping* then set the dependent parameters listed in Table 6-8.

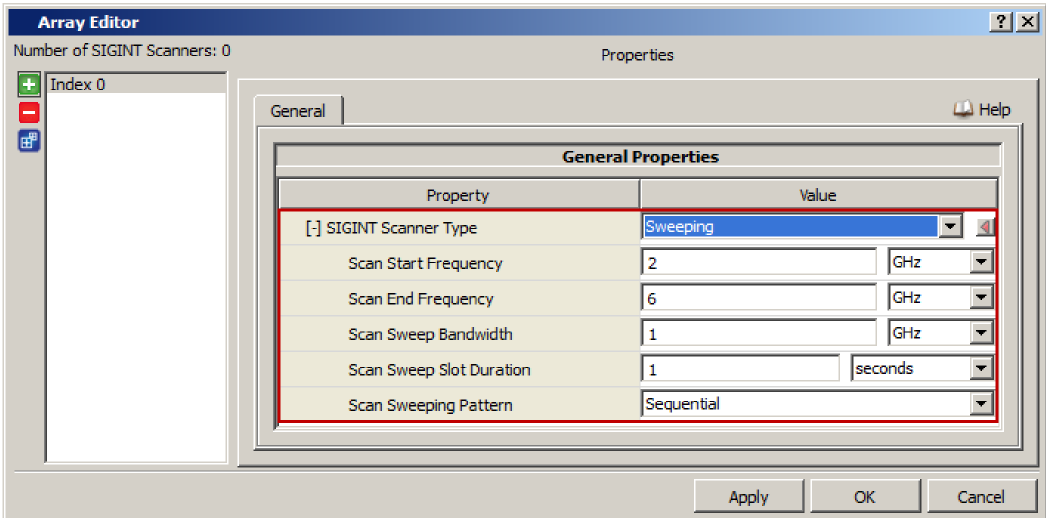


FIGURE 6-9. Setting Sweep Scanner Parameters

TABLE 6-8. Command Line Equivalent of Sweep Scanner Parameters

GUI Parameter	Scope of GUI Parameter	Command Line Parameter
Scan Start Frequency	Global	`SIGIN-START-FREQUENCY`
Scan End Frequency	Global	`SIGIN-END-FREQUENCY`
Scan Sweep Bandwidth	Global	`SIGINT-SWEEP-BANDWIDTH`
Scan Sweep Slot Duration	Global	`SIGINT-SWEEP-SLOT-DURATION`
Scan Sweeping Pattern	Global	`SIGINT-SWEEP-PATTERN`

6. If **SIGINT Scanner Type** is set to *File* then set the dependent parameters listed in Table 6-9.

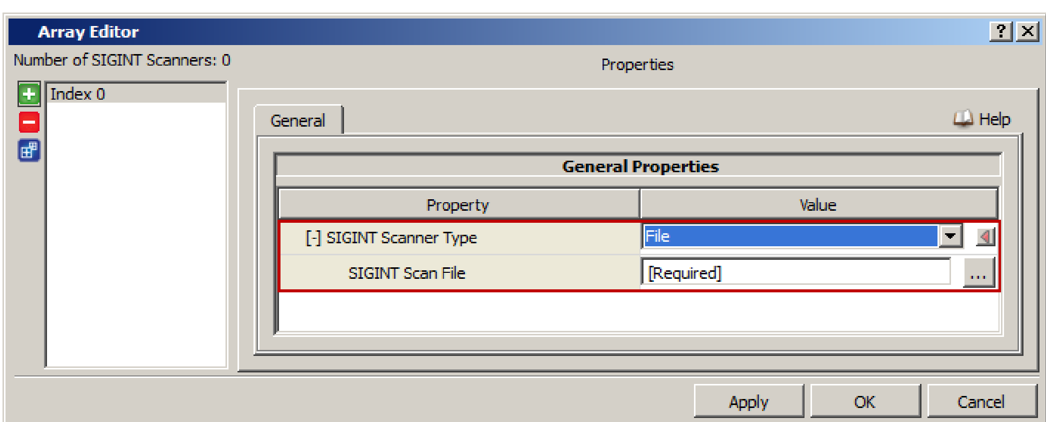


FIGURE 6-10. Setting Custom Scanner Parameters

TABLE 6-9. Command Line Equivalent of Custom Scanner Parameters

GUI Parameter	Scope of GUI Parameter	Command Line Parameter
SIGINT Scan File	Global	`SIGINT-SCANNER-FILE`

Setting Parameters

- Set **SIGINT Scanner File** to the name of the SIGINT scanner file. The format of this file is described in Section 6.3.3.1.

6.3.5 Runtime Commands for SIGINT Model

SIGINT attacks are launched and terminated during the scenario execution from the Human-In-The-Loop (HITL) interface of the EXata GUI (see Section 6.1). This section describes how to launch and terminate SIGINT attacks at runtime.

Launching a SIGINT Attack

To launch a SIGINT attack from a node, execute the following command at the HITL interface.

```
sigint <src> <scanner-instance-id> [<duration>] [-v] [-s]
```

where

`<src>`	Node ID or IP address of the SIGINT node.
`<scanner-index>`	Index of the scanner (see Section 6.3.3 and Section 6.3.4) to be used by the SIGINT model.
`<duration>`	Duration of the scanning. (The scanning starts as soon as the command is sent.) If the duration is not specified or is set to 0s, the scanning will continue until explicitly stopped via an HITL command or until the end of simulation, whichever occurs first.
`-v`	Option to show verbose output. If this option is specified, information about each signal that the SIGINT model has detected is printed to the **Output Window** of the GUI.
`-s`	Option to show periodic summary output. If this option is specified, a summary SIGINT report is printed to the **Output Window** of the GUI every three seconds.

Terminating a SIGINT Attack

To terminate the SIGINT attack from a node, execute the following command at the HITL interface:

```
stop sigint <src>
```

where

`<src>`	Node ID or IP address of the SIGINT node.

6.3.6 Statistics

No statistics are collected for the SIGINT model.

6.3.7 References

None.

6.4 Virus Attack Model

6.4.1 Description

An attacker in the network launches a virus attack at a victim's computer by sending malformed packets with malicious payloads. The effect of the attacks could range from shutting down the victim's computer to taking over complete ownership.

In EXata, a virus attack is modeled as the attacker node sending packets with payloads that contain signatures of some well-known attacks. Note that these packets *do not* contain any actual virus payload, only their signatures. It is expected that any Intrusion Detection Systems (IDS) or Anti-Virus Software can detect the signature of these packets and classify them as malicious.

6.4.2 Configuration Recommendations

To observe the effect of a virus attack in a scenario, it is recommended that the victim node be configured as an External Node. i.e., one that is mapped to an operational host (refer to Chapter 5 of *EXata User's Guide*). In this case, the malformed packets sent by the attacker are forwarded to the counterpart operational host. At the operational host, the effect of receiving the malformed packets can be observed by running an IDS, such as Snort (www.snort.org), which will detect malformed packet payloads and trigger alerts.

6.4.3 Runtime Commands for Virus Attack Model

To launch a Virus attack, execute the following command from the HITL interface (see Section 6.1):

```
attack <attacker-node-id> <victim-IP-Address>
```

where

`<attacker-node-id>`	ID of the node that is launching the attack.
`<victim-IP-Address>`	IP address of the victim node at which the attack is targeted.

Example:

The following command launches a virus attack from node 15 on interface 192.168.1.102:

```
attack 15 192.168.1.102
```

6.4.4 Statistics

No statistics are collected for the Virus Attack model.

6.4.5 References

None.

6.5 Wireless Eavesdropping Attack Model

6.5.1 Description

Eavesdropping is a *passive* attack where an intruder node attempts to capture private information from a network. In wireless eavesdropping, the intruder node configures its radio to be on the same channel as the victim network and promiscuously listens for broadcast transmissions that are destined for member nodes of the network.

In EXata, an eavesdropping attack is modeled as the eavesdropping node's MAC layer operating in promiscuous mode, enabling it to promiscuously listen to nearby wireless communication.

6.5.2 Network Configuration Recommendations

To observe the effect of an eavesdropping attack in a scenario, it is recommended that the eavesdropper node be configured as an External Node. i.e., one that is mapped to an operational host (refer to Chapter 5 of EXata *User's Guide*). In this case, the eavesdropped traffic is forwarded to the counterpart operational host. At the operational host, the eavesdropped traffic can be observed by running a generic packet-sniffing application such as wireshark or an application appropriate for the type of the eavesdropped traffic.

6.5.3 Command Line Configuration

To enable a node to be an eavesdropper, the MAC layer of the interface on which the node is to eavesdrop must be configured to operate in promiscuous mode.

To configure promiscuous mode in the command line interface, include the following parameter in the scenario configuration (.config) file:

```
[<interface-address>] PROMISCUOUS-MODE       YES
```

where

`<interface-address>`	Address of the interface on which the node is to eavesdrop.

Refer to EXata *User's Guide* for details.

6.5.4 GUI Configuration

To configure promiscuous mode in the GUI, do the following:

1. Go to one of the following locations:

- To set properties at the subnet level, go to **Wireless Subnet Properties Editor > MAC Layer**.
- To set properties at the interface level, go to one of the following locations:
  - **Interface Properties Editor > Interfaces > Interface # > MAC Layer**.
  - **Default Device Properties Editor > Interfaces > Interface # > MAC Layer**.

In this section, we show how to configure ANODR parameters for a specific node using the Default Device Properties Editor. Parameters can be set in the other properties editors in a similar way.

2. Set **Enable Promiscuous Mode** to *Yes*.

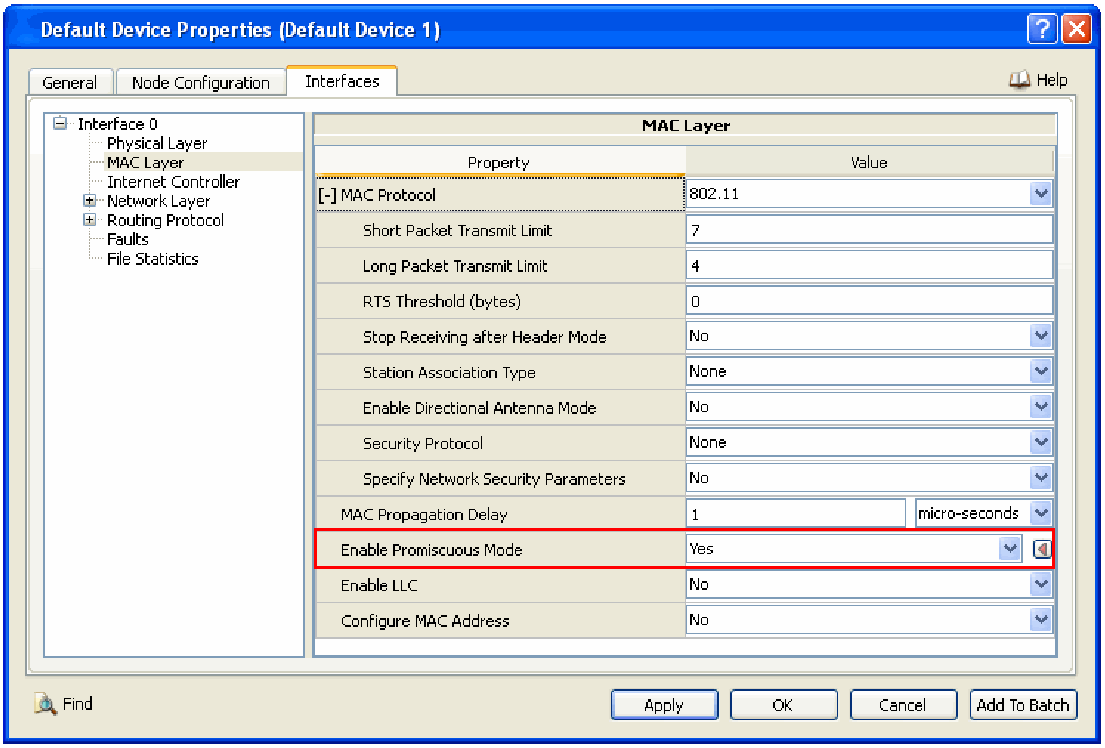


FIGURE 6-11. Enabling Promiscuous Mode

TABLE 6-10. Command Line Equivalent of Promiscuous Mode Parameters

GUI Parameter	Scope of GUI Parameter	Command Line Parameter
Enable Promiscuous Mode	Subnet, Interface	`PROMISCOUS-MODE`

Refer to EXata *User's Guide* for details.

6.5.5 Runtime Commands for Eavesdropping Attack Model

To launch an eavesdropping attack, execute the following command from the HITL interface (see Section 6.1):

```
eaves <eavesdropper-node-id>
```

or

```
eaves <eavesdropper-node-id> switchheader
```

or

```
eaves <eavesdropper-node-id> hla
```

where

`<eavesdropper-node-id>`	ID of the node that is passively eavesdropping the wireless channel.
`switchheader`	Option to change the destination address in the IP header to the address of corresponding operational host.
`hla`	Option to eavesdrop external traffic arriving over an HLA interface.

Example:

The following command causes node 5 to start eavesdropping on traffic on the wireless channel:

```
eaves 5
```

6.6 Wireless Jamming Attack Model

6.6.1 Description

Radio jamming, or simply jamming, is transmission of radio signals at sufficiently high energy to cause disruption of communication for nearby radios. The signals transmitted by jammers interfere with other legitimate signals in the vicinity of the jammer, causing the signal to noise ratio of the latter signals to drop significantly and resulting in corruption of those signals.

The Jammer model in EXata supports the following strategies of frequency selection for jamming:

- Wideband Jamming: The jammer jams all transmissions in a specified frequency range.
- Sweep Jamming: The jammer divides the frequency range in contiguous blocks of sweep bandwidth each. The jammer jams each frequency block at a time for the sweep slot duration before moving to next frequency block. The sweep order can be either sequential or random.
- Custom Jamming: This mode offers greatest configuration control over the frequency ranges and scanning patterns that must be jammed by the jammer. The jammer jams frequency ranges according to the scanning patterns specified in a file.

The Jammer model in EXata supports the following strategies for jamming:

- Continuous Jamming: The jammer continuously transmits a radio signal in frequency range(s) as configured with the frequency selection strategy. This is the default mode.
- Silent Jamming: The jammer transmits a radio signal only when it detects another signal transmission on the channel and stops when that signal transmission has ended.

6.6.2 Features and Assumptions

This section describes the implemented features, omitted features, assumptions and limitations of the Jammer model.

6.6.2.1 Implemented Features

- Wideband jamming
- Sequential and Random Sweep jamming
- Custom frequency sweep jamming
- Commands to launch and terminate the Jammer model during runtime

6.6.2.2 Omitted Features

- Cross-channel and partial-channel overlap jamming

6.6.2.3 Assumptions and Limitations

In Continuous Jamming mode, the Jammer model is effective only against the following PHY models:

- Abstract PHY model, 802.11a PHY model, and 802.11b PHY model (see *Wireless Model Library*)
- 802.15.4 PHY model (see *Sensor Networks Model Library*)
- GSM PHY model (see *Cellular Model Library*)

In Silent Jamming mode, the Jammer model is effective only against the following PHY models:

- 802.11a PHY model
- 802.11b PHY model

6.6.3 Command Line Configuration

To configure the Jammer model, the frequency range and scanning strategy must be configured in the scenario configuration (.config) file, as described in Section 6.6.3.1. Commands to activate the Jammer model are specified in the application configuration (.app) file, as described in Section 6.6.3.2.

6.6.3.1 Scenario Configuration File Parameters

Table 6-11 lists the Jammer model parameters that are specified in the scenario configuration (.config) file. See Section 1.2.1.1 for a description of the format used for the parameter table.

TABLE 6-11. Jammer Model Scenario Configuration File Parameters

Parameter	Value	Description
`JAMMER-SCANNER-TYPE` *Optional* *Scope:* Global *Instances:* scanner index	List • `BASIC` • `SWEEP` • `FILE` *Default:* `BASIC`	Specifies how the frequencies are selected for jamming as follows: `BASIC`: Selects a wideband range of frequency for jamming. `SWEEP`: Sweeps across a frequency range in small blocks of bandwidth at a time. `FILE`: Uses a custom strategy of frequency selection for jamming.
`JAMMER-START-FREQUENCY` *Optional* *Scope:* Global *Instances:* scanner index	Integer *Range:* `> 0` *Unit:* Hz	Specifies the lower end of frequency range selected for jamming. **Note:** This parameter is required if `JAMMER-SCANNER-TYPE` is set to `BASIC` or `SWEEP`.
`JAMMER-END-FREQUENCY` *Optional* *Scope:* Global *Instances:* scanner index	Integer *Range:* `> 0` *Unit:* Hz	Specifies the upper end of the frequency range selected for jamming. **Note:** This is a “half-right-open” bound. For example, if `JAMMER-START-FREQUENCY` is set to `1e9` and `JAMMER-END-FREQUENCY` is set to `2e9`, frequencies greater than or equal to 1 GHz and strictly less than 2 GHz are included. **Note:** This parameter is required if `JAMMER-SCANNER-TYPE` is set to `BASIC` or `SWEEP`.
`JAMMER-SWEEP-BANDWIDTH` *Optional* *Scope:* Global *Instances:* scanner index	Integer *Range:* `> 0` *Unit:* Hz	Specifies the bandwidth of frequency that is jammed in a single sweep. **Note:** This parameter is required if `JAMMER-SCANNER-TYPE` is set to `SWEEP`.

TABLE 6-11. Jammer Model Scenario Configuration File Parameters (Continued)

Parameter	Value	Description
JAMMER-SWEEP-SLOT-DURATION *Optional* *Scope:* Global *Instances:* scanner index	Time *Range:* > 0S	Specifies the duration of a single sweep. **Note:** This parameter is required if JAMMER-SCANNER-TYPE is set to SWEEP.
JAMMER-SWEEP-PATTERN *Optional* *Scope:* Global *Instances:* scanner index	List • SEQ • RANDOM	Specifies the pattern for sweeping the frequency blocks within the frequency range as follows: SEQ: Sweeps the frequency blocks sequentially. RANDOM: Sweeps the frequency blocks in random order. **Note:** This parameter is required if JAMMER-SCANNER-TYPE is set to SWEEP.
JAMMER-SCANNER-FILE *Optional* *Scope:* Global *Instances:* scanner index	Filename	Specifies a filename that contains the custom frequency range and scanning strategy for jamming. The format of this file is described in Section 6.6.3.1.1. **Note:** This parameter is required if JAMMER-SCANNER-TYPE is set to FILE.
JAMMER-STATISTICS *Optional* *Scope:* All	List: • YES • NO *Default:* NO	Indicates whether statistics are collected for the Jammer model.

6.6.3.1.1 Format of Jammer Scanner File

The Jammer Scanner file defines a custom frequency range and scanning strategy for the Jammer model. Each line of the file defines the frequency range and the time interval, as follows:

```
<start-frequency> <end-frequency> <start-time> <end-time>
```

where

`<start-frequency>`	Lower bound of frequency range to be jammed.
`<end-frequency>`	Upper bound of frequency range to be jammed. **Note:** This is “half right open” bound, which means the `<start-frequency>` value is included in the range, but `<end-frequency>` value is not.
`<start-time>`	Time when jamming should start. **Note:** This is not the absolute time, but is relative to the time when the jammer is activated.
`<end-time>`	Time when jamming should end. **Note:** This is not the absolute time, but is relative to the time when the jammer is activated.

Note: The jamming pattern repeats after the largest `<end-time>` specified in the file.

Example of Jammer Scanner File

The following is an example of a Jammer Scanner File:

```
1e9  2e9 0S 10S
5e9  6e9 0S 10S
1e9 10e9 10S 20S
```

The above configures a jammer that jams frequency range of [1 GHz, 2 GHz) and [5 GHz, 6 GHz) for 10 seconds, and then a range of [1 GHz, 10 GHz) for the next 10 seconds. At the end of 20 seconds, the jammer will repeat this pattern.

6.6.3.2 Application Configuration File Parameters

To activate the Jammer model, include the following statement in the application configuration (.app) file:

```
JAMMER <jammer-node> <start-time> <end-time> <scanner-index>
       [SILENT <min-data-rate>]
```

Note: All parameters should be entered on the same line.

The Jammer model parameters are described in Table 6-12. See Section 1.2.1.1 for a description of the format used for the parameter table.

TABLE 6-12. Jammer Model Parameters

Parameters	Value	Description
`<jammer-node>` *Required*	Integer or IP Address	Node ID or IP address of the jammer.
`<start-time>` *Required*	Time *Range:* ≥ `0S`	Time when the jammer should start.
`<end-time>` *Required*	Time *Range:* ≥ `0S`	Time when the jammer should stop. **Note:** `<end-time>` should be `0` or greater than `<start-time>`. If `<end-time>` is set to `0`, the jamming continues till the end of simulation.
`<scanner-index>` *Required*	Integer *Range:* ≥ `0`	Index of the scanner (see Section 6.6.3.1) to use for jamming.
`SILENT <min-data-rate>` *Optional*	Integer *Range:* ≥ `0` *Unit:* bits/sec	Threshold packet data rate above which the jammer transmits a jamming signal.

6.6.4 GUI Configuration

To configure the Jammer model, the frequency range and scanning strategy must be configured using the Scenario Properties Editor, as described in Section 6.6.4.1. The Jammer model is activated on a node, as described in Section 6.6.4.2. Section 6.6.4.3 describes how to configure statistics parameters for the Jammer model.

6.6.4.1 Configuring Scanner Properties for the Jammer

To configure the frequency selection and scanning strategy for the Jammer model in the GUI, do the following:

1. Go to **Scenario Configuration Editor > Cyber**.
2. Set **Number of Jammer Scanners** to the desired value as shown in Figure 6-12.

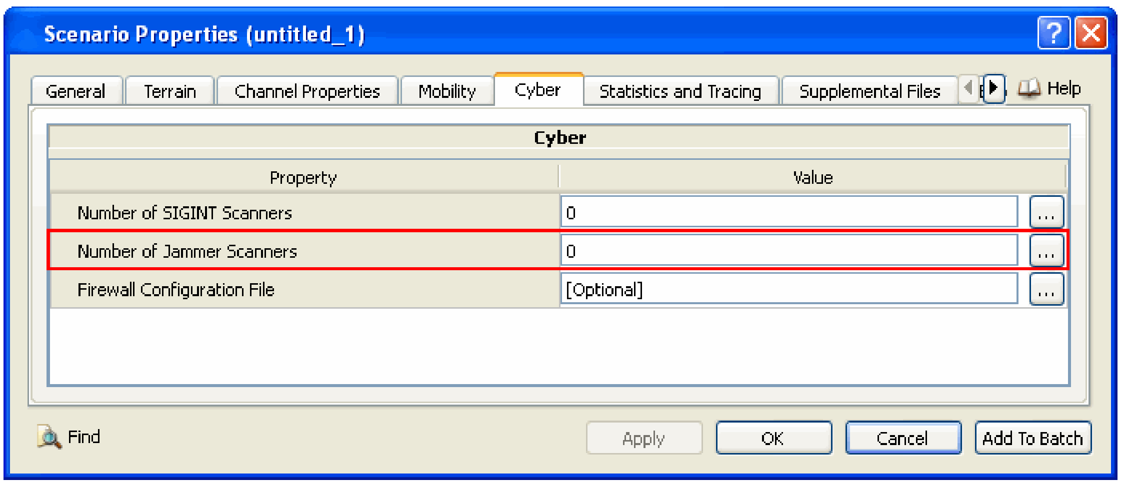


FIGURE 6-12. Setting the Number of Jammer Scanners

3. To configure the jammer scanner properties, do the following:
   a. Click the **Open Array Editor** [...] button in the **Value** column. This opens the Array Editor (see Figure 6-13).
   b. Set the parameters listed in Table 6-13 for each scanner index.

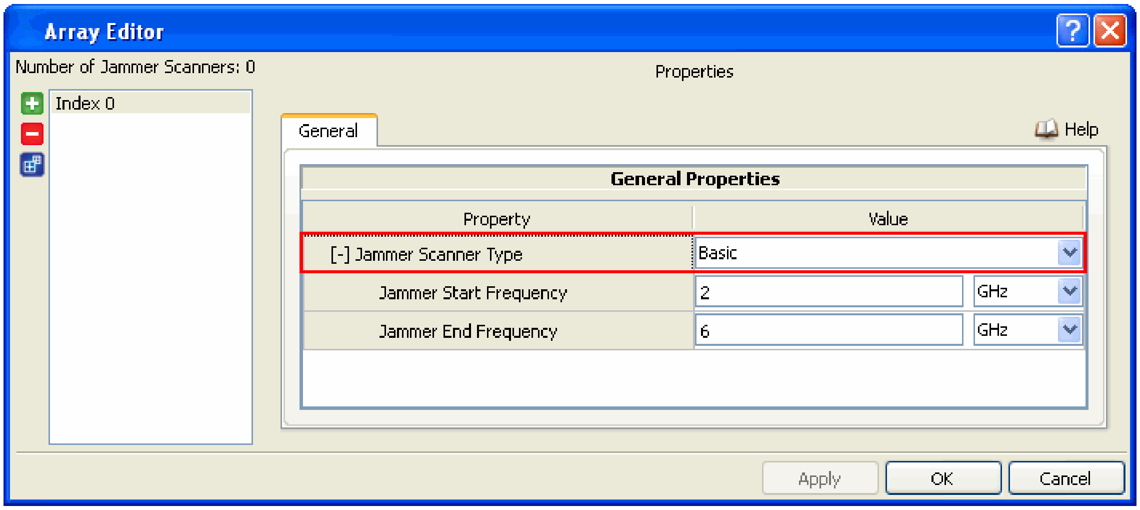


FIGURE 6-13. Setting Jammer Scanner Type

TABLE 6-13. Command Line Equivalent of Jammer Scanner Type Parameters

GUI Parameter	Scope of GUI Parameter	Command Line Parameter
Jammer Scanner Type	Global	`JAMMER-SCANNER-TYPE`

4. If **Jammer Scanner Type** is set to *Basic* then set the dependent parameters listed in Table 6-14.

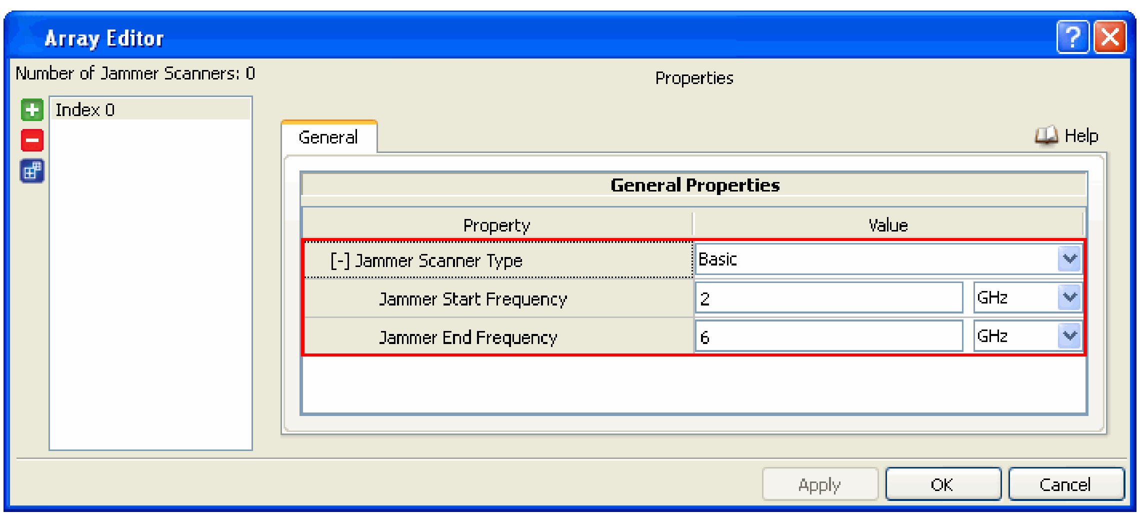


FIGURE 6-14. Setting Basic Scanner Parameters

TABLE 6-14. Command Line Equivalent of Basic Scanner Parameters

GUI Parameter	Scope of GUI Parameter	Command Line Parameter
Jammer Start Frequency	Global	`JAMMER-START-FREQUENCY`
Jammer End Frequency	Global	`JAMMER-END-FREQUENCY`

5. If **Jammer Scanner Type** is set to *Sweeping* then set the dependent parameters listed in Table 6-15.

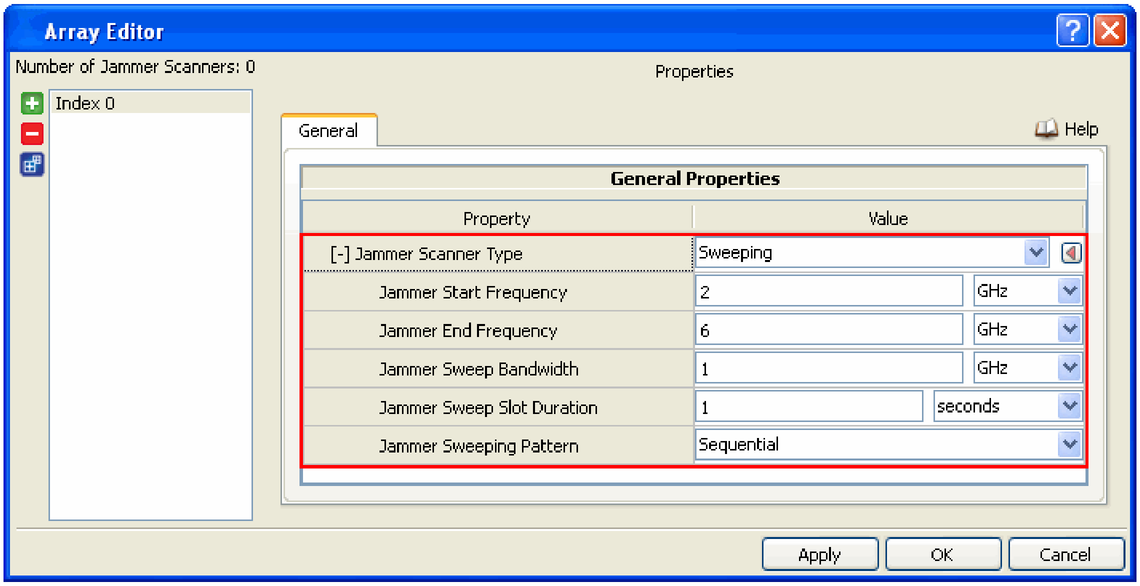


FIGURE 6-15. Setting Sweep Scanner Parameters

TABLE 6-15. Command Line Equivalent of Sweep Scanner Parameters

GUI Parameter	Scope of GUI Parameter	Command Line Parameter
Jammer Start Frequency	Global	`JAMMER-START-FREQUENCY`
Jammer End Frequency	Global	`JAMMER-END-FREQUENCY`
Jammer Sweep Bandwidth	Global	`JAMMER-SWEEP-BANDWIDTH`
Jammer Sweep Slot Duration	Global	`JAMMER-SWEEP-SLOT-DURATION`
Jammer Sweeping Pattern	Global	`JAMMER-SWEEP-PATTERN`

6. If **Jammer Scanner Type** is set to *File* then set the dependent parameters listed in Table 6-16.

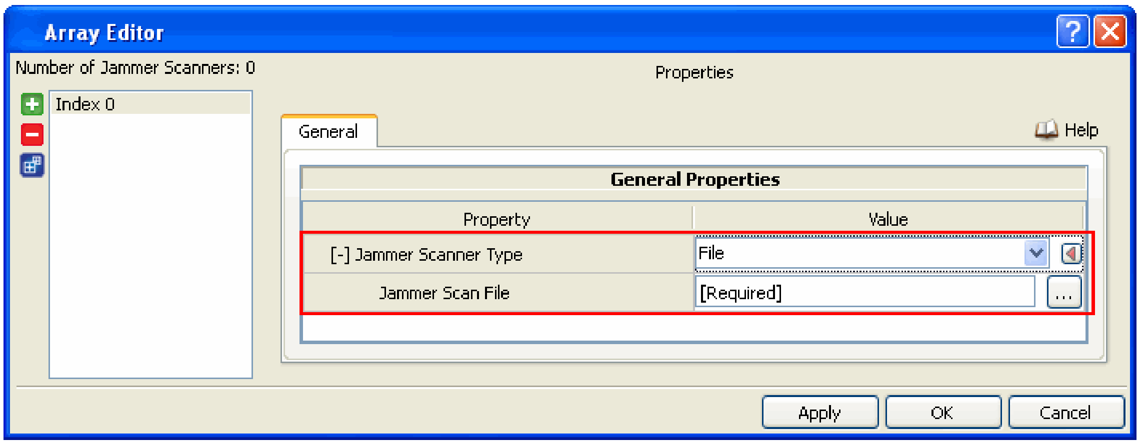


FIGURE 6-16. Setting Custom Scanner Parameters

TABLE 6-16. Command Line Equivalent of Custom Scanner Parameters

GUI Parameter	Scope of GUI Parameter	Command Line Parameter
Jammer Scan File	Global	`JAMMER-SCANNER-FILE`

Setting Parameters

- Set **Jammer Scanner File** to the name of the jammer scanner file. The format of this file is described in Section 6.6.3.1.1.

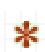
6.6.4.2 Configuring the Jammer Model

To configure the Jammer model at a node, perform the following steps:

1. Click the **JAMMER** button in the **Cyber Attacks** tab of the Standard Toolset.

2. On the canvas, click on the node that is to be the jammer.

3. Open the Jammer Properties Editor by doing one of the following:

   c. Right-click on the  symbol next to the jammer node on the canvas and select **Properties** from the menu.

   d. In the **Applications** tab of Table View either double-click on the application row or right-click on the Jammer application row and select **Properties** from the menu.

4. Set the parameters listed in Table 6-17.

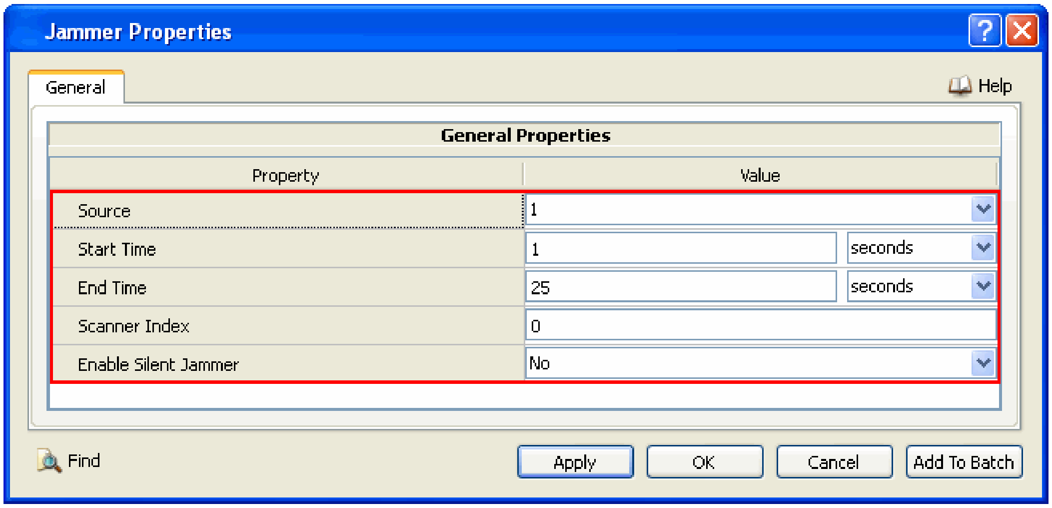


FIGURE 6-17. Setting Jammer Parameters

TABLE 6-17. Command Line Equivalent of Jammer Parameters

GUI Parameter	Command Line Parameter
Source	<jammer-node>
Start Time	<start-time>
End Time	<end-time>
Scanner Index	<scanner-index>
Enable Silent Jammer (set to Yes)	SILENT

Setting Parameters

- To specify an IP address as the source (jammer), set **Source** to one of the IP addresses listed in the drop-down list.
- Set **Scanner Index** to the index in the Array Editor (Index 0, Index 1, etc.) corresponding to the scanner to be used for jamming (see Figure 6-13).
- To enable Silent jamming mode, set **Enable SIlent Jammer** to *Yes*; otherwise, set **Enable SIlent Jammer** to *No*.

5. If **Enable SIlent Jammer** is set to *Yes*, then set the parameters listed in Table 6-18.

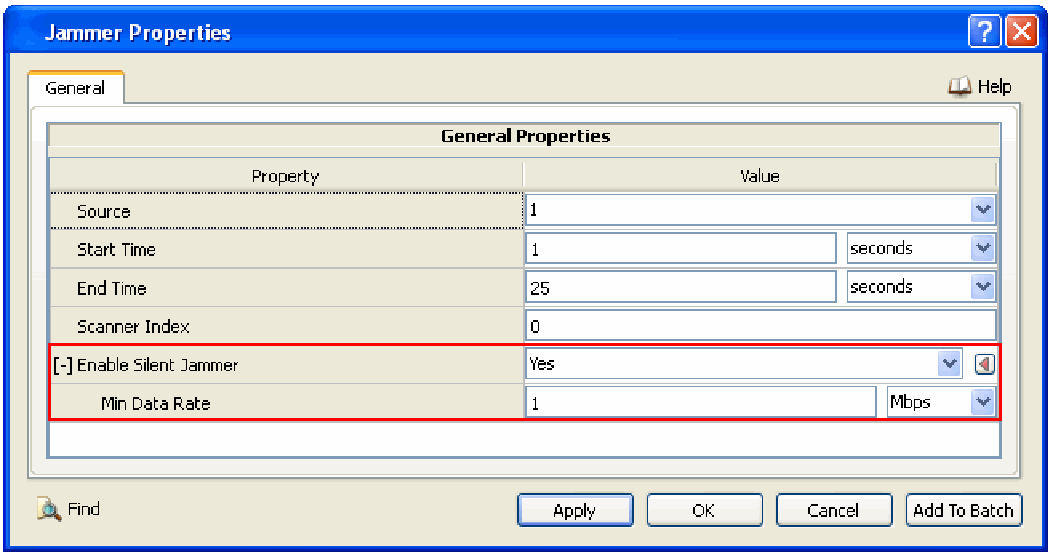


FIGURE 6-18. Setting Jammer Parameters

TABLE 6-18. Command Line Equivalent of Jammer Parameters

GUI Parameter	Command Line Parameter
Min Data Rate	`<min-data-rate>`

6.6.4.3 Configuring Statistics Parameters for the Jammer Model

Statistics for the Jammer model can be collected at the global, node, subnet, and interface levels. See Section 4.2.9 of *EXata User's Guide* for details of configuring statistics parameters.

To enable statistics collection for the Jammer model, check the box labeled **Jammer Statistics** in the appropriate properties editor.

TABLE 6-19. Command Line Equivalent of Statistics Parameters

GUI Parameter	Scope of GUI Parameter	Command Line Parameter
Jammer Statistics	Global, Node, Subnet, Interface	`JAMMER-STATISTICS`

6.6.5 Runtime Commands for Jammer Model

Jammer attacks can be launched and terminated during the scenario execution from the Human-In-The-Loop (HITL) interface of the EXata GUI (see Section 6.1). This section describes how to launch and terminate jammer attacks at runtime.

Launching Jammer Attacks at Runtime

To launch a jammer attack at runtime, execute the following command from the HITL interface:

```
jammer <jammer-node> <duration> <scanner-index>
        [SILENT <min-data-rate>]
```

where

`<jammer-node>`	Node ID or IP address of the jammer.
`<duration>`	Duration of the jamming attack. (The attack starts as soon as the command is sent.)
`<scanner-index>`	Index of the scanner (see Section 6.6.3.1 and Section 6.6.4) to use for jamming.
`<min-data-rate>`	Threshold packet data rate (in bits/sec) above which the jammer transmits a jamming signal. This parameter is needed only if the keyword `SILENT` is included in the command to activate the silent mode of jammer attack.

Terminating Jammer Attacks at Runtime

To terminate all jammer attacks from a node, execute the following command from the HITL interface:

```
stop jammer <jammer-node>
```

where

`<jammer-node>`	Node ID or IP address of the jammer to be stopped.

6.6.6 Statistics

Table 6-20 lists the Jammer model statistics that are output to the statistics (.stat) file at the end of simulation.

TABLE 6-20. Jammer Model Statistics

Statistic	Description
Signals received and forwarded to MAC during the jam duration	Number of PHY signals that are received and forwarded to MAC layer during the jamming duration
Signals locked on by PHY during the jam duration	Number of PHY signals that are locked during the jamming duration
Signals received but with errors during the jam duration	Number of PHY signals that are received but with errors during the jamming duration
Total jam duration in (s)	Total time in seconds for jamming duration

6.6.7 Scenarios Included in EXata

The EXata distribution includes several sample scenarios for the Jammer model. All scenarios are located in the directory EXATA_HOME/scenarios/cyber/jammer. Table 6-21 lists the sub-directory where each scenario is located.

TABLE 6-21. Jammer Scenarios Included in EXata

Scenario	Description
multi-networks	Jamming PHY-ABSTRACT, PHY-802.11a, PHY-802.11b, and PHY-802.15.4 models.
silent-jammer	Silent jamming PHY-ABSTRACT, PHY-802.11a, PHY-802.11b, and PHY-802.15.4 models.

6.6.8 References

None.

7 OS Resource Models

This chapter describes features, configuration requirements and parameters, statistics, and scenarios for OS Resource Models in the Cyber Model Library, and consists of the following section:

- CPU and Memory Resource Model

7.1 CPU and Memory Resource Model

7.1.1 Description

The CPU and Memory Resource Model monitors the allocation, consumption, and depletion of resources for a node. This model is used in conjunction with Denial of Service Attack (DOS) model (Section 6.2). The DOS attack model attempts to consume the resources at the victim node, causing the victim node to fail when the resources are completely depleted.

The CPU and Memory Resource Model in EXata models the following two types of resources:

- Random Access Memory
- CPU utilization

7.1.1.1 Modeling Memory Usage

The memory usage is modeled as number of bytes of memory consumed by the protocol stack.

The following actions result in allocation of memory:

- A packet is enqueued in the network input or output queues.
- A fragmented IP packet is received at the destination and at least one fragment is currently outstanding.
- A new TCP connection is established.

The following actions result in de-allocation of memory:

- A packet is dequeued from the network input or output queues.
- The last pending fragment of an IP packet is received.
- A TCP connection is terminated.

The Memory OS Resource is configured with the following parameters:

- The maximum capacity of memory, at the start of simulation.
- The memory consumed in storing a packet or fragment. This is equal to the sum of the size of the packet and a constant overhead.
- The memory consumed in creating a TCP connection. This is a constant value.

7.1.1.2 Modeling CPU Utilization

The CPU utilization measurement technique in EXata is slightly different from those in conventional Operating Systems such as Windows and Linux. The latter systems monitor the CPU utilization of multiple long-running processes, whereas the EXata monitors only one process, viz. the protocol stack. Moreover, in case of EXata, the load on the CPU is discrete in nature, that is, CPU resources are consumed at discrete events such as timer expiration, packet reception etc. This section describes how the CPU and Memory Resource Model monitors the CPU utilization.

The basic unit of CPU utilization is *load,* which is denoted as the time interval required in processing an event. For example, if the CPU can process 1000 events/sec, the magnitude of each event load will be 1 millisecond. The CPU utilization model in EXata uses the concept of *CPU processing backlog*, or backlog in short. If the CPU receives events while it is processing an existing one, all these additional events are cumulatively added into the backlog queue. As an example, suppose CPU is processing a load of 1 msec, with an empty backlog. Suppose further that before this event is completely processed five additional events are requested. At this point the CPU backlog will be 5 msec. The backlog diminishes as time advances. Assuming that no other requests were made in the previous example, the backlog at time

= 2 msec will be 3 msec (since two events would have been processed by that time), and at time = 10 msec the backlog will be 0.

The CPU resource is utilized when:

- A protocol timer expires, i.e., the protocol stack executes some action.
- A packet is received from the network interface.
- A packet is created by the protocol stack.

7.1.1.3 Resource Depletion Behavior

When the memory or CPU resource is completely depleted, the node can be configured to exhibit one of the following two behaviors:

- Shutdown: The node shuts down completely: it will not send or receive any packet to and from the network; and thus, effectively, is removed from the network.
- Recover: The node attempts to free the resource that was depleted.

  In case of memory resource exhaustion, the node will recover lost memory by:

  - Dropping all packets in the network input and output queues.
  - Dropping all fragment packets in the fragmentation buffers.
  - Closing all open TCP connections.

  In case of 100% CPU utilization (that is, the CPU backlog exceeds the backlog threshold), the node will go offline for a duration equal to the current backlog. During this interval the node will not send or receive any packet to and from the network. Since the node will not be processing any events or packets during this interval, the CPU utilization backlog will eventually fall back to 0 seconds.

7.1.2 Features and Assumptions

This section describes the implemented features, omitted features, assumptions and limitations of the CPU and Memory Resource Model.

7.1.2.1 Implemented Features

- CPU and memory resource modeling for random access memory and CPU resources.
- Allocation and consumption of these resources.
- Failure behavior when resources are depleted.

7.1.2.2 Omitted Features

- Monitoring memory usage of UDP connections.

7.1.2.3 Assumptions and Limitations

- In the modeling of OS resources, it is assumed that the node has a dedicated pool of memory for protocol stack operations that is not shared by any other processes.
- In the modeling of OS resources, it is assumed that the node has a dedicated backlog queue of CPU utilization that is not shared by any other processes.
- It is assumed that all TCP connections have equal overhead.
- It is assumed that the processing time for all events (incoming packet, outgoing packet, and protocol timers) is equal.

7.1.3 Command Line Configuration

To enable the CPU and Memory Resource Model at a node, include the following parameter in the scenario configuration (.config) file:

```
[<Qualifier>]   OS-RESOURCE-MODEL     YES
```

The scope of this parameter declaration can be Global or Node. See Section 1.2.1.1 for a description of `<Qualifier>` for each scope.

CPU and Memory Resource Model Parameters

Table 7-1 lists the configuration parameters for the CPU and Memory Resource model. See Section 1.2.1.3 for a description of the format used for the parameter table.

TABLE 7-1. CPU and Memory Resource Model Parameters

Parameter	Value	Description
`OS-MEMORY-CAPACITY` *Optional* *Scope:* Global, Node	Integer *Range:* ≥ `0` *Default:* `0` *Unit:* Bytes	Maximum capacity of the memory resource. A value of 0 indicates an infinite resource.
`OS-CPU-MAX-BACKLOG` *Optional* *Scope:* Global, Node	Time *Range:* ≥ `0S` *Default:* `0S`	CPU backlog threshold. A value of 0 indicates an infinite backlog capacity.
`OS-MEMORY-PACKET-OVERHEAD` *Optional* *Scope:* Global, Node	Integer *Range:* ≥ `0` *Default:* `0` *Unit:* Bytes	Memory overhead in storing a packet. The total memory required to store a packet, therefore, is equal to the sum of the size of packet and this constant overhead. This memory is allocated when a packet is stored in network input queue, output queue, or fragmentation buffers.
`OS-MEMORY-CONNECTION-USAGE` *Optional* *Scope:* Global, Node	Integer *Range:* ≥ `0` *Default:* `0` *Unit:* Bytes	Amount of memory allocated for a new TCP connection. This memory is allocated when a new TCP connection is established and freed when the connection is closed.

TABLE 7-1. CPU and Memory Resource Model Parameters (Continued)

Parameter	Value	Description
OS-CPU-PROCESSING-SPEED *Optional* *Scope:* Global, Node	Integer *Range:* ≥ 0 *Default:* 0 *Unit:* events/ second	Maximum rate at which the CPU can process events without accruing backlog. A value of 0 indicates that the CPU is not utilized in processing of events.
OS-RESOURCE-FAILURE-MODE *Optional* *Scope:* Global, Node	List • SHUTDOWN • RECOVER *Default:* RECOVER	Behavior of node when a resource is completely depleted. In SHUTDOWN mode, the node cannot send or receive packets for the remaining duration of the simulation. In RECOVER mode for memory resources, the node will drop all packets from queues and buffers and close all TCP connections. In RECOVER mode for CPU resources, the node will remain offline until the CPU backlog falls back to 0.

7.1.4 GUI Configuration

This section describes how to configure the CPU and Memory Resource Model in EXata GUI.

Configuring the CPU and Memory Resource Model Parameters

1. Go to **Default Device Properties Editor > Node Configuration > CPU and Memory Resource Model**.
2. Set **Enable CPU and Memory Resource Model** to *YES* and set the dependent parameters listed in Table 7-2.

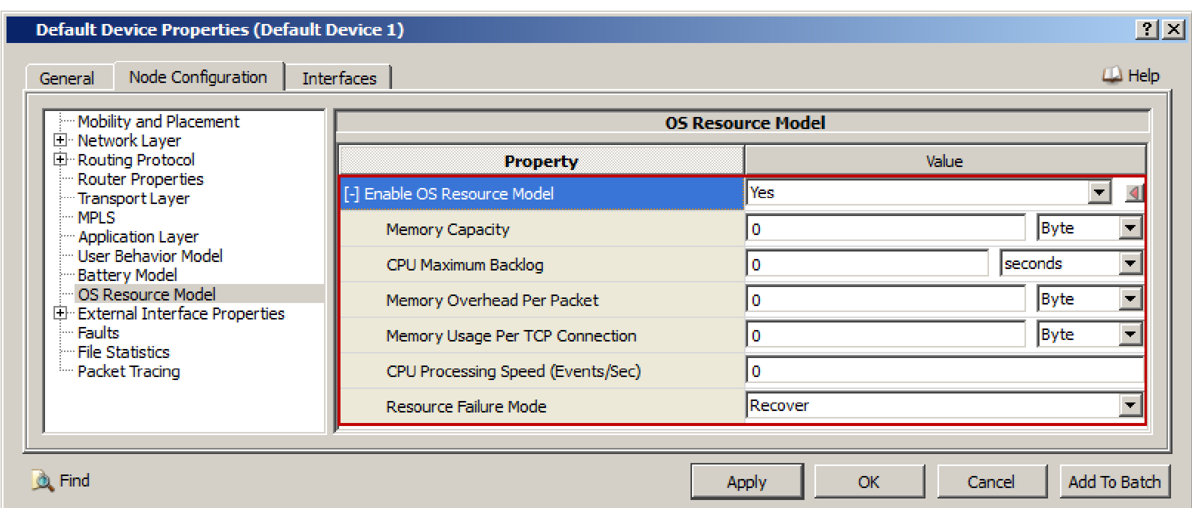


FIGURE 7-1. Setting CPU and Memory Resource Model Parameters

TABLE 7-2. Command Line Equivalent of CPU and Memory Resource Model Parameters

GUI Parameter	Scope of GUI Parameter	Command Line Parameter
Enable CPU and Memory Resource Model	Node	`OS-RESOURCE-MODEL`
Memory Capacity	Node	`OS-MEMORY-CAPACITY`
CPU Maximum Backlog	Node	`OS-CPU-MAX-BACKLOG`
Memory Overhead Per Packet	Node	`OS-MEMORY-PACKET-OVERHEAD`
Memory Usage Per TCP Connection	Node	`OS-MEMORY-CONNECTION-USAGE`
CPU Processing Speed	Node	`OS-CPU-PROCESSING-SPEED`
Resource Failure Mode	Node	`OS-FAILURE-MODE`

7.1.5 Statistics

This section describes the file and dynamic statics collected for the CPU and Memory Resource model.

7.1.5.1 File Statistics

Table 7-3 lists the CPU and Memory Resource model statistics that are output to the statistics (.stat) file at the end of simulation.

TABLE 7-3. CPU and Memory Resource Model Statistics

Statistic	Description
Peak memory usage	Peak value of memory used by the node during the simulation run.
Peak CPU backlog	Peak value of CPU backlog at the node during the simulation run.
Memory failures	Number of times the node failed due to memory resource depletion.
CPU failures	Number of times the node failed due to CPU backlog reaching its threshold value.

7.1.5.2 Dynamic Statistics

The following dynamic statistics are enabled for the CPU and Memory Resource model (refer to Chapter 6 of *EXata User's Guide* for details of viewing dynamic statistics in the GUI during the simulation):

- Current Memory Usage
- Current CPU Backlog

7.1.6 Scenarios Included in QualNet

The QualNet distribution includes several sample scenarios for the CPU and Memory Resource Model. All scenarios are located in the directory EXATA_HOME/scenarios/cyber/os-resource-model Table 7-4 lists the sub-directory where each scenario is located.

TABLE 7-4. OS Resource Model Scenarios Included in QualNet

Scenario	Description
connection_memory_recover	Shows the connection memory recover capability in TCP mode
connection-memory-shutdown	Shows the connection memory shutdown capability in TCP mode
cpu_recover	Shows the CPU constraint after CPU recover failure
cpu_shutdown	Shows the CPU constraint after shutdown failure
packet_memory_recover	Shows the packet memory recover capability
packet_memory_shutdown	Shows the packet memory shutdown capability

7.1.7 References

None.