



# ADVISE Meta – Alpha Tool Workshop

## August 16, 2016

Andy Crapo, Brett Feddersen, Alfredo Galbadon, Ken Keefe, Carol Muehrcke, Michael Rausch, Bill Sanders, and Ron Wright

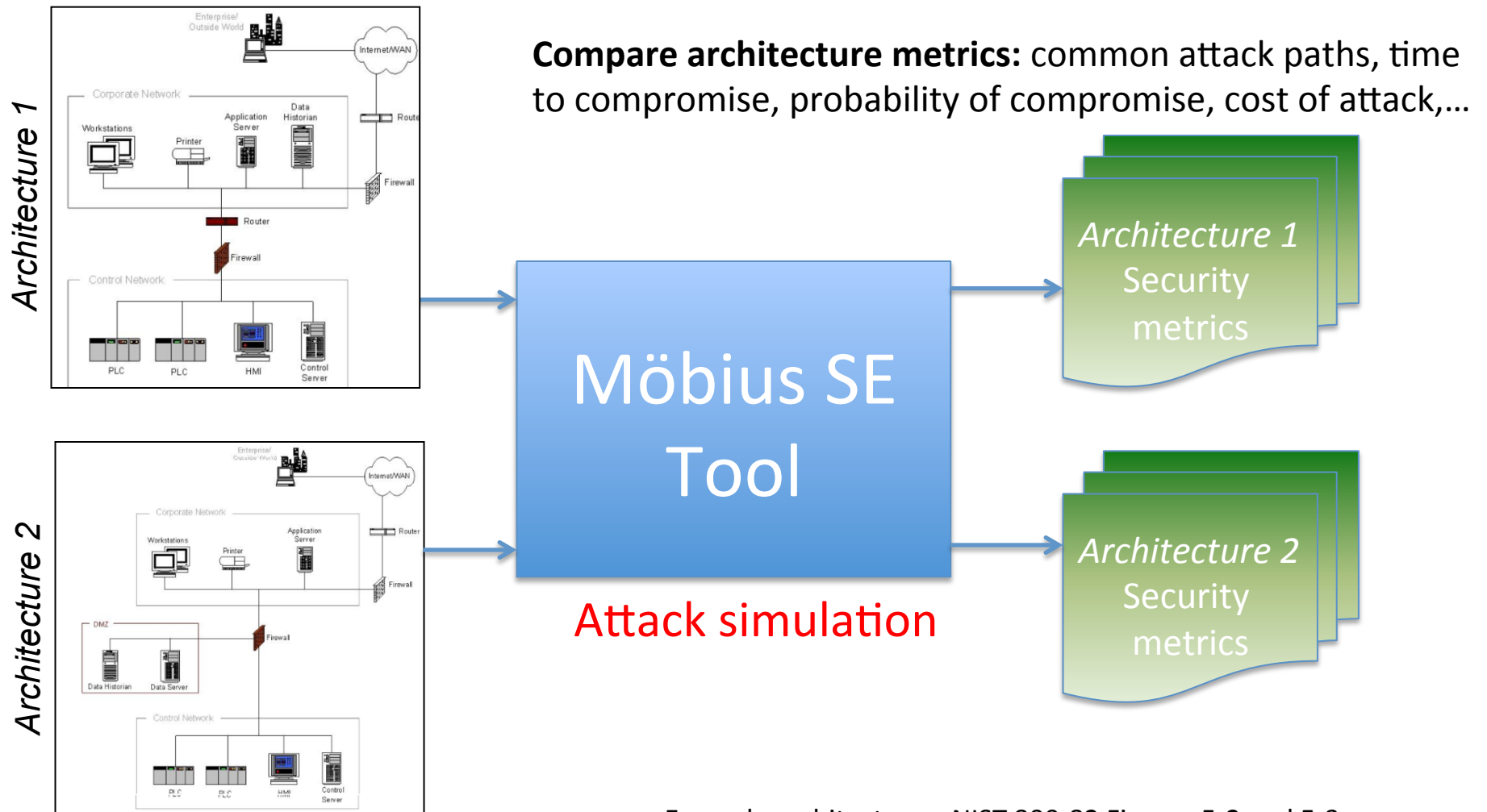
# Agenda

- Registration and Continental Breakfast
- Welcome
- Goals
  - Tool
  - Workshop
- Steps to Use ADVISE Meta
- Hands on Sessions
- Case Studies and Custom Ontologies
- Wrap Up

# ADVISE Meta Introduction

- Today: no scientific basis for designing security architectures
  - Follows from: no scientific basis for estimating effectiveness of security measures before deployment
- Today: security metrics
  - Before deployment, count countermeasures
    - Judge effectiveness based on experience, intuition
  - After deployment, count intrusions
- Purpose of ADVISE Meta
  - Provide scientific basis for design decisions by calculating security metrics at design time
  - Auditable results
  - No requirement for deep modeling or cybersecurity expertise

# ADVISE Meta Tool



Example architectures: NIST 800-82 Figures 5-2 and 5-3



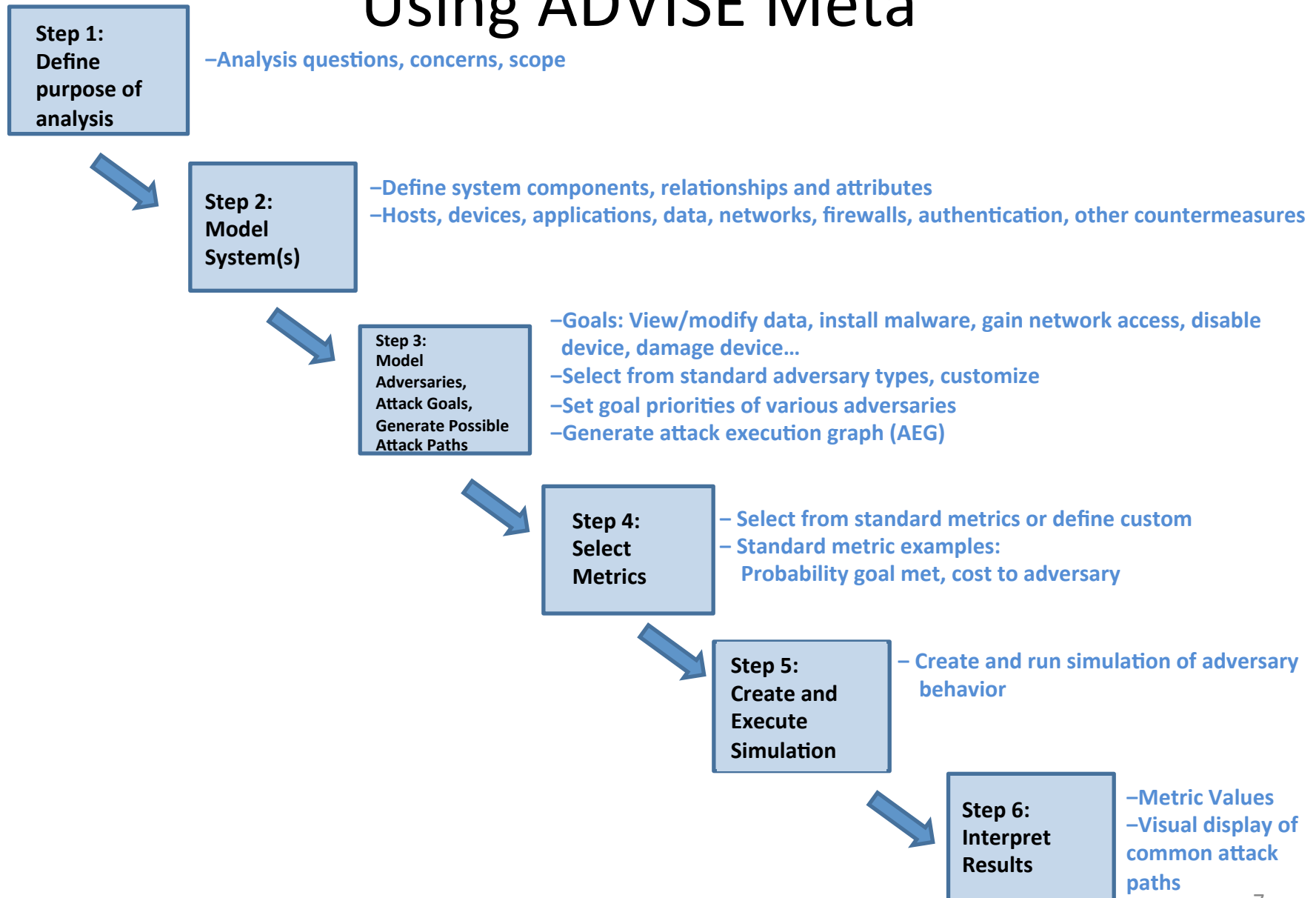
## Workshop Goals

- Introduce the tool to the community
- Gather feedback about all aspects of the tool
  - High level concepts
  - Workflow
  - UI
  - Usability
- Feedback discussion at the end of each hands-on session

# Agenda

- Registration and Continental Breakfast
- Welcome
- Goals
  - Tool
  - Workshop
- Steps to Use ADVISE Meta
- Hands on Sessions
- Case Studies and Custom Ontologies
- Wrap Up

# Using ADVISE Meta



# Hands On Sessions Agenda



# Agenda

- Registration and Continental Breakfast
- Welcome
- Goals
  - Tool
  - Workshop
- Steps to Use ADVISE Meta
- Hands on Sessions
- Case Studies and Custom Ontologies
- Wrap Up

# Step 1 – Define Purpose of Analysis

## What can be analyzed?

- With the base ontology: Enterprise system architectures that may have:
  - Networks hosting cyber and cyber-physical devices
  - Applications, data
  - Internet connections
  - Boundary protections and other common countermeasures
  - Design phase or existing
- With arbitrary ontology:
  - In theory, any system built of components, where attacks are constructed by linking attack steps against components

## What kinds of questions can be answered?

- How susceptible is a system to cyber attacks?
- What can be done to decrease susceptibility?

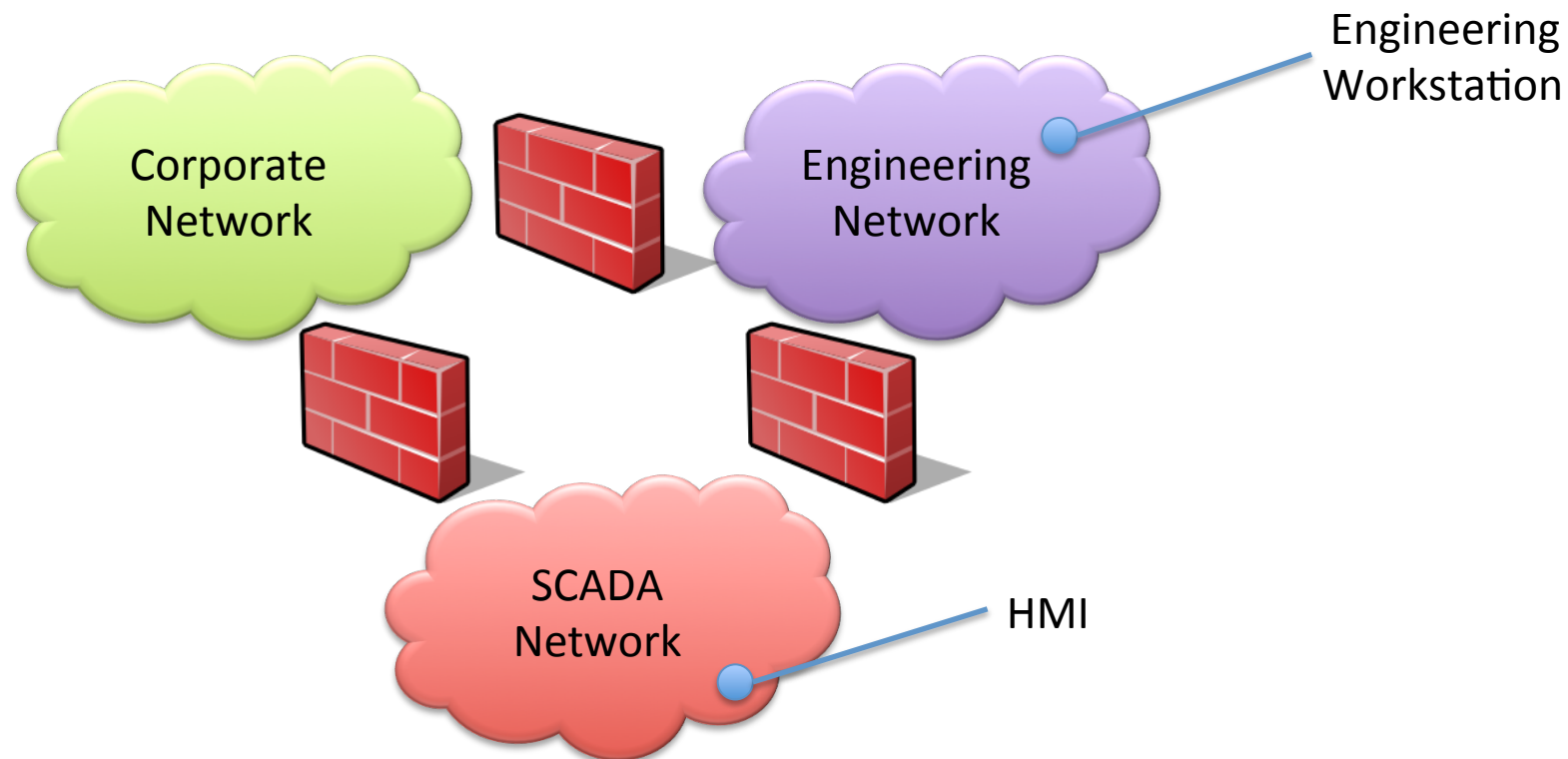
## Typical examples of analyses

- Which among alternative architectures should be recommended?
- What are security weak points of an architecture?
- Is a proposed countermeasure worthwhile?
- How will proposed functional architecture changes impact security?

# Example Enterprise Systems

- Architecture to support stock trading
- SCADA architecture supporting an electric utility
- Control systems in a water treatment plant
- Operations and administration systems for a telecommunications provider
- 911 computer systems architecture
- Reactor safety architecture for a nuclear power plant
- Systems in a hospital that process patient information
- Air traffic or train control systems
- Computer infrastructure for a research and development facility
- Computer infrastructure for an ISP

## Step 1 – Define Purpose of Analysis – Small SCADA Networks



- Predicted electricity demand data stored on the SCADA LAN is found on the desk of an engineer not authorized to view this data. This data can be sold to competing electricity vendors to aid in their pricing.
- How could the engineer even gain access to the SCADA LAN?
- The engineer has physical access to all networks shown and to the HMI, and is an authorized user of the engineering workstation.
- What changes to the architecture would make this less likely to happen again?

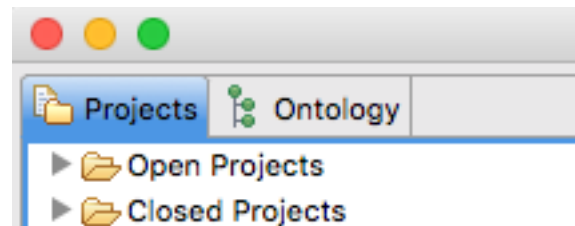


# Step 1 – Define Purpose of Analysis - Feedback

- Open questions/discussion on Step 1
  - Take notes here during meeting
- Questions for group:
  - What are some typical security architecture decisions for which rationale is hard to come by?
  - Are there specific systems that don't fall under enterprise systems, but that might benefit from this type of analysis?

## Step 2 – Model Systems

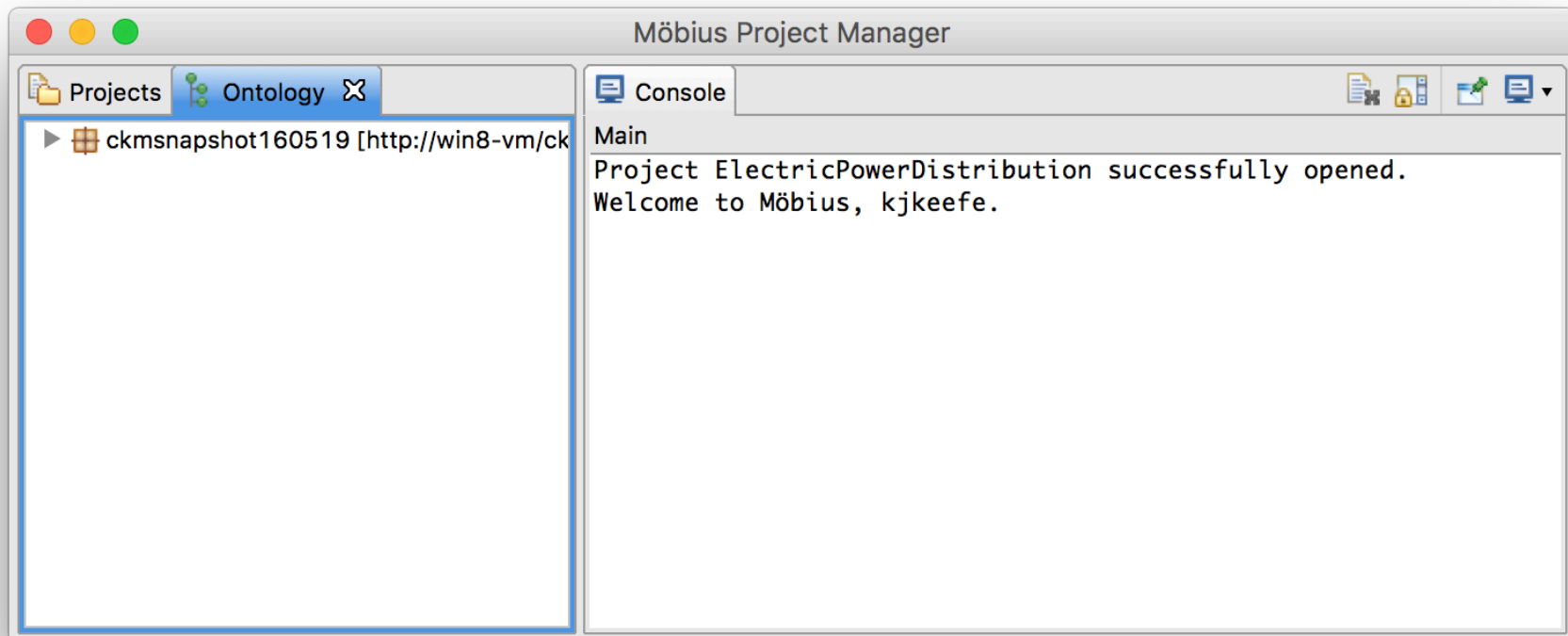
When you start Mobius, you will see two tabs in the upper left hand corner



- The Project tab contains projects with models of systems, attackers, and metrics.
- The Ontology tab is where individual system components, relationships, attributes, etc. are defined.

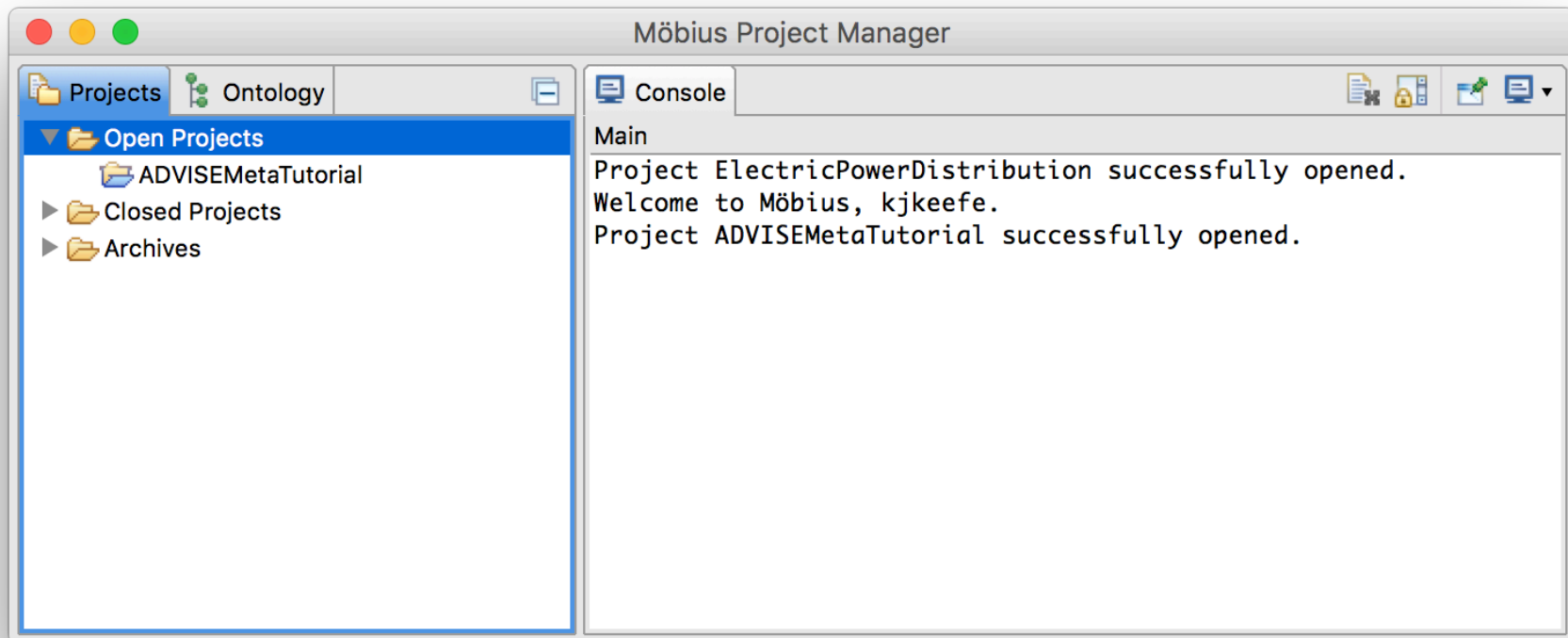
## Step 2 – Model Systems

1. Download the ontology file: <http://go.illinois.edu/am16ont>
2. Right click the Ontology pane.
3. Select Import...
4. Choose the downloaded ontology file



## Step 2 – Model Systems

1. Select the Projects tab.
2. Right click Open Projects and select New Project...
3. Name the project **ADVISEMetaTutorial** and click Finish



## Step 2 – Model Systems

1. Right click on the ADVISEMetaTutorial project and select New...
2. Select the Atomic category in the bottom pane and click Next.
3. Select ADVISE Meta Model, enter the name **MetaModel1**, and click Finish.



# Step 2 – Model Systems

The screenshot displays the MetaModel application window. On the left, there is a sidebar with two main sections: 'Components' and 'Ontology'. The 'Components' section lists 'Data', 'PhysicalThing', and 'System'. The 'Ontology' section has an 'Edit' button and a table with columns 'Name' and 'Version'. The table contains one entry: 'ckmsnapshot16...' with version '2016.07.06.0'. Below the sidebar is a navigation bar with tabs for 'System', 'Goals', 'Adversaries', 'Metrics', 'Configurations', and 'Generator'. The main area of the window is a large white space labeled 'System Model Canvas' in red text. Two red arrows point from the text '1. Available Components' and '2. Loaded Ontologies' to the 'Components' and 'Ontology' sections respectively.

MetaModel

MetaModel

**Components**

- Data
- PhysicalThing
- System

1. Available Components

**System Model Canvas**

**Ontology** Edit

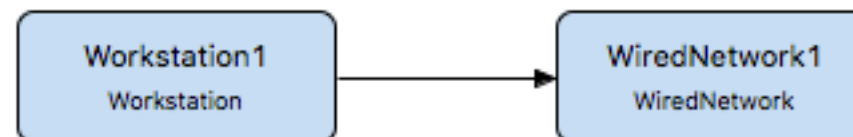
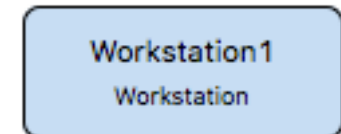
| Name             | Version      |
|------------------|--------------|
| ckmsnapshot16... | 2016.07.06.0 |

2. Loaded Ontologies

System Goals Adversaries Metrics Configurations Generator

## Step 2 – Model Systems

- Component
  - Part or element of the system
  - Physical objects, e.g., computer, firewall, building, etc.
  - Logical objects, e.g., data, software, network, etc.
  - Components are represented as blue rectangles on the system diagram
- Relationship
  - A semantic connection between two components.
  - For example, a computer is connected to a network through a **onNetwork** relationship, or a data is managed by a software application through a **managedBy** relationship.
  - A relationship is represented as arcs on the system diagram.



# Step 2 – Model Systems

- Attributes
  - Properties associated with a component.
  - For example, a component might use a specific type of authentication mechanism.
  - Attributes are listed in the “Details” of the component.

Workstation1 Details

**Component Details**  
Specify the details for the component.

Name:

Attributes

|                                   |   |
|-----------------------------------|---|
| componentAnomalyDetectionStrength | 0 |
| credentialMonitoring              | 0 |
| deviceStatusControl               | 1 |
| deviceStatusDetection             | 2 |
| mediaPortEnabled                  | 1 |
| physicalAttackAttribution         | 2 |
| resistanceToKineticDamage         | 0 |
| resistanceToLogicalDisable        | 0 |
| resistanceToPhysicalDisable       | 0 |
| softwareTrustedSourceSecurity     | 5 |
| softwareTrustedSourceSecurity     | 5 |
| strengthOfUserAuthentication      | 0 |
| userAuthenticationType            | N |
| userCyberSecurityAwareness        | 3 |
| userCyberSecurityAwareness        | 3 |

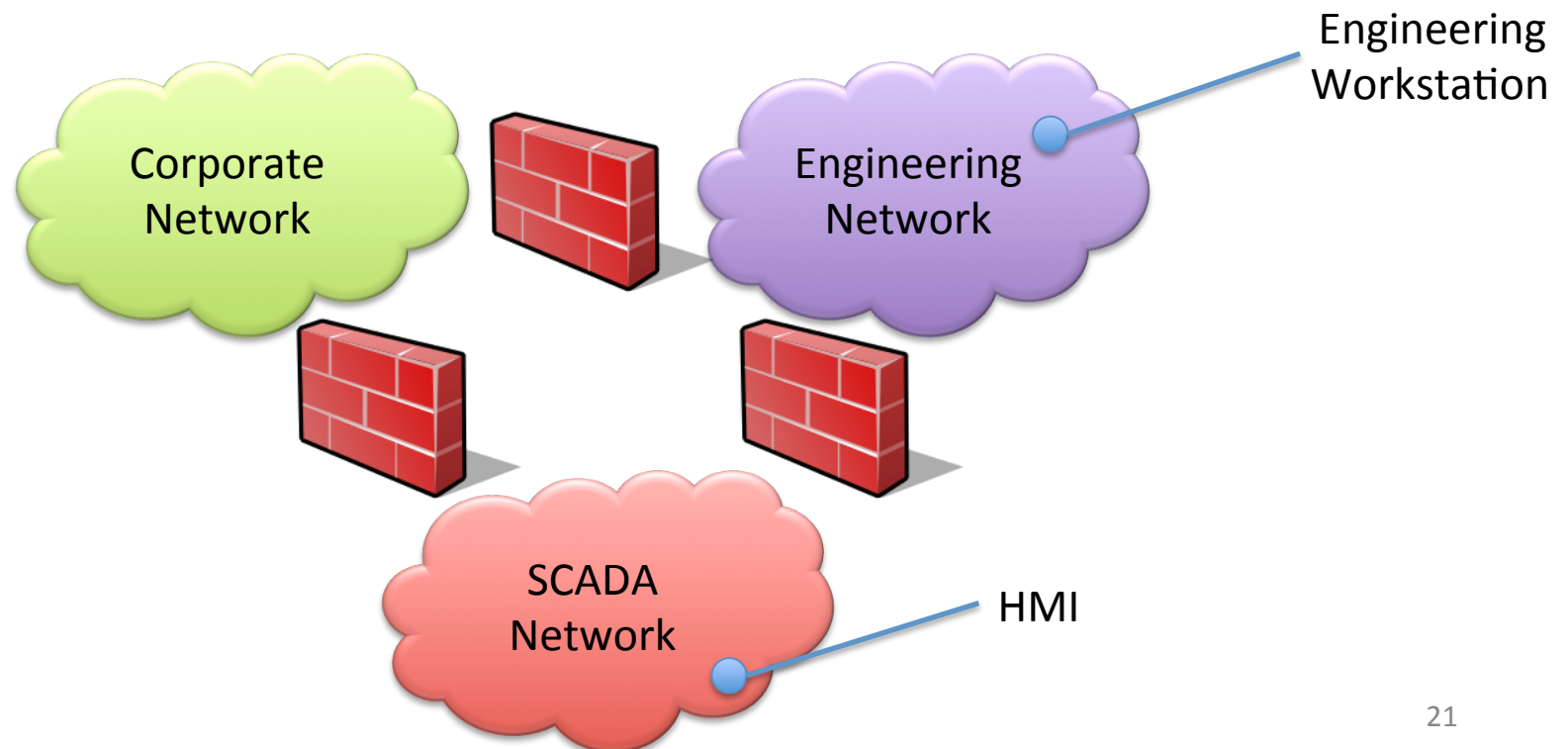
Cancel Finish



## Step 2 – Model Systems

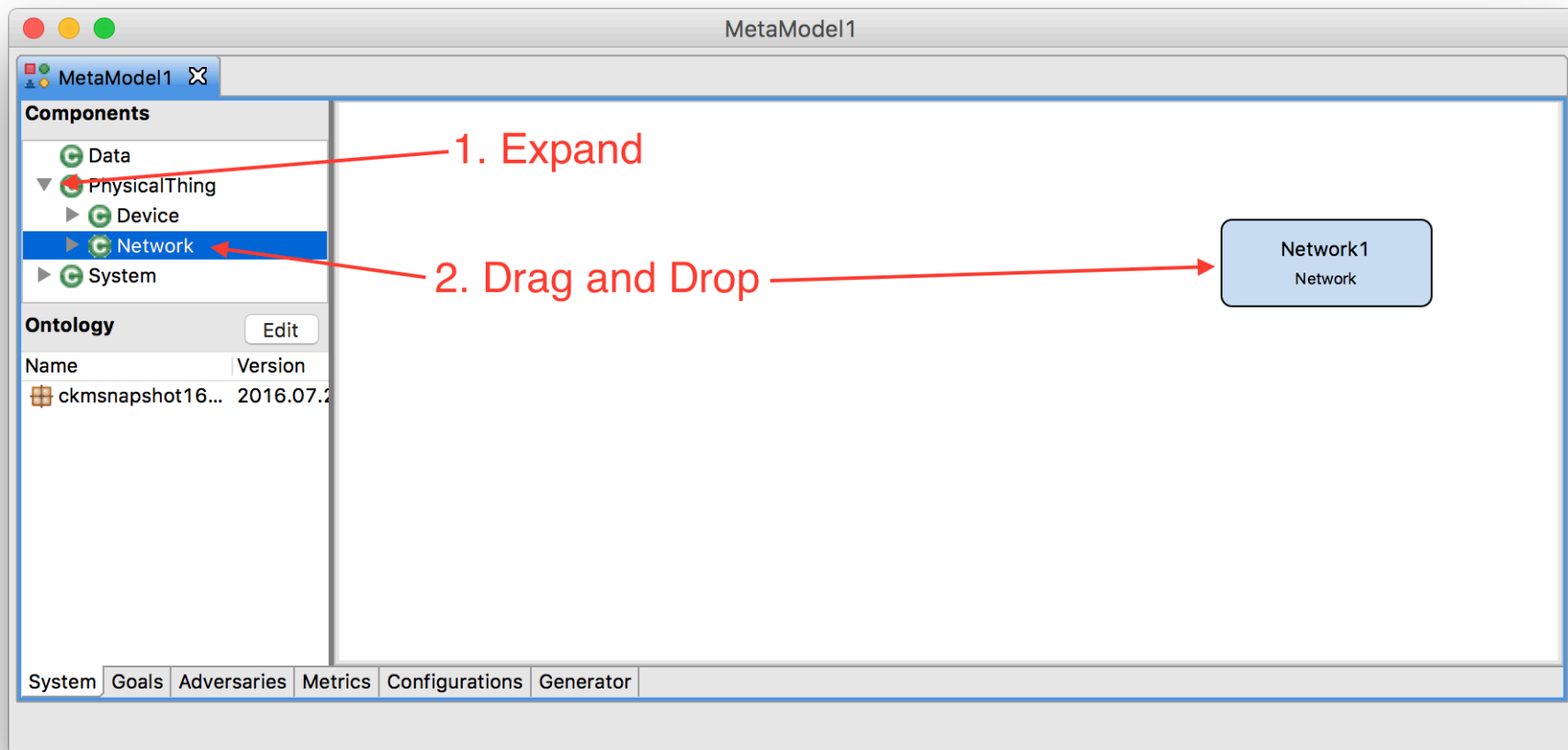
Our simple example consists of:

- SCADA Network with a local terminal (HMI)
- Engineering Network with a local Linux workstation running an SSH server
- Corporate LAN
- All networks are interconnected through firewalls



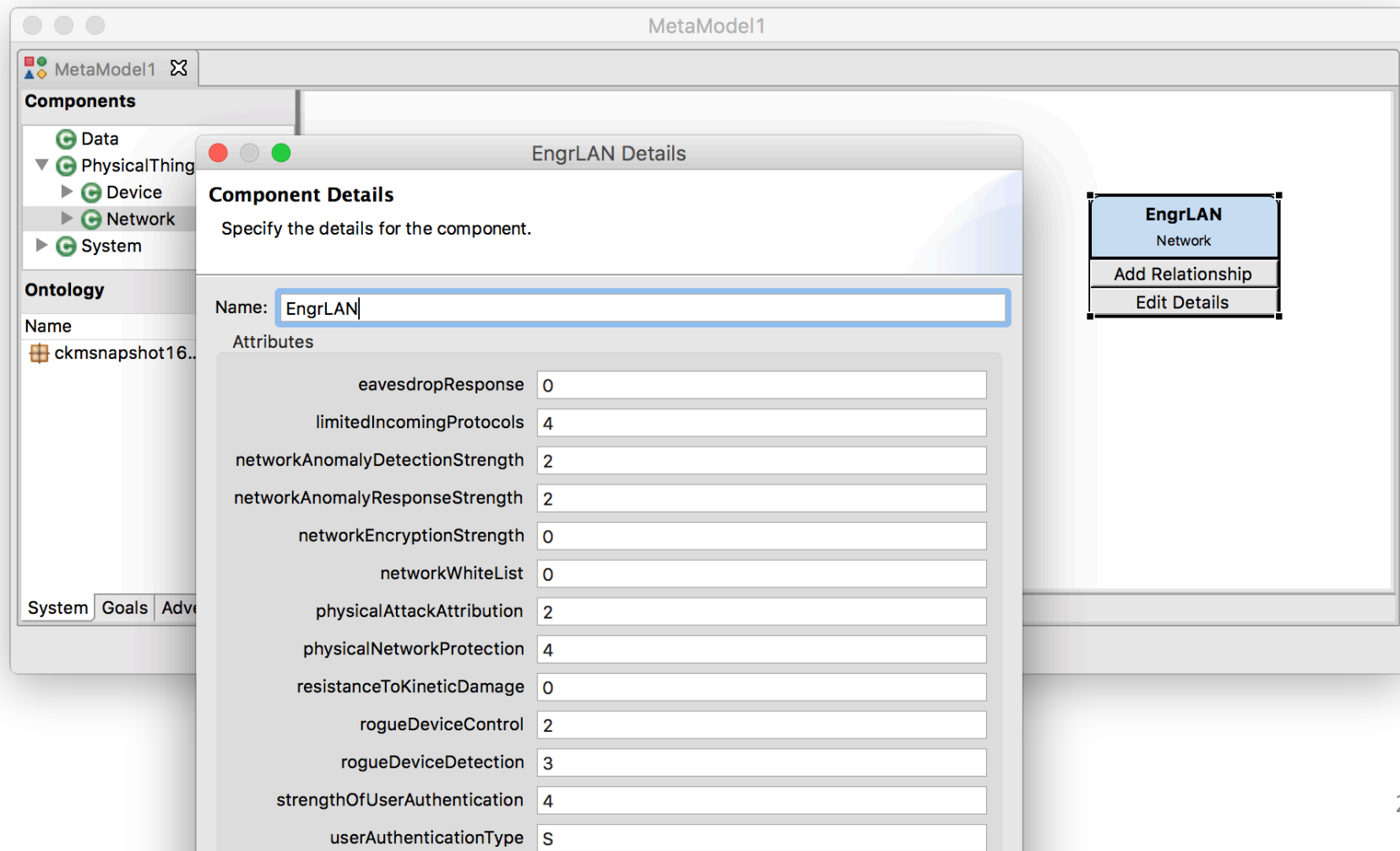
## Step 2 – Model Systems

1. Expand the **PhysicalThing** node in the available components tree.
2. Drag and drop an instance of a **Network** onto the canvas.



# Step 2 – Model Systems

1. Select the **Network1** component and click the Edit Details button.
2. Change the name to **EngrLAN** and click Finish.



MetaModel1

Components

- Data
- PhysicalThing
  - Device
  - Network
  - System

Ontology

Name

ckmsnapshot16...

System Goals Adv

EngrLAN Details

Component Details

Specify the details for the component.

Name:

Attributes

|                                 |                                |
|---------------------------------|--------------------------------|
| eavesdropResponse               | <input type="text" value="0"/> |
| limitedIncomingProtocols        | <input type="text" value="4"/> |
| networkAnomalyDetectionStrength | <input type="text" value="2"/> |
| networkAnomalyResponseStrength  | <input type="text" value="2"/> |
| networkEncryptionStrength       | <input type="text" value="0"/> |
| networkWhiteList                | <input type="text" value="0"/> |
| physicalAttackAttribution       | <input type="text" value="2"/> |
| physicalNetworkProtection       | <input type="text" value="4"/> |
| resistanceToKineticDamage       | <input type="text" value="0"/> |
| rogueDeviceControl              | <input type="text" value="2"/> |
| rogueDeviceDetection            | <input type="text" value="3"/> |
| strengthOfUserAuthentication    | <input type="text" value="4"/> |
| userAuthenticationType          | <input type="text" value="S"/> |

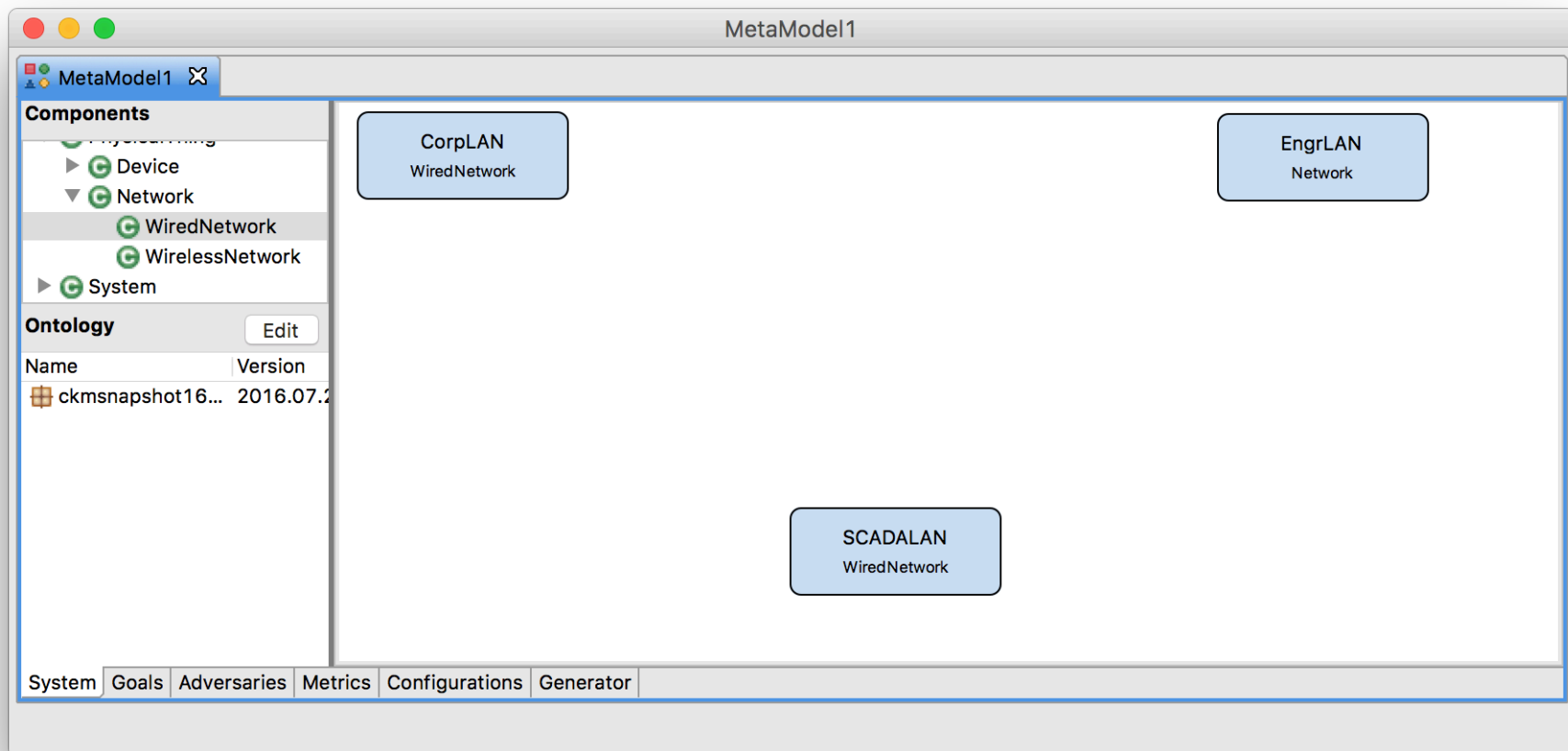
EngrLAN  
Network

Add Relationship

Edit Details

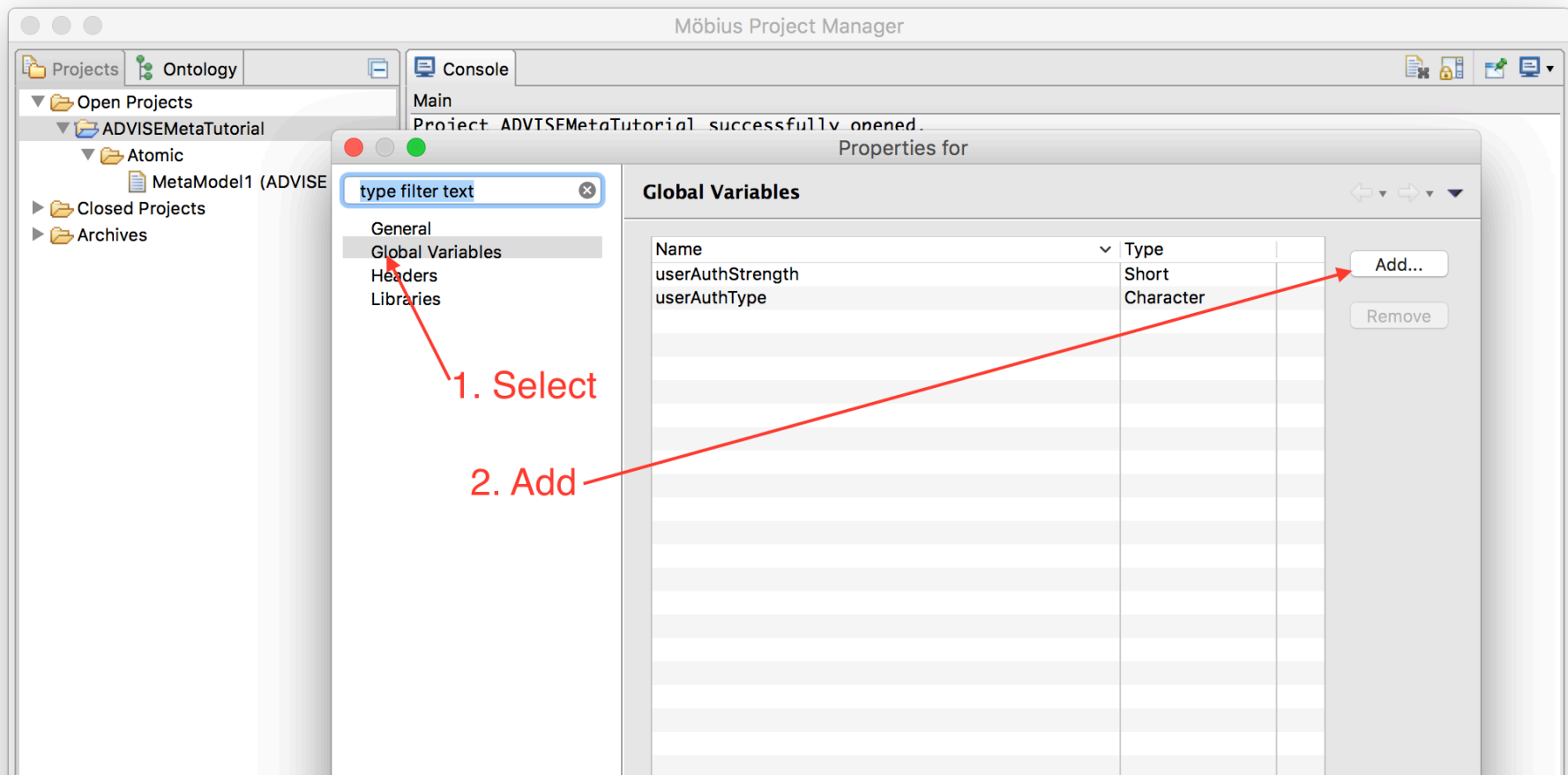
# Step 2 – Model Systems

1. Create a **WiredNetwork** called **CorpLAN**
2. Create a **WiredNetwork** called **SCADALAN**



## Step 2 – Model Systems

1. Right click the project **ADVISEMetaTutorial** and select **Properties...**
2. Select the Global Variables section.
3. Add a new variable called **userAuthType** with type **Character**.
4. Add a new variable called **userAuthStrength** with type **Short**.



## Step 2 – Model Systems

1. Create a **FirewallHosted** and define its attributes like so:

| Attribute                    | Value             |
|------------------------------|-------------------|
| Name                         | CorpLanScadaLanFW |
| strengthOfUserAuthentication | userAuthStrength  |
| userAuthenticationType       | userAuthType      |

The screenshot shows a dialog box titled "CorpLanScadaLanFW Details" with a "Component Details" section. The "Name" field is set to "CorpLanScadaLanFW". Below, a list of attributes is shown with corresponding values. The "strengthOfUserAuthentication" and "userAuthenticationType" attributes are highlighted with a red circle.

| Attribute                         | Value            |
|-----------------------------------|------------------|
| componentAnomalyDetectionStrength | 0                |
| credentialMonitoring              | 0                |
| deviceStatusControl               | 1                |
| deviceStatusDetection             | 2                |
| firewallConfigControl             | 2                |
| firewallConfigDetection           | 5                |
| mediaPortEnabled                  | 1                |
| physicalAttackAttribution         | 2                |
| resistanceToKineticDamage         | 0                |
| resistanceToLogicalDisable        | 0                |
| resistanceToPhysicalDisable       | 0                |
| softwareTrustedSourceSecurity     | 5                |
| softwareTrustedSourceSecurity     | 5                |
| strengthOfUserAuthentication      | userAuthStrength |
| userAuthenticationType            | userAuthType     |
| userCyberSecurityAwareness        | 3                |

## Step 2 – Model Systems

### 1. Create another **FirewallHosted**

| Attribute                    | Value            |
|------------------------------|------------------|
| Name                         | CorpLanEngrLanFW |
| strengthOfUserAuthentication | userAuthStrength |
| userAuthenticationType       | userAuthType     |

### 2. Create another **FirewallHosted**

| Attribute                    | Value             |
|------------------------------|-------------------|
| Name                         | EngrLanScadaLanFW |
| strengthOfUserAuthentication | userAuthStrength  |
| userAuthenticationType       | userAuthType      |

## Step 2 – Model Systems

1. Select the **CorpLanEngrLanFW** component and click the Add Relationship button.
2. Select the **CorpLan** component.
3. Check that the **onNetwork** relationship is selected and click Finish.

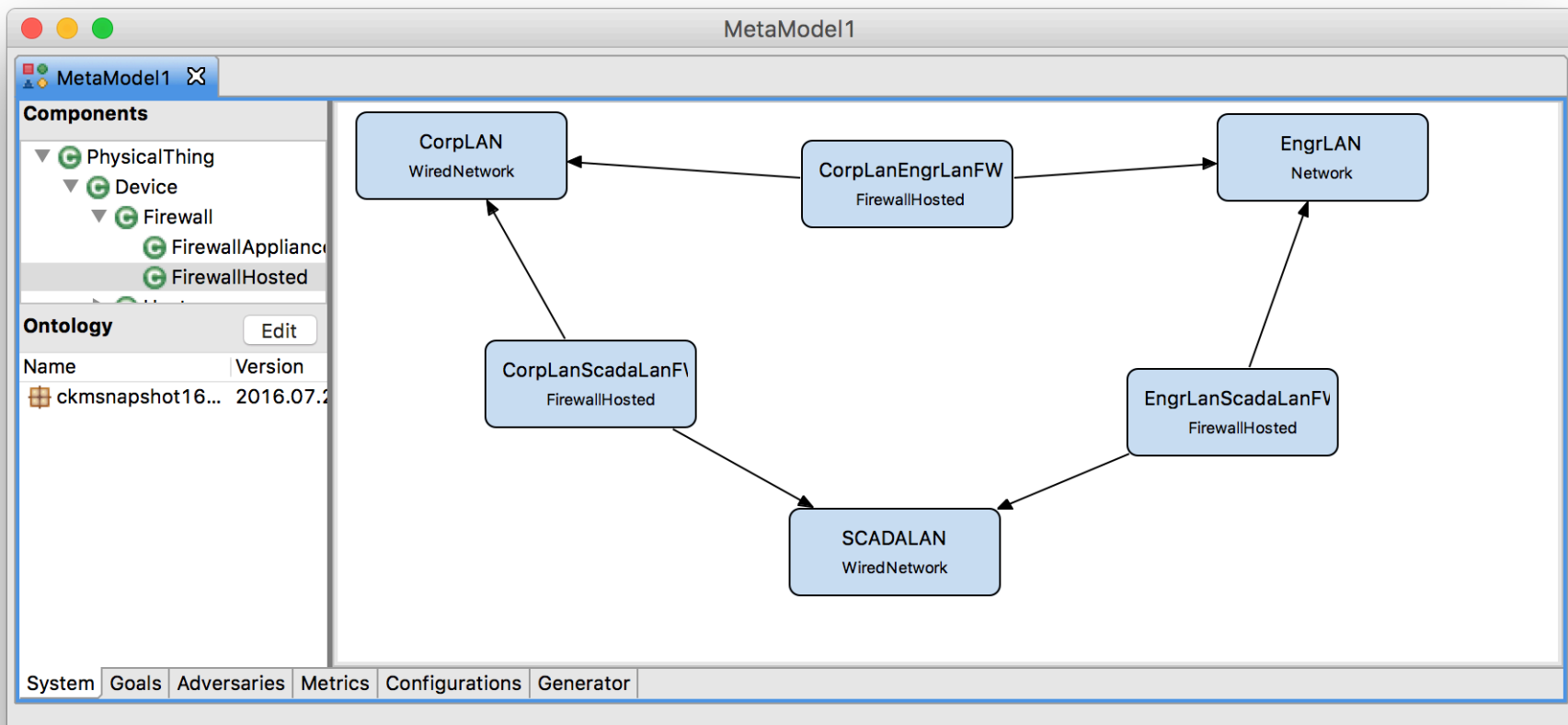
The screenshot displays the MetaModel1 interface with a dialog box open for adding a relationship. The dialog box, titled "Add Relationship", prompts the user to "Select the relationship you wish to add between the source and target components." The "Source" is set to "CorpLanEngrLanFW" (FirewallHosted) and the "Target" is set to "CorpLAN". The "Relationship" dropdown menu is set to "onNetwork". The "Add Relationship" dialog box is overlaid on the main interface, which shows a diagram with components: CorpLAN (WiredNetwork), CorpLanEngrLanFW (FirewallHosted), CorpLanScadaLanF (FirewallHosted), and SCADALAN (WiredNetwork). The "Components" panel on the left shows a tree view with "PhysicalThing" expanded to "Device" and "Firewall". The "Ontology" panel shows a table with columns "Name" and "Version".

| Name             | Version   |
|------------------|-----------|
| ckmsnapshot16... | 2016.07.2 |



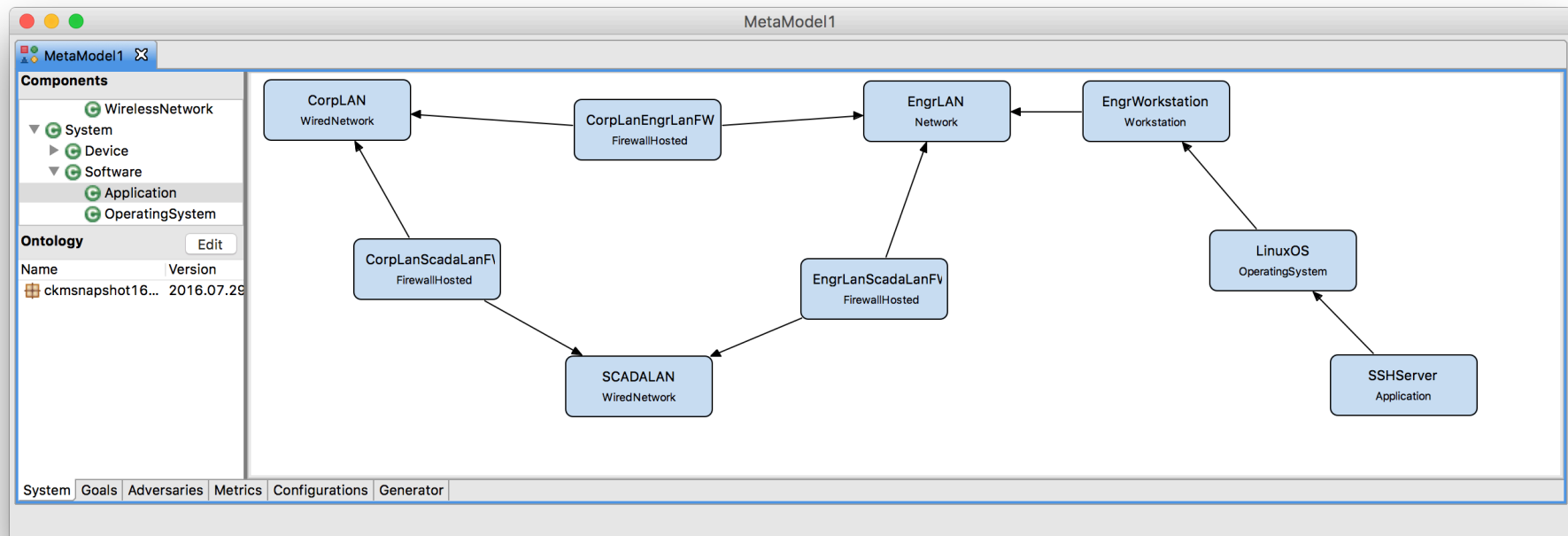
# Step 2 – Model Systems

1. Create additional **onNetwork** relationships between:
  - **CorpLanEngrLanFW** to **EngrLAN**
  - **CorpLanScadaLanFW** to **CorpLAN**
  - **CorpLanScadaLanFW** to **SCADALAN**
  - **EngrLanScadaLanFW** to **EngrLAN**
  - **EngrLanScadaLanFW** to **SCADALAN**



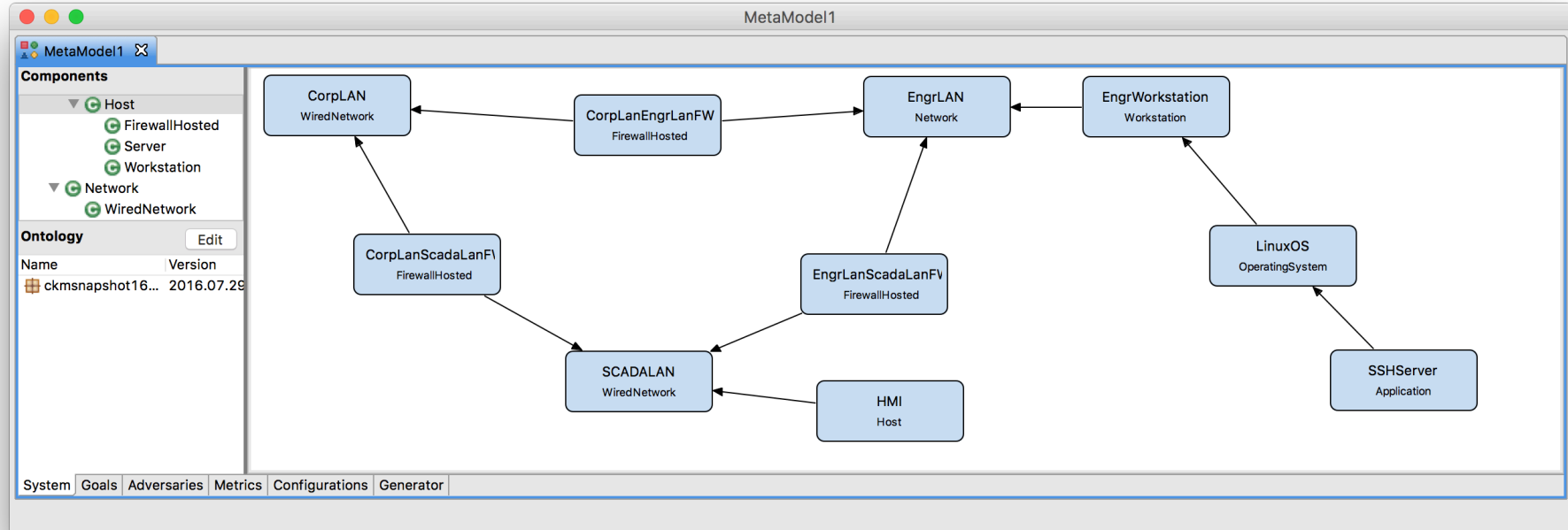
## Step 2 – Model Systems

1. Create a **Workstation** called **EngrWorkstation**
2. Define an **onNetwork** relationship from **EngrWorkstation** to **EngrLAN**.
3. Create an **OperatingSystem** called **LinuxOS**.
4. Define a **hardwarePlatform** relationship from **LinuxOS** to **EngrWorkstation**.
5. Create an **Application** called **SSHServer**.
6. Define an **applicationOS** relationship from the **SSHServer** to **LinuxOS**.



# Step 2 – Model Systems

1. Create a **Host** called **HMI**.
2. Define an **onNetwork** relationship from **HMI** to **SCADALAN**



## Step 2 – Model Systems – Feedback

- How challenging was this step?
- Was adding components, creating relationships, and defining attributes easy?
- Could this part of the tool be useful for designing system diagrams for uses outside of the tool?
- How would you handle larger, more complex models? How would you expect the tool to help you with those models?
- Was the available components tree intuitive?

## Step 3 – Attack Goals, Adversaries, and Generation

- Possible attack goals are dependent on the system diagram
  - Choose a set of state variables (Access, Skill, Knowledge, SSV) the goal state is a function of.
  - Define the functional expression that indicates whether the goal has been achieved.

# Small SCADA Networks – Step 3

```
return SCADALAN_NetworkAccess->Mark();
```

MetaModel

Available State Variables

- CorpLanEngrLanFW
- CorpLanScadaLanFW
- EngrLan
- EngrLanScadaLanFW
- EngrWorkstation
- HMI
- LinuxOS
- SCADALAN
  - Damaged
  - HasUserCredentials
  - MalwareInstalledOn
  - NetworkAccess**
  - PhysicalAccess
  - SpecializedTechnicalHardwareExpertise
- SSHServer

Ontology

| Name           | Version          | URL       |
|----------------|------------------|-----------|
| ckmsnapshot... | 2016.07.28.06... | http://wi |

Goals

Add Delete

Name

Goal\_GainNetworkAccessOnScadaNetwork

1. Click Add

2. Define Name

Name: Goal\_GainNetworkAccessOnScadaNetwork

Dependent State Variables:

Name

SCADALAN\_NetworkAccess

3. Drag and Drop

Goal Expression:

return SCADALAN\_NetworkAccess->Mark();

4. Enter Expression

## Step 3 – Attack Goals, Adversaries, and Generation

- Adversaries are created from Adversary Templates (defined in the ontology)
  - Attributes are customizable
  - Possible initial state depends on system diagram

# Small SCADA Networks – Step 3

**1. Drag and Drop**

**2. Specify Name and Decision Parameters**

**3. Specify Initial Access**

**4. Specify Skills and Goals**

**MetaModel**

**Adversary Templates**

- Customer
- EconomicCompetitorLimited
- EconomicCompetitorWellRes
- ForeignGovernmentLimitedI
- ForeignGovernmentWellRes
- HackerGroup
- IndependentInsider**
- OrganizedCrime
- TerroristOrganization

**Adversaries** [Delete]

Name: EngineerInsider

**Decision Parameters**

Planning Horizon: 5

Cost of Detection: 25000

**Access**

| Name                              | Initial Value |
|-----------------------------------|---------------|
| SSHServer_HasUserCredentials      | 1             |
| LinuxOS_HasUserCredentials        | 1             |
| EngWorkstation_HasUserCredentials | 1             |

**Knowledge**

| Name | Initial Value |
|------|---------------|
|------|---------------|

**Skills**

| Name              | Initial Value |
|-------------------|---------------|
| BasicCyberOffense | 1000          |
| Cryptanalysis     | 200           |

**Ontology** [Edit]

| Name           | Version      |
|----------------|--------------|
| ckmsnapshot... | 2016.07.28.0 |

System Goals Adversaries Metrics Configurations Generator



# Small SCADA Networks – Step 3

1. Independent Insider
2. Name: EngineerInsider  
 Planning Horizon: 5  
 Cost of Detection: 20000
3. Access:
  - CorpLAN\_PhysicalAccess
  - CorpLanEngrLanFW\_PhysicalAccess
  - CorpLanScadaLanFW\_PhysicalAccess
  - EngrLAN\_NetworkAccess
  - EngrLAN\_PhysicalAccess
  - EngrLanScadaLanFW\_PhysicalAccess
  - EngrWorkstation\_HasUserCredentials
  - EngrWorkstation\_LogicalAccess
  - EngrWorkstation\_PhysicalAccess
  - EngrWorkstation\_UIAccess
  - InsiderAccess
  - HMI\_PhysicalAccess
  - LinuxOS\_HasUserCredentials
  - LinuxOS\_LogicalAccess
  - LinuxOS\_UIAccess
  - SCADALAN\_PhysicalAccess
  - SSHServer\_HasUserCredentials
  - SSHServer\_LogicalAccess
  - SSHServer\_UIAccess
4. Goal: Goal\_GainNetworkAccessOnScadaNetwork 80000

# Small SCADA Networks – Step 3

The screenshot displays the MetaModel1 configuration window. The interface is divided into several sections:

- Configuration List:** A list on the left shows 'WeakFirewalls' selected. A red arrow labeled '1. Select Configuration' points to this list.
- Name:** A text field contains 'WeakFirewalls'. A red arrow labeled '2. Specify Name' points to this field.
- Description:** An empty text area for describing the configuration.
- Goals:** A section with an 'Add...' button and a 'Remove' button. A dropdown menu is open, showing 'Goal\_GainNetworkAccessOnSc...'. A red arrow labeled '3. Add Goal' points to this dropdown.
- Adversary:** A dropdown menu showing 'EngineerInsider'. A red arrow labeled '4. Select Adversary' points to this dropdown.
- Metrics:** An empty text area for defining metrics, with 'Add' and 'Remove' buttons to its right.

At the bottom, a navigation bar includes tabs for System, Goals, Adversaries, Metrics, Configurations, and Generator.

# Small SCADA Networks – Step 3

The screenshot displays the MetaModel1 application window. The interface is divided into several sections:

- Destination Project:** A text field containing "ADVISEMetaTutorial" and a "Select..." button.
- Configurations:** A list box with a dropdown arrow, currently showing "WeakFirewalls" selected. A red arrow points to this selection with the text "1. Select Configuration".
- Component Prefix:** An empty text field.
- Component Suffix:** An empty text field.
- Generate Components:** A group box containing four checked checkboxes: "ADVISE Atomic Model", "Performance Variables Model", "Range Study", and "Simulator".
- Buttons:** "Generate" and "Stop" buttons are located at the bottom of the configuration area. A red arrow points to the "Generate" button with the text "2. Click Generate".
- Status:** A text area on the right side of the window showing the following log output:

```
Constructing complete attack execution graph... Complete! (0.006sec)
Trimming AEG for WeakFirewalls configuration...Complete! (0.082sec)
New ADVISE model: WeakFirewalls
Creating Performance Variables Model...Complete! (0.137sec)
Creating Range Study...Complete! (0.087sec)
Creating Simulator...Complete! (0.101sec)
```
- Navigation:** A tabbed interface at the bottom with tabs for "System", "Goals", "Adversaries", "Metrics", "Configurations", and "Generator".

## Step 3 – Attack Goals, Adversaries, and Generation – Feedback

- How challenging was this step?
- Did the goal definition seem intuitive?
- What other adversary templates would you look for?
- Does the configuration of an adversary's attributes make sense?

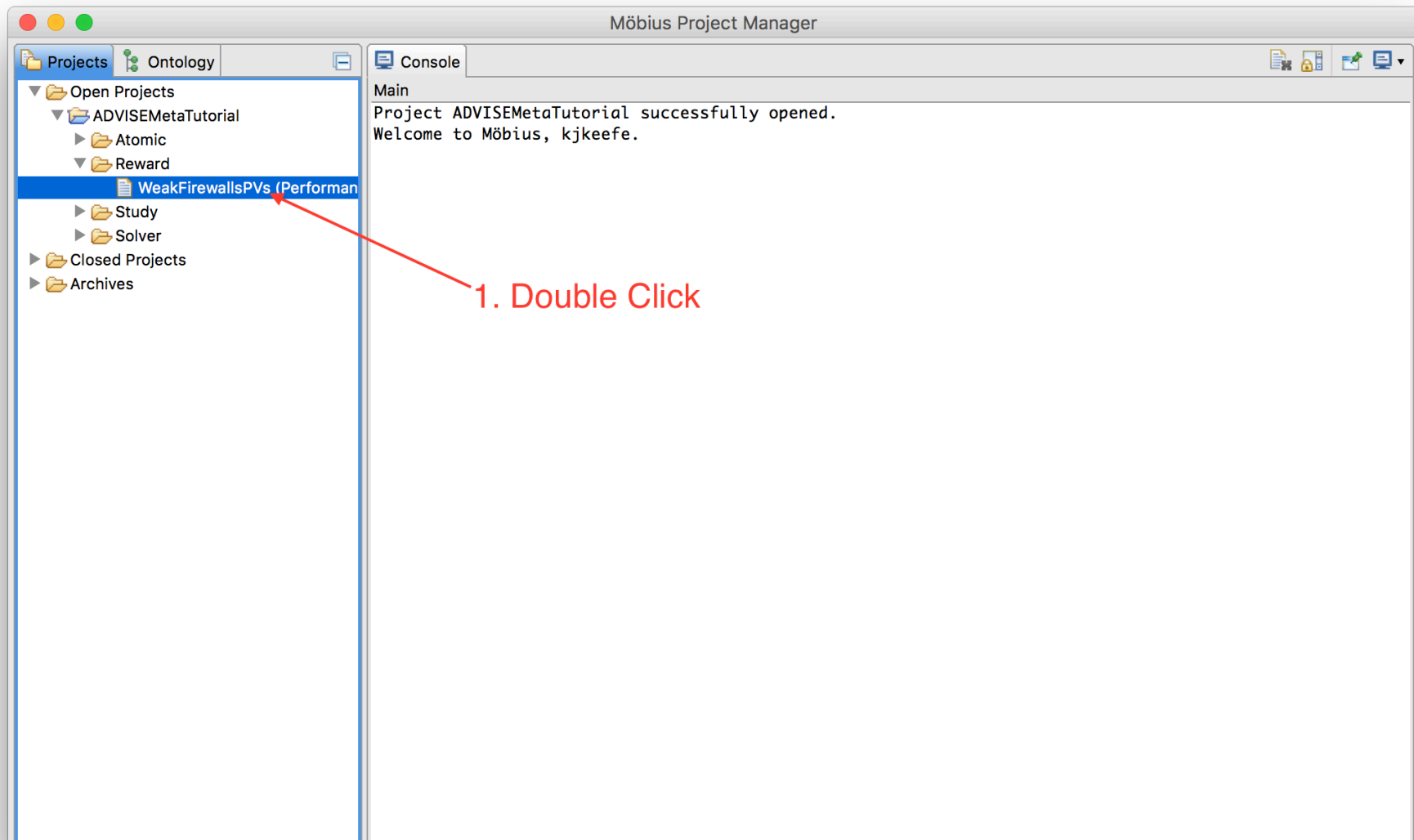
## Step 4 – Metrics and Experiments

- Performance Variables
  - Rate-based
  - Event-based
  - Time instants, intervals, and steady state
- Example Metrics
  - Goal Achieved – Gain Network Access on SCADA Net
  - Time – Instants (0, 10, 20, 30, 40, 50, 60)
- Study
  - Set of experiments
  - Varying global variable values (initial model parameters)

# Small SCADA Networks – Step 4

Open the Reward Model

- Double click **WeakFirewallsPVs**



# Small SCADA Networks – Step 4

## Create a new Performance Variable

- Enter the variable name **GoalAchieved**, click Add Variable

ADVISER MetaTutorial: WeakFirewallsPVs

File Edit Help

Performance Variables Model

GoalAchieved

Add Variable:

Variable List

1. Enter PV Name

2. Click

Variable Name:

Submodels Rate Rewards Impulse Rewards Time ▶

Available Submodels  
(Select the models used for this reward variable)

Rename Copy Delete Up Down

Apply Changes Discard Changes

Möbius Performance Variable Editor 2.5  
WeakFirewallsPVs

PERFORM

# Small SCADA Networks – Step 4

## Enter Rate Reward Expression

- Click the Rate Rewards tab
- Enter the expression:
  - `return WeakFirewalls->Goal_GainNetworkAccessOnScadaNetwork->Mark();`

ADVISEMetaTutorial: WeakFirewallsPVs

File Edit Help

Performance Variables Model

Add Variable:

Variable List

GoalAchieved

1. Select Tab

Variable Name: GoalAchieved

Submodels Rate Rewards Impulse Rewards Time Simulation

Available State Variables (double click to insert)

WeakFirewalls->CorpLanEngrLanFW\_LogicalAccess  
WeakFirewalls->SCADALAN\_PhysicalAccess

Reward Function

return WeakFirewalls->Goal\_GainNetworkAccessOnScadaNetwork->Mark();

2. Enter Expression

Rename Copy Delete Up Down

Apply Changes Discard Changes

Möbius Performance Variable Editor 2.5  
Model WeakFirewallsPVs

PERFORM



# Small SCADA Networks – Step 4

## Define Timing Instants

- Click the Time tab, Change the method to Incremental Range, Set Upper Bound to 24 and Step Size to 2.

The screenshot shows the ADVISEMetaTutorial: WeakFirewallsPVs interface. The window title is "ADVISEMetaTutorial: WeakFirewallsPVs". The menu bar includes "File", "Edit", and "Help". The interface is divided into two main sections: "Performance Variables" and "Model".

In the "Performance Variables" section, there is a text input field for "(Enter new variable name)", an "Add Variable:" button, and a "Variable List" containing "GoalAchieved".

In the "Model" section, the "Variable Name" is "GoalAchieved". The "Time" tab is selected, showing the following configuration:

- Type: Instant of Time
- Time Point definition method: Incremental Range
- First time point in series: 0.0
- Upper Bound of series: 24
- Step size in series: 2
- Length of time interval: 0.0
- Number of Time Measurements: 13
- Time Series: 0.0, 2.0, 4.0, ... 20.0, 22.0, 24.0

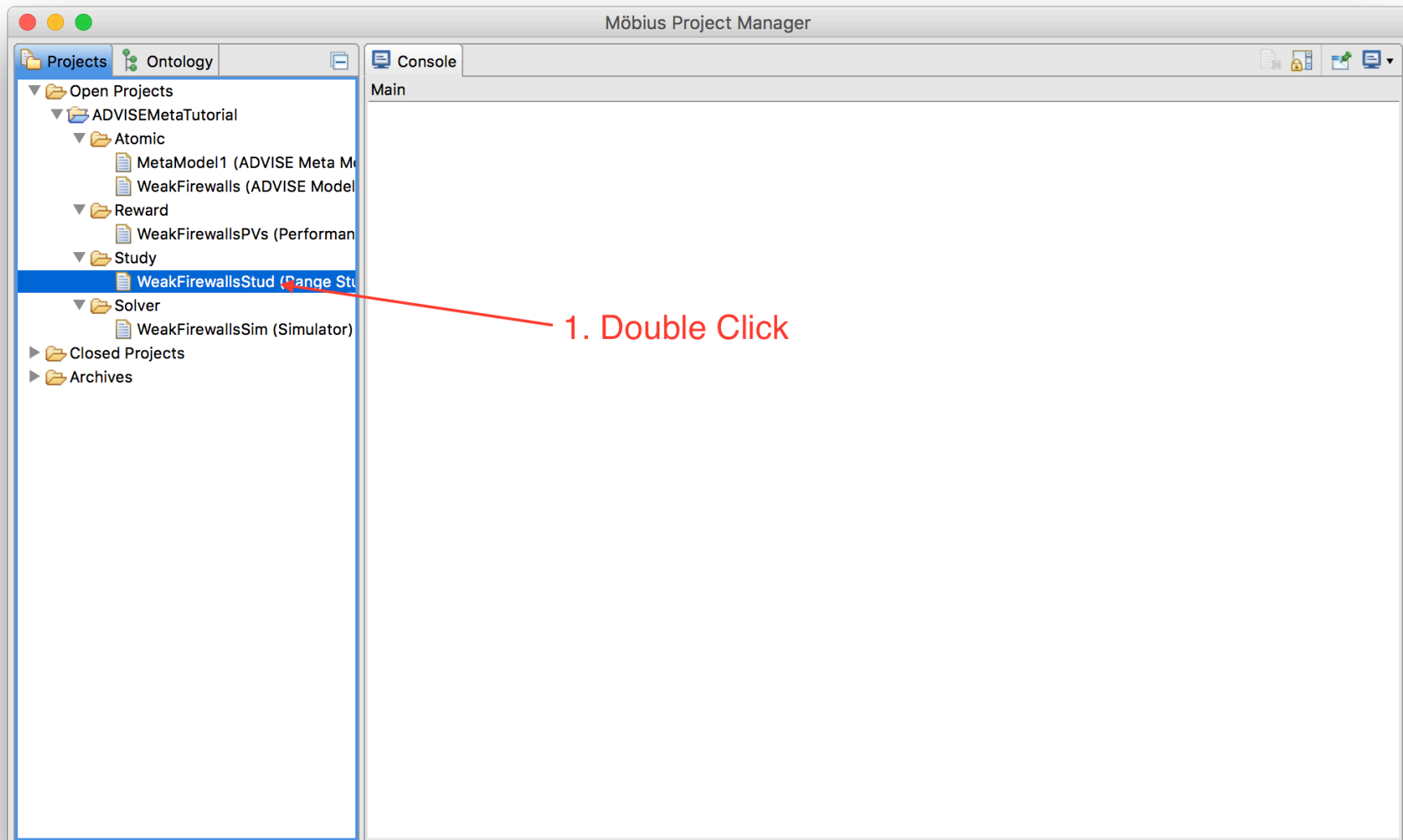
Red arrows and text annotations highlight the changes:

- "1. Change" points to the "Incremental Range" dropdown menu.
- "2. Change" points to the "Upper Bound of series" and "Step size in series" input fields.

# Small SCADA Networks – Step 4

Open the study

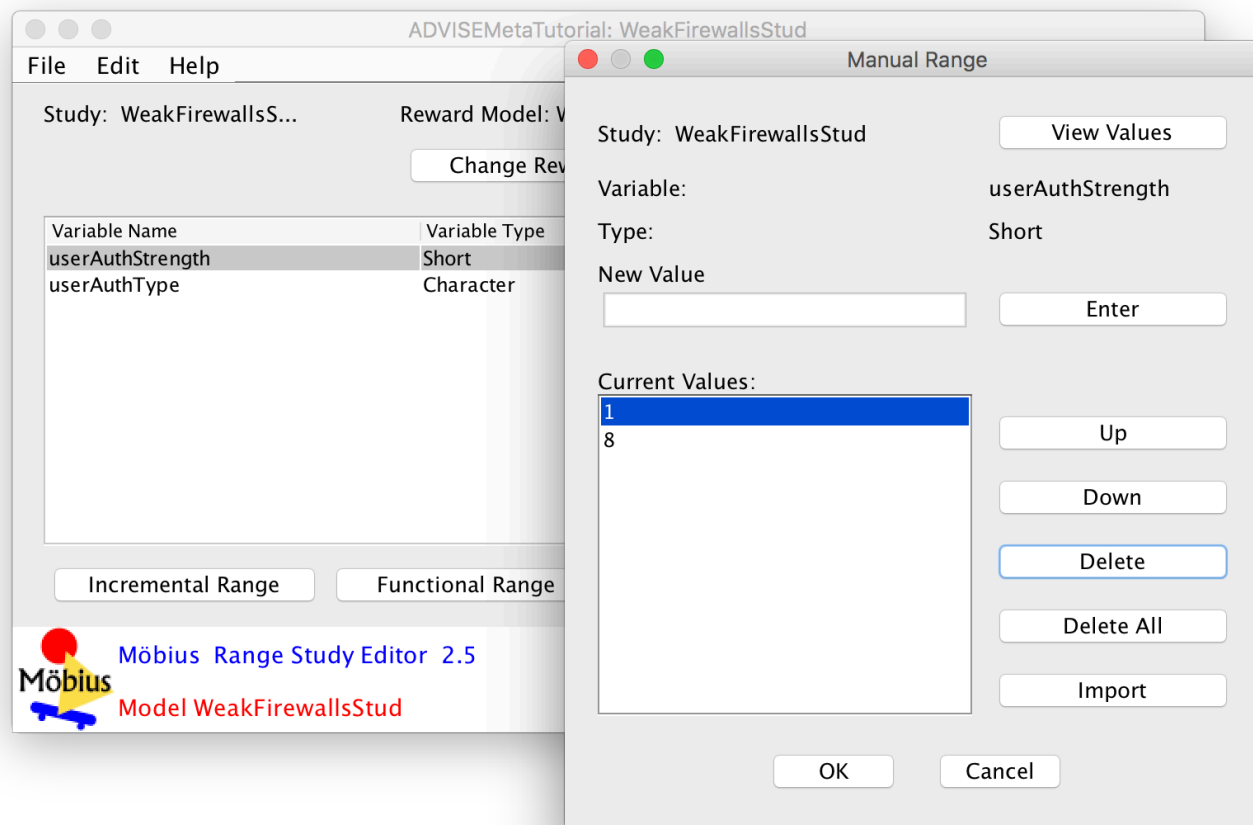
1. Double click **WeakFirewallsStud**.



# Small SCADA Networks – Step 4

Change the values for **userAuthStrength**.

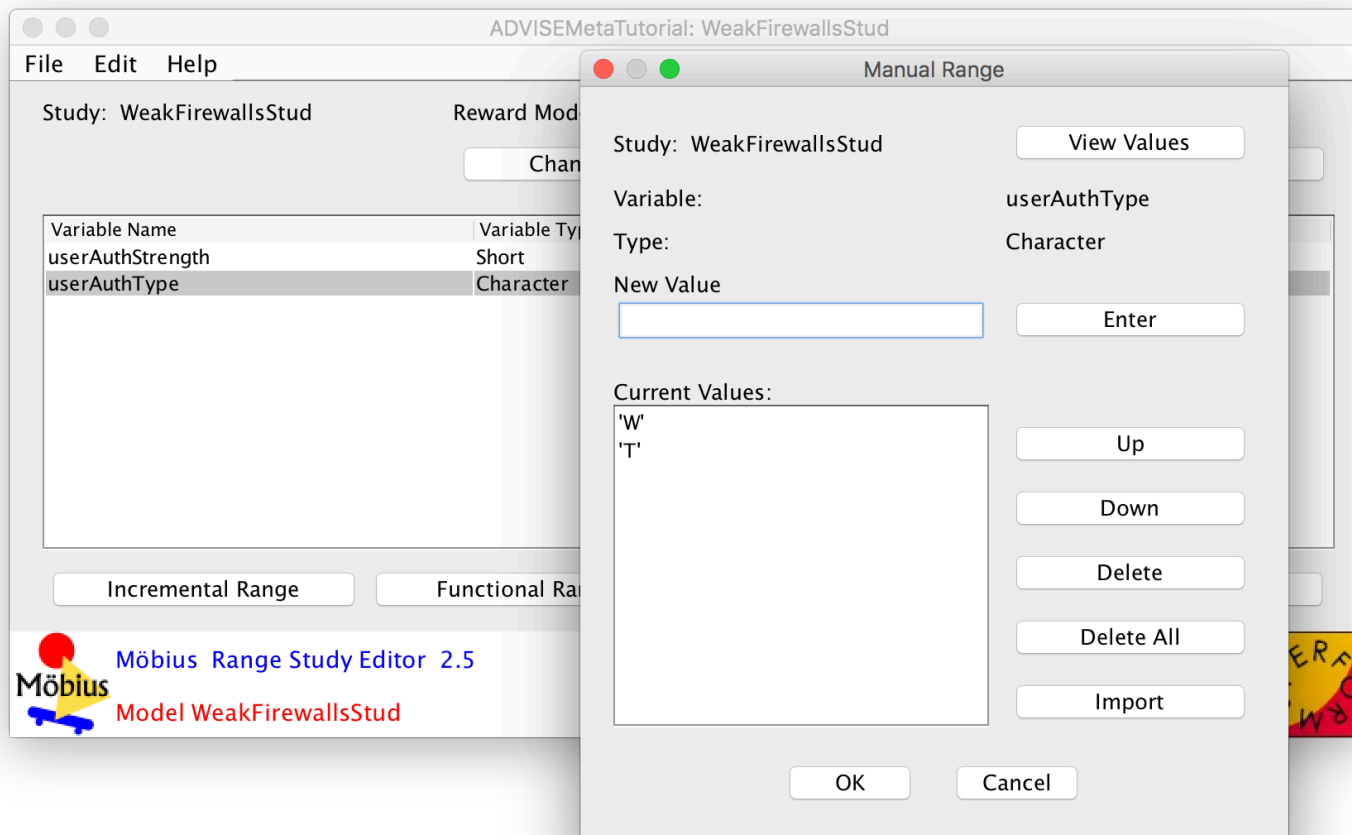
1. Select **userAuthStrength** and click Manual Range.
2. Input **1** in the New Value box and click Enter.
3. Input **8** in the New Value box and click Enter.
4. Select the default **0** and click Delete.



# Small SCADA Networks – Step 4

Change the values for **userAuthType**.

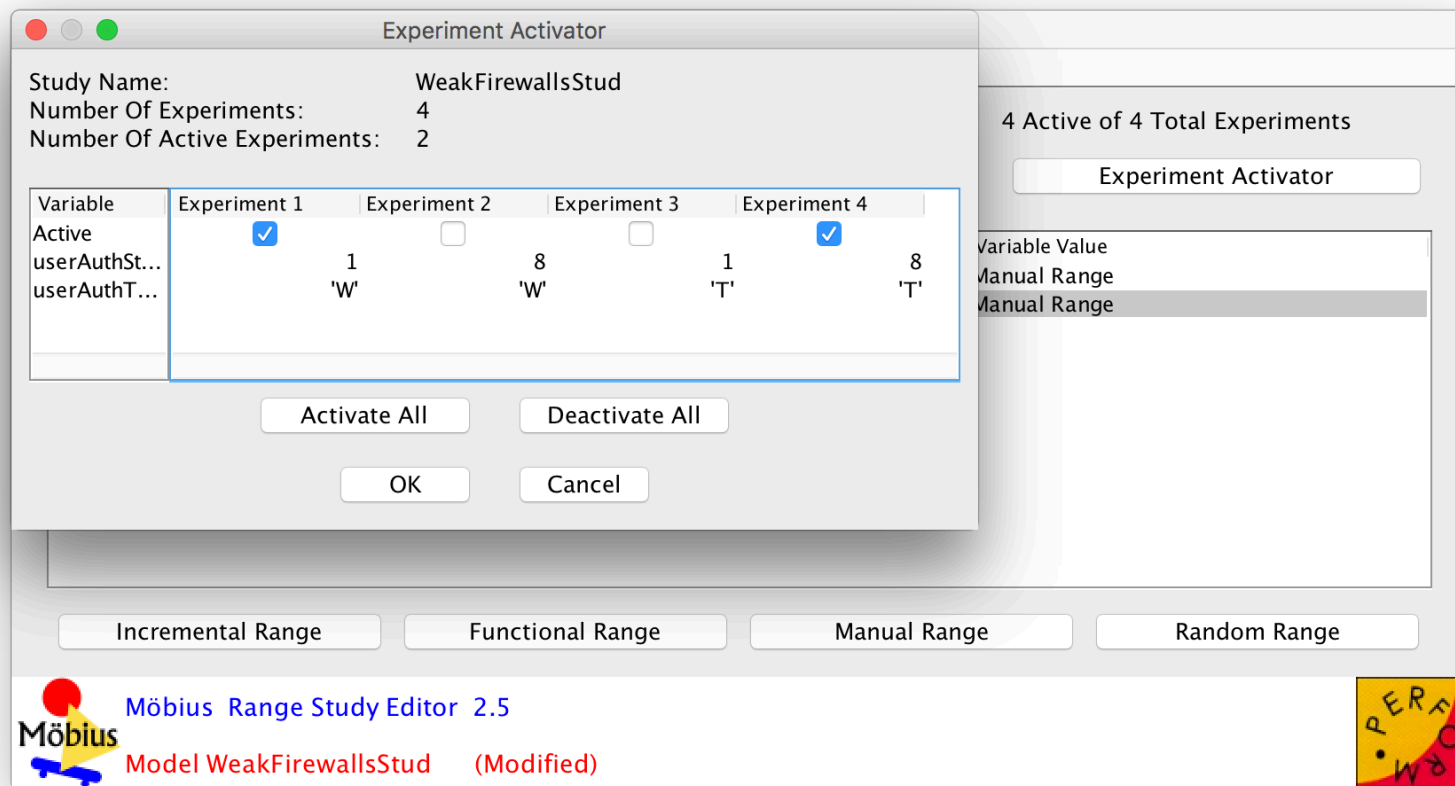
1. Select **userAuthType** and click Manual Range.
2. Input **'W'** in the New Value box and click Enter.
3. Input **'T'** in the New Value box and click Enter.
4. Select the default **0** and click Delete.



# Small SCADA Networks – Step 4

## Deactivate Unnecessary Experiments

1. Click the Experiment Activator button.
2. Uncheck experiments 2 and 3.
3. Click OK.

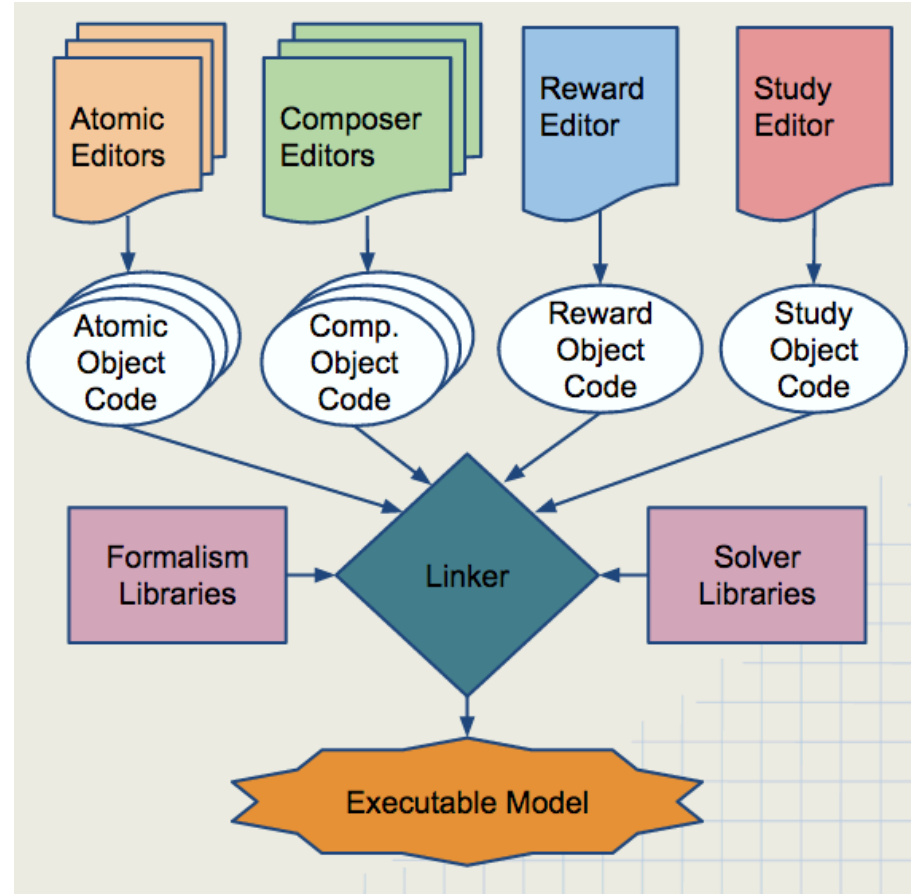


## Step 4 – Metrics and Experiments – Feedback

- How challenging was this step?
- What kind of metrics would you want to define in a system?
- How could you define it as a performance variable?
- What results do you expect to see once the model executes?
- Do you understand how you can develop many experiments?

## Step 5 – Execute Models

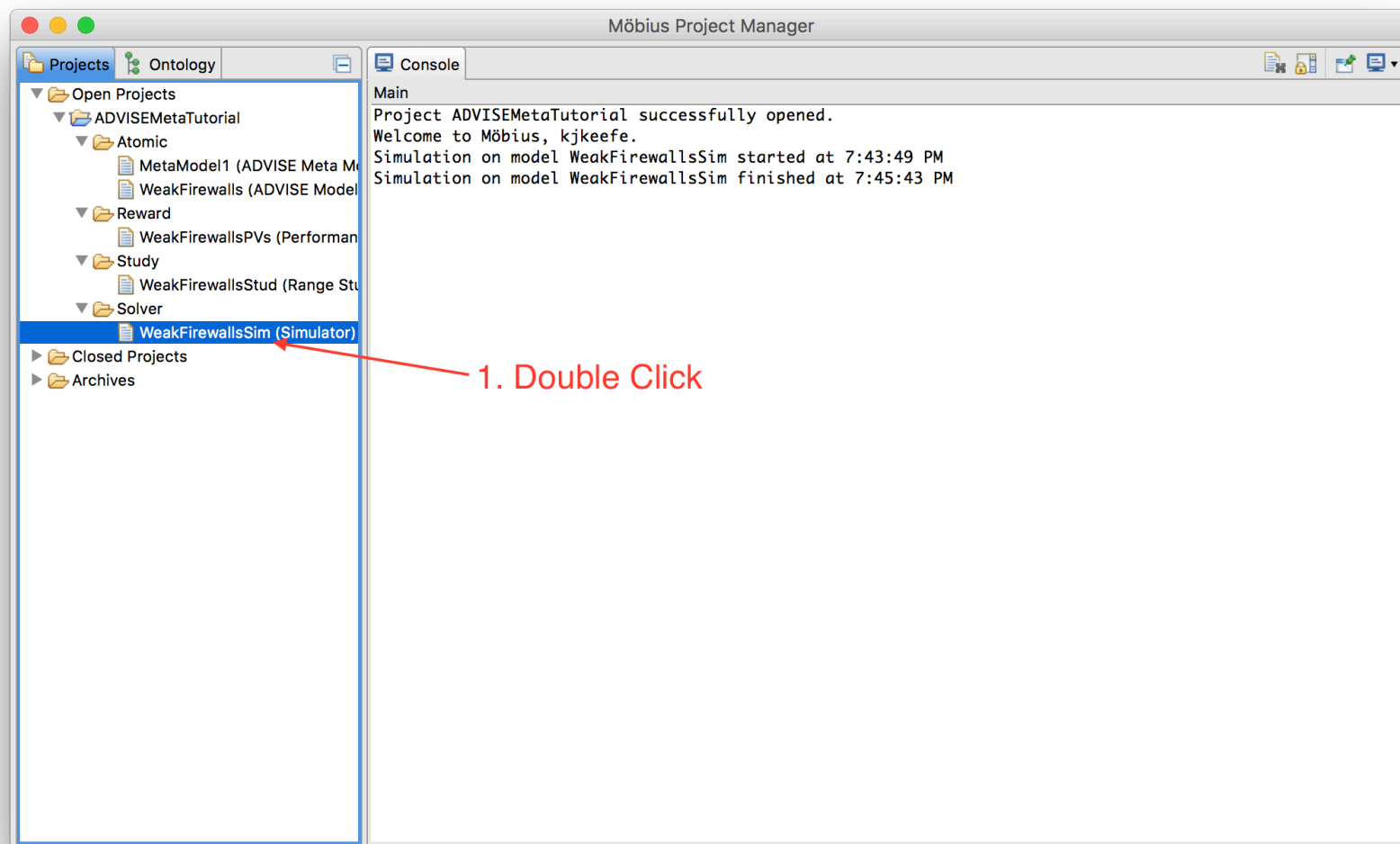
- Möbius creates an executable model by:
  - Generating C++ code representations of project models
  - Compiling the code and linking formalism and solver libraries
  - Executing the binary to gather observations and calculate statistics



# Small SCADA Networks – Step 5

## Open Simulator

1. Double click the **WeakFirewallsSim**.

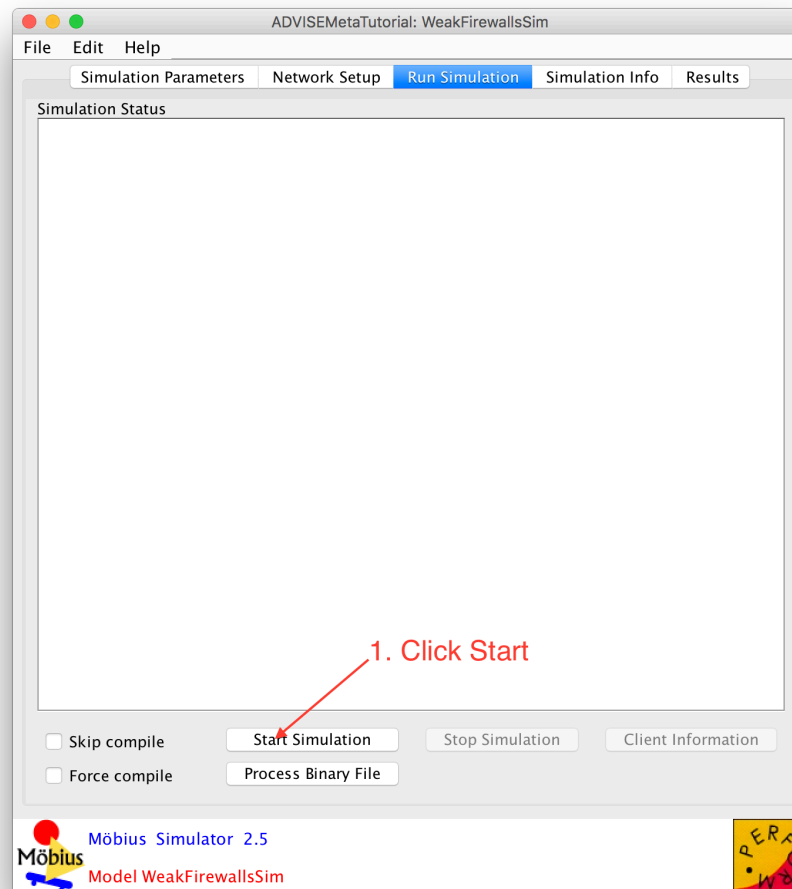




# Small SCADA Networks – Step 5

## Run Simulation

1. Click the Start Simulation button.



# Small SCADA Networks – Step 5

## Simulation Info

1. Wait.

The screenshot shows the ADVISEMetaTutorial: WeakFirewallsSim application window. The window has a menu bar (File, Edit, Help) and a tabbed interface with tabs for Simulation Parameters, Network Setup, Run Simulation, Simulation Info (selected), and Results. The Simulation Info tab contains a table with the following data:

| Experiment   | Status     | # CPUs | Batches |
|--------------|------------|--------|---------|
| Experiment 1 | Running    | 1      | 0       |
| Experiment 4 | No Results | 0      | 0       |

Below the table are three buttons: "Terminate selected Experiment", "Terminate All Experiments", and "Show Results".

Selected Experiment: Experiment 1  
Running      Batches: 0      Running Time: 0.0 seconds

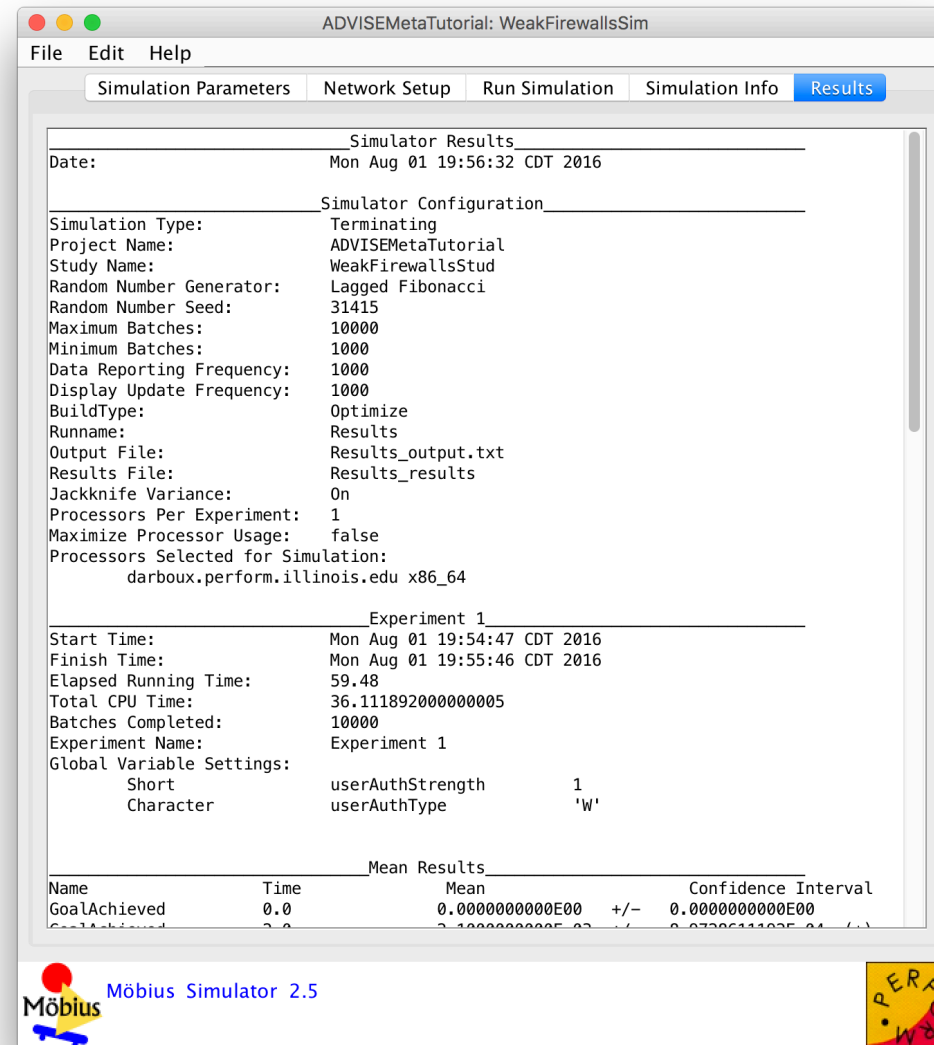
Below this is a table with the following data:

| Performance Variable Name | Time       | Mean | Mean's Confidence Interval |
|---------------------------|------------|------|----------------------------|
| GoalAchieved              | Inst: 0.0  | 0.0  | +/- 0.0                    |
| GoalAchieved              | Inst: 2.0  | 0.0  | +/- 0.0                    |
| GoalAchieved              | Inst: 4.0  | 0.0  | +/- 0.0                    |
| GoalAchieved              | Inst: 6.0  | 0.0  | +/- 0.0                    |
| GoalAchieved              | Inst: 8.0  | 0.0  | +/- 0.0                    |
| GoalAchieved              | Inst: 10.0 | 0.0  | +/- 0.0                    |
| GoalAchieved              | Inst: 12.0 | 0.0  | +/- 0.0                    |
| GoalAchieved              | Inst: 14.0 | 0.0  | +/- 0.0                    |
| GoalAchieved              | Inst: 16.0 | 0.0  | +/- 0.0                    |
| GoalAchieved              | Inst: 18.0 | 0.0  | +/- 0.0                    |
| GoalAchieved              | Inst: 20.0 | 0.0  | +/- 0.0                    |
| GoalAchieved              | Inst: 22.0 | 0.0  | +/- 0.0                    |
| GoalAchieved              | Inst: 24.0 | 0.0  | +/- 0.0                    |

At the bottom of the window, there is a logo for Möbius Simulator 2.5 and a small graphic with the text "PERTORM".

# Small SCADA Networks – Step 5

## Simulation Is Complete



ADVISEMetaTutorial: WeakFirewallsSim

File Edit Help

Simulation Parameters Network Setup Run Simulation Simulation Info Results

Simulator Results

Date: Mon Aug 01 19:56:32 CDT 2016

Simulator Configuration

Simulation Type: Terminating  
Project Name: ADVISEMetaTutorial  
Study Name: WeakFirewallsStud  
Random Number Generator: Lagged Fibonacci  
Random Number Seed: 31415  
Maximum Batches: 10000  
Minimum Batches: 1000  
Data Reporting Frequency: 1000  
Display Update Frequency: 1000  
BuildType: Optimize  
Runname: Results  
Output File: Results\_output.txt  
Results File: Results\_results  
Jackknife Variance: On  
Processors Per Experiment: 1  
Maximize Processor Usage: false  
Processors Selected for Simulation:  
darboux.perform.illinois.edu x86\_64

Experiment 1

Start Time: Mon Aug 01 19:54:47 CDT 2016  
Finish Time: Mon Aug 01 19:55:46 CDT 2016  
Elapsed Running Time: 59.48  
Total CPU Time: 36.111892000000005  
Batches Completed: 10000  
Experiment Name: Experiment 1  
Global Variable Settings:  
Short userAuthStrength 1  
Character userAuthType 'W'

Mean Results

| Name            | Time | Mean            | Confidence Interval |
|-----------------|------|-----------------|---------------------|
| GoalAchieved    | 0.0  | 0.0000000000E00 | +/- 0.0000000000E00 |
| GoalNotAchieved | 0.0  | 0.1000000000E00 | +/- 0.0700000000E00 |

Möbius Simulator 2.5

PERFORM

## Step 5 – Execute Models – Feedback

- How challenging was this step?
- Did the simulation run faster/slower than you expected?
- Were the steps to execute the simulation too complex?

## Step 6 – Interpret Results

- We will examine...
  - Numerical results from the simulation
  - A visual presentation of the model's behavior

# Small SCADA Networks – Step 6

## Numerical Results

- Adversary was more successful, more quickly when firewalls were hardened.

```

batches Completed:      10000
Experiment Name:        Experiment 1
Global Variable Settings:
  Short Character      userAuthStrength      1
                       userAuthType        'W'
  
```

| Mean Results |      |                  |                          |
|--------------|------|------------------|--------------------------|
| Name         | Time | Mean             | Confidence Interval      |
| GoalAchieved | 0.0  | 0.0000000000E00  | +/- 0.0000000000E00      |
| GoalAchieved | 2.0  | 2.1000000000E-03 | +/- 8.9728611192E-04 (*) |
| GoalAchieved | 4.0  | 2.8200000000E-02 | +/- 3.2448213338E-03 (*) |
| GoalAchieved | 6.0  | 9.8300000000E-02 | +/- 5.8356008775E-03     |
| GoalAchieved | 8.0  | 1.8260000000E-01 | +/- 7.5726082533E-03     |
| GoalAchieved | 10.0 | 2.5560000000E-01 | +/- 8.5499070658E-03     |
| GoalAchieved | 12.0 | 3.2230000000E-01 | +/- 9.1606582755E-03     |
| GoalAchieved | 14.0 | 3.7860000000E-01 | +/- 9.5072237807E-03     |
| GoalAchieved | 16.0 | 4.3560000000E-01 | +/- 9.7188577416E-03     |
| GoalAchieved | 18.0 | 4.8330000000E-01 | +/- 9.7950219940E-03     |
| GoalAchieved | 20.0 | 5.3120000000E-01 | +/- 9.7813910489E-03     |
| GoalAchieved | 22.0 | 5.7350000000E-01 | +/- 9.6940223358E-03     |
| GoalAchieved | 24.0 | 6.1140000000E-01 | +/- 9.5541466358E-03     |

```

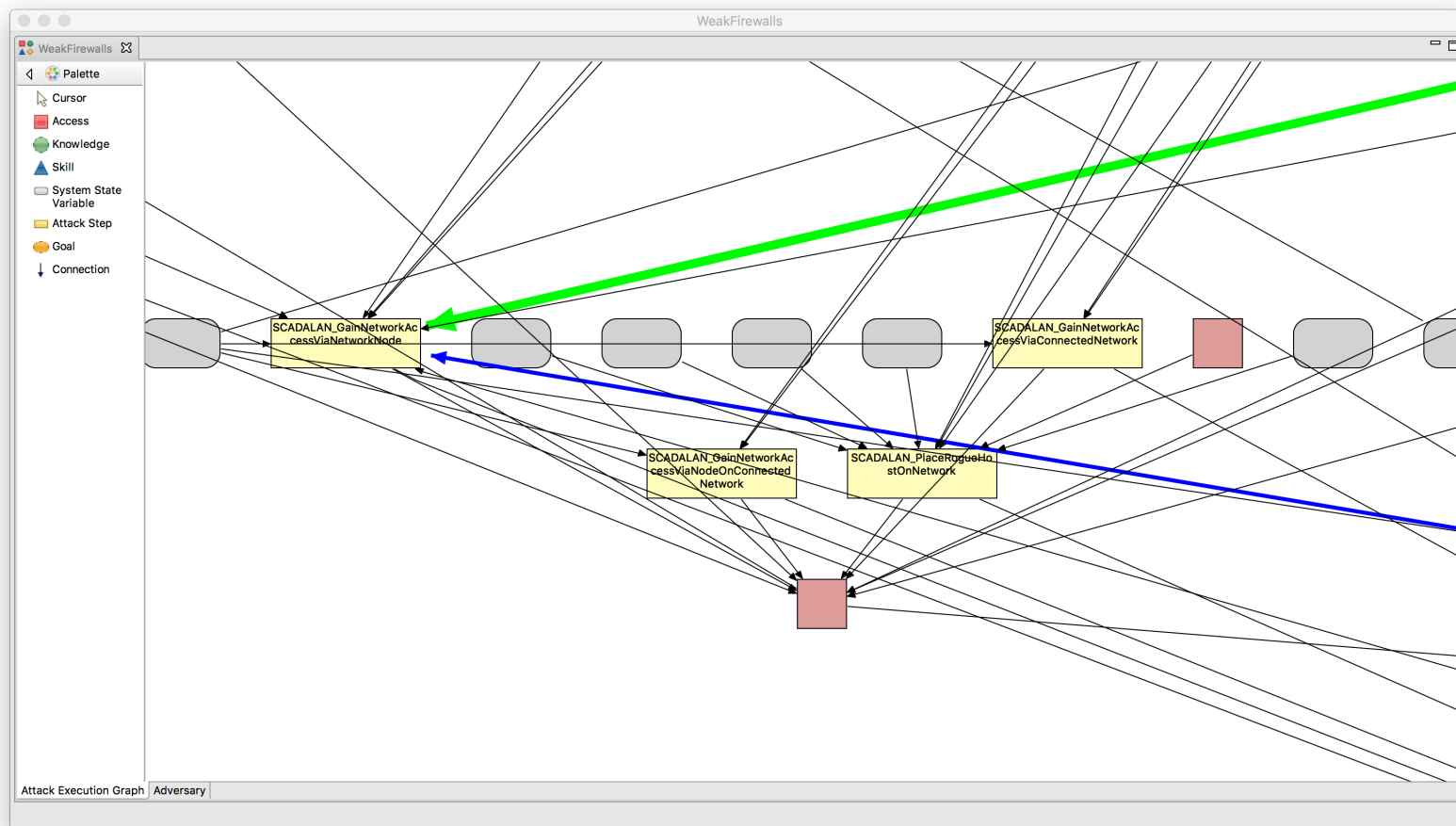
batches Completed:      1000
Experiment Name:        Experiment 4
Global Variable Settings:
  Short Character      userAuthStrength      8
                       userAuthType        'T'
  
```

| Mean Results |      |                  |                      |
|--------------|------|------------------|----------------------|
| Name         | Time | Mean             | Confidence Interval  |
| GoalAchieved | 0.0  | 0.0000000000E00  | +/- 0.0000000000E00  |
| GoalAchieved | 2.0  | 5.1500000000E-01 | +/- 3.0991872098E-02 |
| GoalAchieved | 4.0  | 9.7000000000E-01 | +/- 1.0578396025E-02 |
| GoalAchieved | 6.0  | 1.0000000000E00  | +/- 0.0000000000E00  |
| GoalAchieved | 8.0  | 1.0000000000E00  | +/- 0.0000000000E00  |
| GoalAchieved | 10.0 | 1.0000000000E00  | +/- 0.0000000000E00  |
| GoalAchieved | 12.0 | 1.0000000000E00  | +/- 0.0000000000E00  |
| GoalAchieved | 14.0 | 1.0000000000E00  | +/- 0.0000000000E00  |
| GoalAchieved | 16.0 | 1.0000000000E00  | +/- 0.0000000000E00  |
| GoalAchieved | 18.0 | 1.0000000000E00  | +/- 0.0000000000E00  |
| GoalAchieved | 20.0 | 1.0000000000E00  | +/- 0.0000000000E00  |
| GoalAchieved | 22.0 | 1.0000000000E00  | +/- 0.0000000000E00  |
| GoalAchieved | 24.0 | 1.0000000000E00  | +/- 0.0000000000E00  |

# Small SCADA Networks – Step 6

## Visual Results

- Adversary chose to directly compromise the HMI, rather than go through the firewalls when the firewalls were hardened.



## Step 6 – Interpret Results – Feedback

- Were you surprised by the results?
- Do you believe unexpected results could be useful?
- What more would you like to know about the model to make design decisions based on what you've learned?
- How could the results presentation be improved?



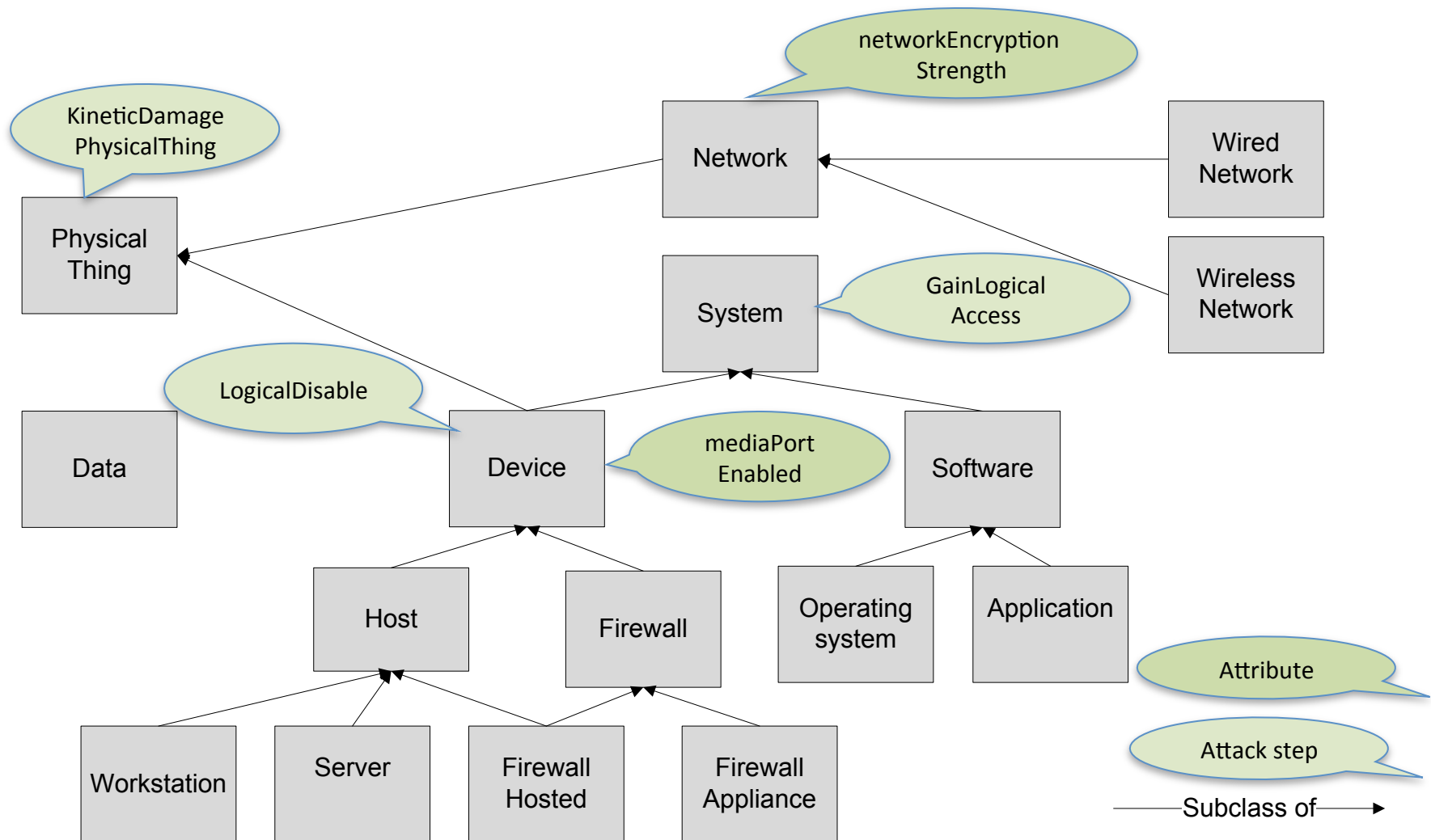
# Agenda

- Registration and Continental Breakfast
- Welcome
- Goals
  - Tool
  - Workshop
- Steps to Use ADVISE Meta
- Hands on Sessions
- Case Studies and Custom Ontologies
- Wrap Up

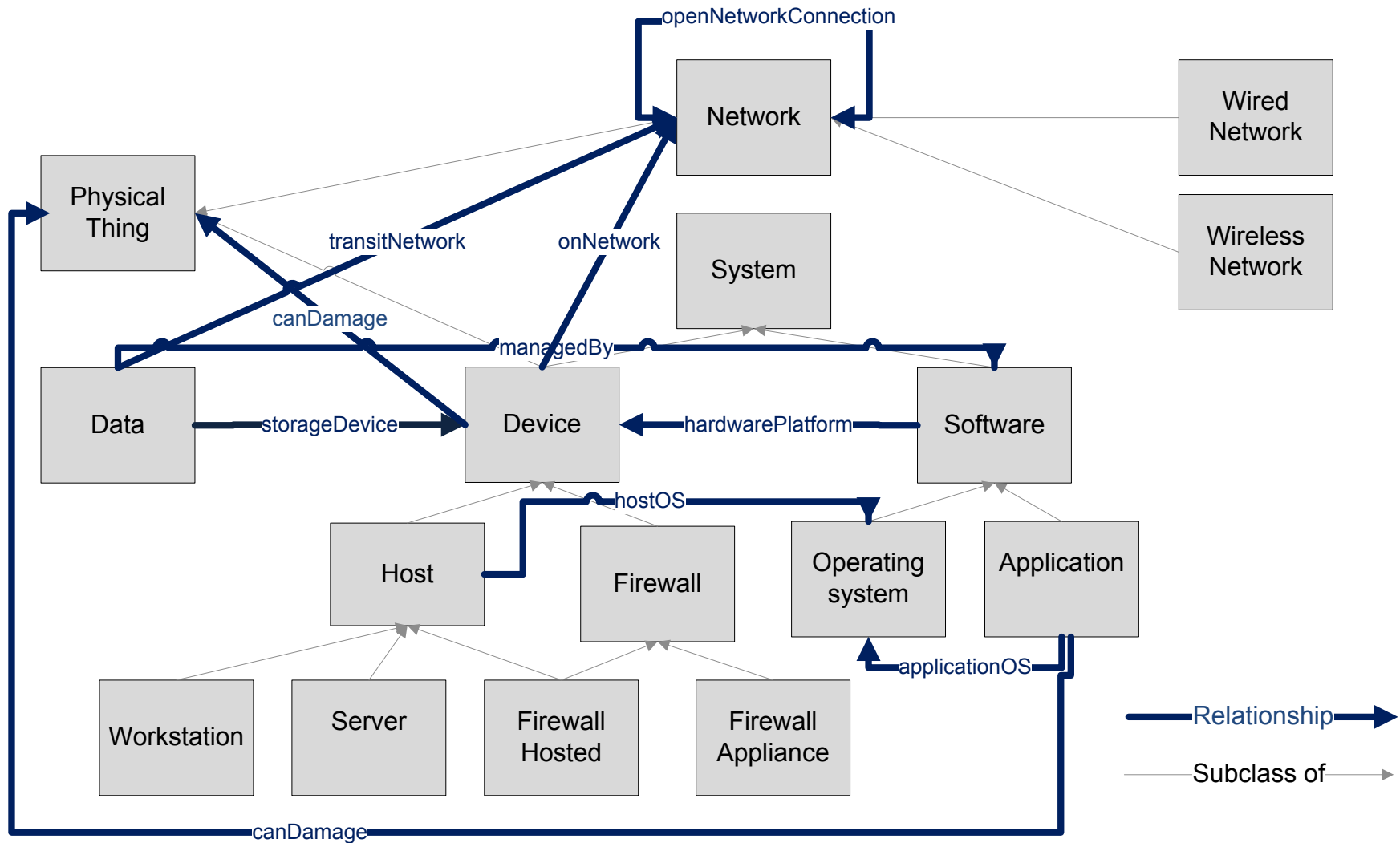
## Base Ontology

- Used for hands-on exercise
- Representative of tool capabilities
- Grounded by:
  - Research on attack methods
  - Study of example analysis previously done with a hand-created AEG
- Not yet a “complete” or vetted dataset

# Base Component Ontology and Inheritance




# Base Ontology Relationships



# Types of Access

- PhysicalAccess(X), where X is a PhysicalThing
  - Not achievable via any attack step, must be given as initial condition
- NetworkAccess(X), where X is a Network
  - Able to read and write bits on the network
- UIAccess(X), where X is a Device or Software
  - Able to touch the login function (if any)
- HasUserCredentials(X), where X is a Device or Software
  - Has the password, token, key, or other credential required to access user functions provided by X
- LogicalAccess(X), where X is a Device or Software
  - Able to access user functions provided by X

# Skills

- See list of skills on ontology tab with symbol 
- Adversary templates define default skill proficiencies
- Most skills are generic
- Reason for adding system specific skills (“Specialized”)
  - Model the tremendous advantage they provide to adversary
- Reason for using broad skill categories
  - Represents how real adversaries accumulate skills
  - Fine grained skill proficiencies (e.g. at stealing passwords or breaking VPNs) unlikely to be known or even guessable in an actual case
  - Haven’t seen reason yet for increasing data input requirements and complexity in attack step models

# What does skill proficiency mean?

|                     |  |      |  |
|---------------------|--|------|--|
| Basic cyber offense | This is a set of skills not further distinguished, some level of which are available via relatively inexpensive tools to any adversary. These include the following elements from the Ethical Hacking Certification Syllabus at <a href="https://www.eccouncil.org/Certification/professional-series/ceh-course-outline">https://www.eccouncil.org/Certification/professional-series/ceh-course-outline</a> : scanning, enumeration, phishing attack, password cracking based on external information (guessing, replay), privilege escalation, hijacking web servers, hacking web applications, SQL injection, buffer overflow, straightforward DoS attacks, network sniffing, social engineering without human contact (for which phishing is an example). | 1000 | Lead individual employed by a nation state to conduct cyber attacks                                |
|                     |  | 800  | Individual with broad skills including stealth, which would earn admirers in the hacking community |
|                     |  | 600  | Individual with solid skills that could be employed to perform ethical hacking engagements         |
|                     |  | 400  | Individual with solid skills in many areas listed, but weak in a few                               |
|                     |  | 50   | Individual can perform simple script-based attacks   |

In the base ontology, probability of success/failure of an attack step often has a linear relationship with one or two skills

- e.g. a generic skill OR a specialized skill impacts outcome

# Base Ontology Attack Steps

## *Damage or disable*

KineticDamagePhysicalThing  
 LogicalDamagePhysicalThing  
 PhysicalDisable  
 LogicalDisable

## *Malware*

CreateTrustedSiteCauseMalwareInstall  
 CreateUnTrustedSiteCauseMalwareInstall  
 CreateRemovableMediaCauseMalwareInstall\*  
 StagePackageCauseMalwareInstall\*  
 InstallMalwareFromFixedMedia\*  
 InstallMalwareFromRemovableMedia\*

\*Not in alpha

## *Gain access*

GainLogicalAccess  
 GainUserCredentials  
 GainLocalUIAccessDevice  
 GainLocalUIAccessOS  
 GainRemoteUIAccessDev  
 GainRemoteUIAccessOS  
 GainNetworkAccessViaNetworkNode  
 GainNetworkAccessViaNodeOnConnectedNetwork  
 GainNetworkAccessViaConnectedNetwork  
 AdminModifyFWOpen  
 CircumventFWRules  
 PlaceRogueHostOnNetwork

## *Compromise data integrity*

ModifyDataLocally

## *Compromise data confidentiality*

ReadManagedDataLocally  
 NetworkEavesdrop



## Attack Steps: Example Probability Calculation

Coded in the Ontology tab->*NetworkEavesdrop* ->Failure:

Probability of the adversary outcome Failure for *NetworkEavesdrop* on a Data element is the probability that the adversary can't break the crypto (if any) on a *network which the data transits*, or that they can break it, but are kicked off the network before harm is done.

(*transitNetwork*, *relationship of Data to Network*)

The probability they can't break the crypto increases with:

- Application layer encryption strength (*attribute of data*)
- Network layer encryption strength (*attribute of network*)

and decreases with:

- Cryptanalysis skill proficiency of the adversary (*adversary parameter*)

The probability that they are kicked off the network increases with:

- the strength of countermeasures on the network to detect and respond to eavesdropping (*attribute of network*)
- attribute defaulted to zero because this is extremely difficult to do

# Features for Future Base Ontology

- Component ontology
  - Types of networks (LAN, WAN, VLAN)
  - VPN connections
  - Routers
  - Gateways
  - Device authentication
  - User roles

## Features for Future Base Ontology (cont.)

- Attack steps
  - Disable or Damage
    - PhysicalDisconnect
    - NetworkFlood
  - Malware
    - CreateRemovableMediaCauseMalwareInstall
    - Stage PackageCauseMalwareInstall
    - InstallMalwareFromRemovableMedia
    - InstallMalwareFromFixedMedia
  - Data Confidentiality
    - Exfiltrate data
  - Network Infrastructure
    - Router and switch attacks
  - 0 - days

# Custom Ontologies

- The base ontology is data, it is not baked into the tool
- A “library designer” may on the ontology tab:
  - Add to or modify base ontology
  - Define a new ontology
- This includes all ontology elements including:
  - Components
    - Relationships
    - Attributes
  - Attack steps
  - Adversaries

# Examples for Custom Ontologies

- Add component types with unique defaults, attributes, and/or attack steps
  - Virtual OS – build model of data center
  - ATM machine – build model of banking organization
  - Smartmeter – build model of planned smart grid architecture
- Add a customized adversary type (e.g. contractor with specific skills)
- Modify formulas used to calculate attack step characteristics
  - Probability of success/failure of attack step
  - Detectability of an attack step outcome
  - Cost of attack step
  - Time to execute attack step
- Build ontology to model internal architecture of a modern electric vehicle together with associated charging stations
- See tutorial to try out creating an ontology  
[https://www.mobius.illinois.edu/wiki/index.php/ADVISE\\_Meta\\_Two\\_Nets\\_Tutorial](https://www.mobius.illinois.edu/wiki/index.php/ADVISE_Meta_Two_Nets_Tutorial)

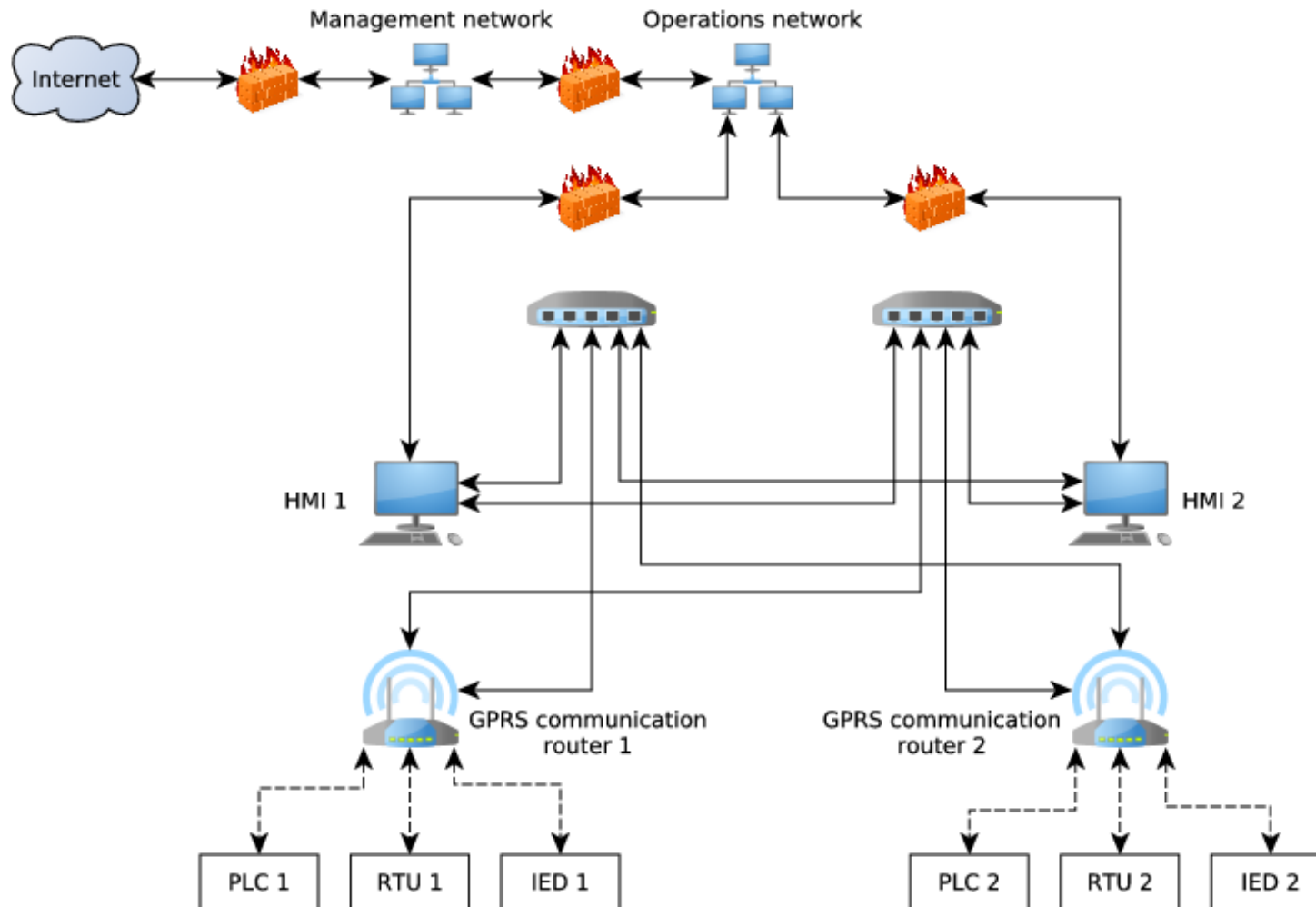
# River Zonal Dispatcher Case Study

## Purpose of analysis

- Investigate the effects on system security of architectural changes to a river zonal dispatcher system with multiple SCADA subsystems.
- In particular, analyze the security impact of intrusion detection systems (IDSes) and isolation, as well as multiple subsystems.
  - How does the behavior of an attacker change when adding IDSes or isolating SCADA subsystems?
  - What key factors would motivate an attacker to choose one SCADA subsystem over another?

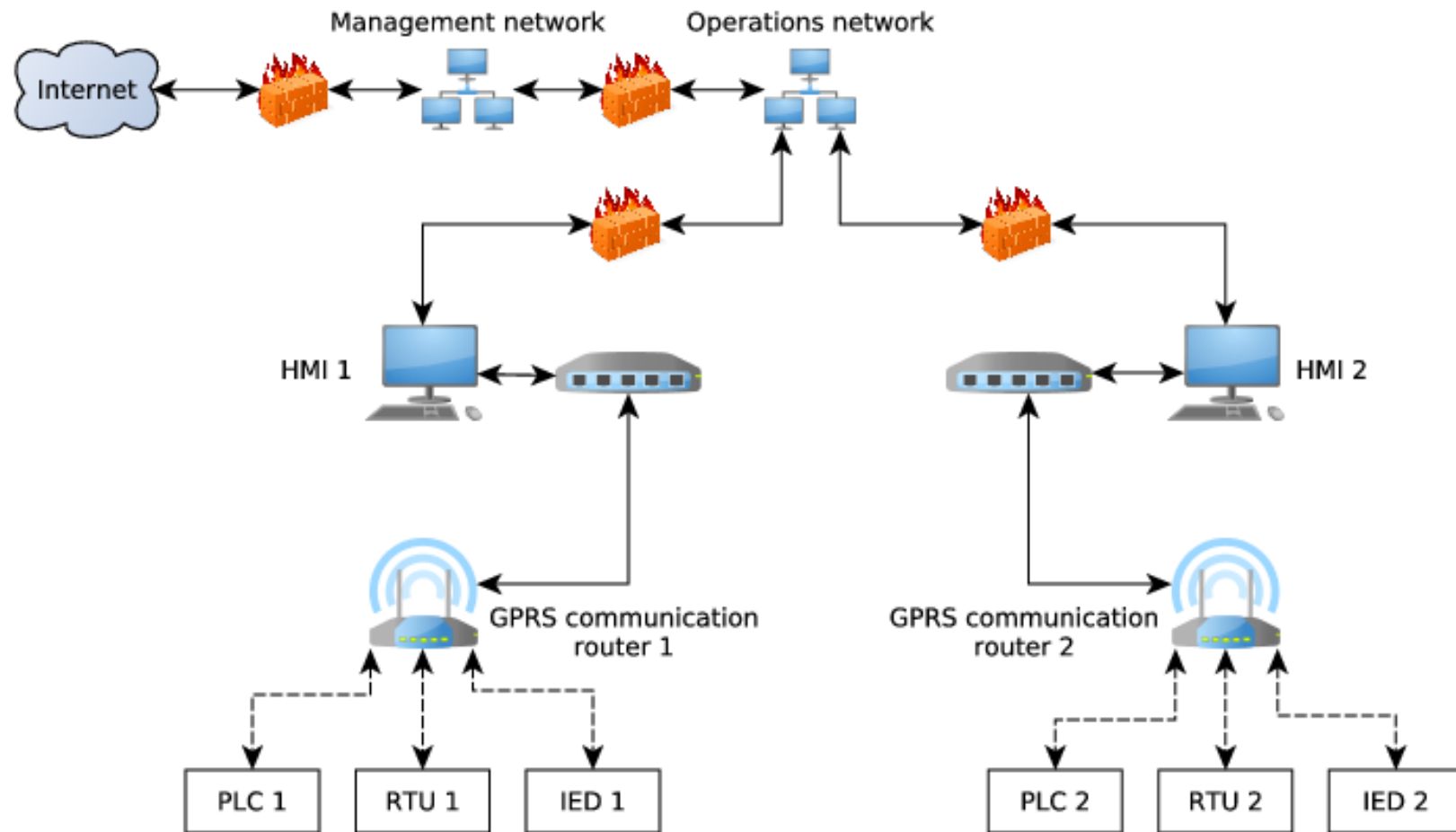
# River Zonal Dispatcher Case Study

## Without Isolation



# River Zonal Dispatcher Case Study

## With Isolation





# River Zonal Dispatcher Case Study

## Model Systems

- Two systems: with and without isolation
- Highlights of models:
  - Four regular networks
  - Devices on two Modbus networks
  - An OPC daemon on an HMI sends commands to the devices controlling the dams
  - Used global variables to control whether IDses and isolation exist

# River Zonal Dispatcher Case Study

## Model Attack Goals and Adversaries

- Install malware on HMI, compromise system via router, compromise system via devices
- Five adversaries are:
  - Foreign government
    - Primarily concerned with installing backdoors on the HMIs and cares little about costs.
  - Hacker
    - Interested in most of the possible goals and is highly skilled, but must consider a balance of concern regarding cost, payoff, and detection.
  - Hostile Organization
    - Highly skilled, but is interested only in compromising the supervisory LANs and is mostly seeking best payoff.
  - Insider Engineer
    - Interested in all goals, but is poorly skilled in attacks.
  - Insider Operator
    - Has access to many parts of the system already, is highly skilled, and is primarily concerned with reprogramming the devices.

# River Zonal Dispatcher Case Study

## Select Metrics (**all standard available metrics**)

- **Average Number of Attempts**
  - Report for each attack step (**maybe not all**)
  - Gives insight on preferred attack path of adversary
- **Probability of Attack Goal Achieved at End Time**
  - Report for each attack goal
  - Gives insight on what goals the adversary is actively pursuing and reaching
- **Average Time-To-Achieve-Goal**
  - For attack goals where the above probability metric is 1 (or close to 1)
  - Gives insight on the speed of the adversary's attack

# River Zonal Dispatcher Case Study

## Generate and Execute Models

- Set up 20 configurations for execution
  - Each of 5 adversaries X 4 system models
  - Calculate all metrics
- Simulation run time set to 8760 seconds
- Ran 1,000 to 10,000 iterations
  - Confidence interval set to 90%
  - Results representing rare events (typically values close to zero) failed to converge

# River Zonal Dispatcher Case Study

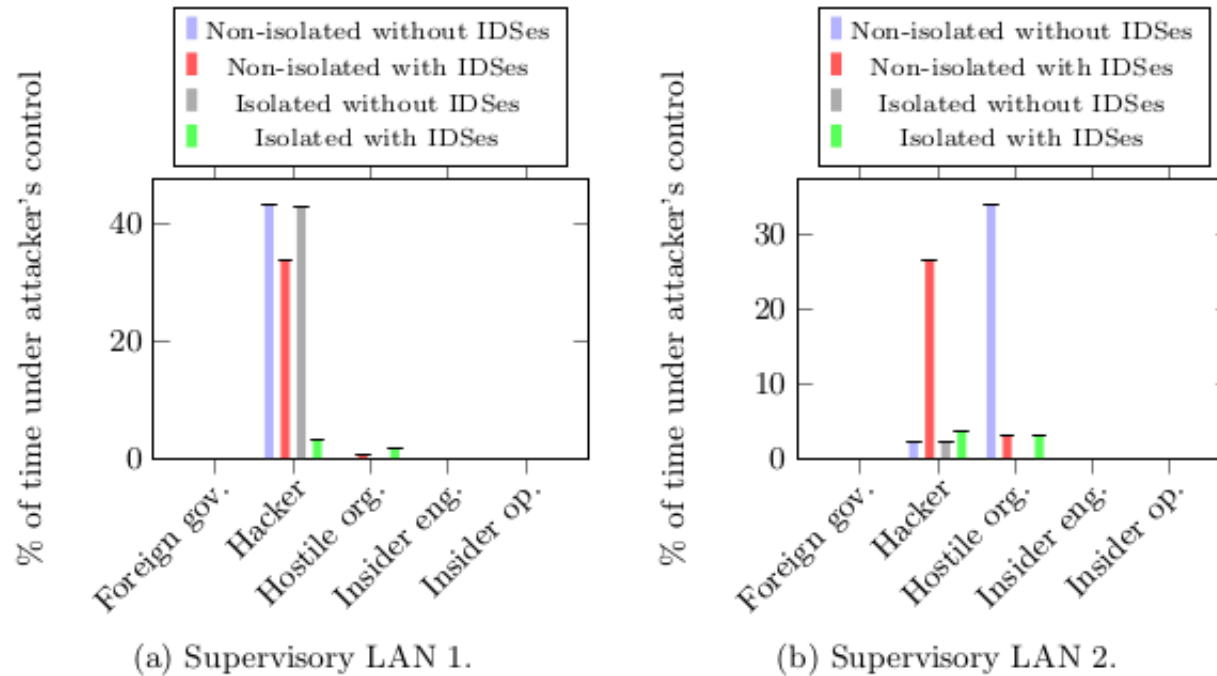


Fig. 6: Average percentages of time in which the attacker has control of a GPRS communication router on a particular supervisory network

# River Zonal Dispatcher Case Study

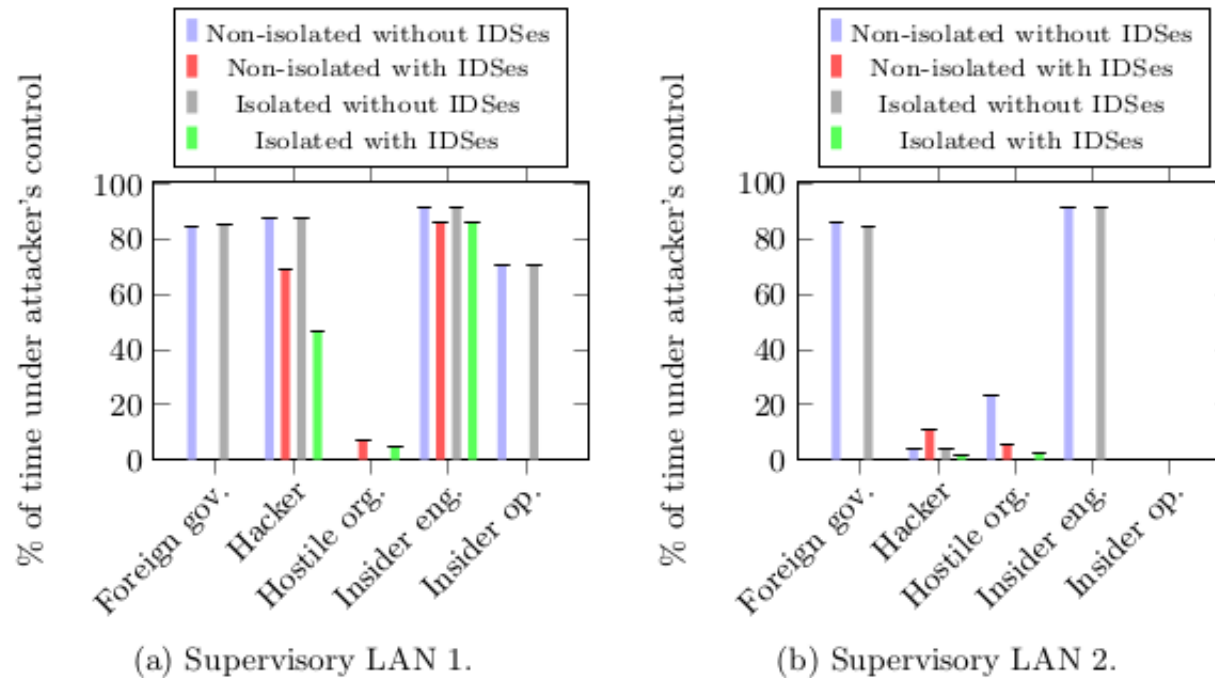


Fig. 7: Average percentages of time that an HMI on a particular network has backdoor software installed.

# River Zonal Dispatcher Case Study

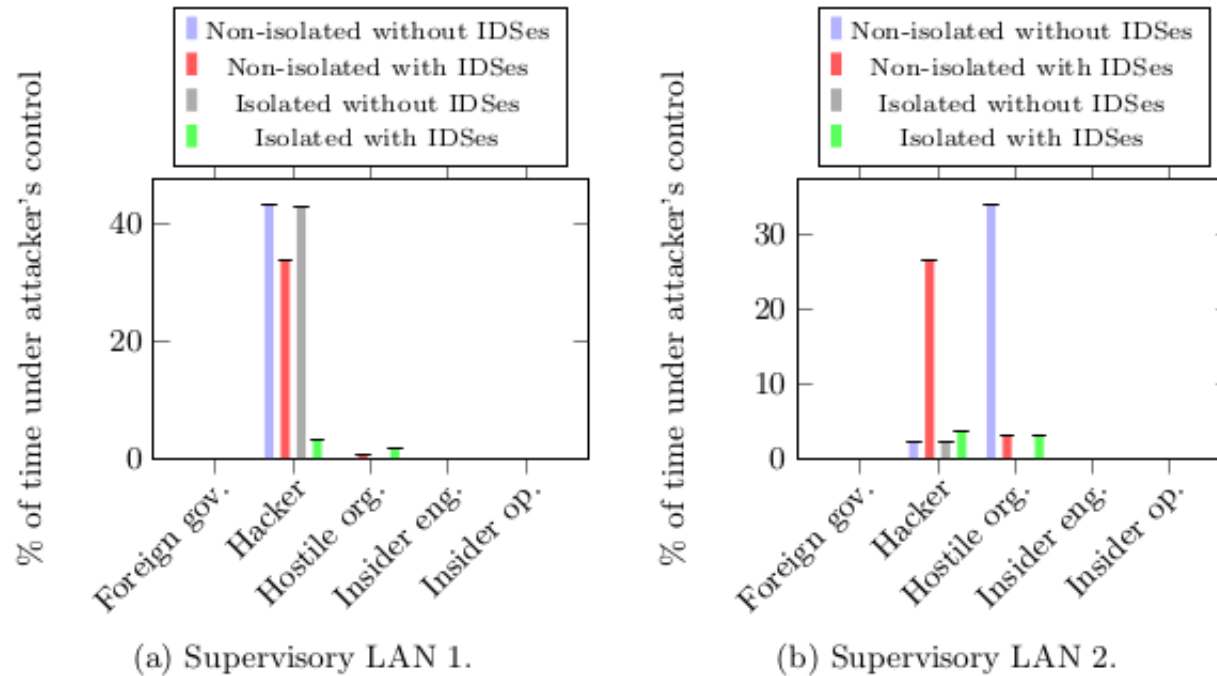


Fig. 6: Average percentages of time in which the attacker has control of a GPRS communication router on a particular supervisory network

# River Zonal Dispatcher Case Study

## Interpret Results

- How does the behavior of an attacker change when adding IDSes or isolating SCADA subsystems?
  - Attacker behavior changes as outcome probabilities and global variables change, affecting the preconditions and attractiveness of attack steps
- What key factors would motivate an attacker to choose one SCADA subsystem over another?
  - Payoff differences (seen by duration of mean time that backdoor SW is installed on HMI)



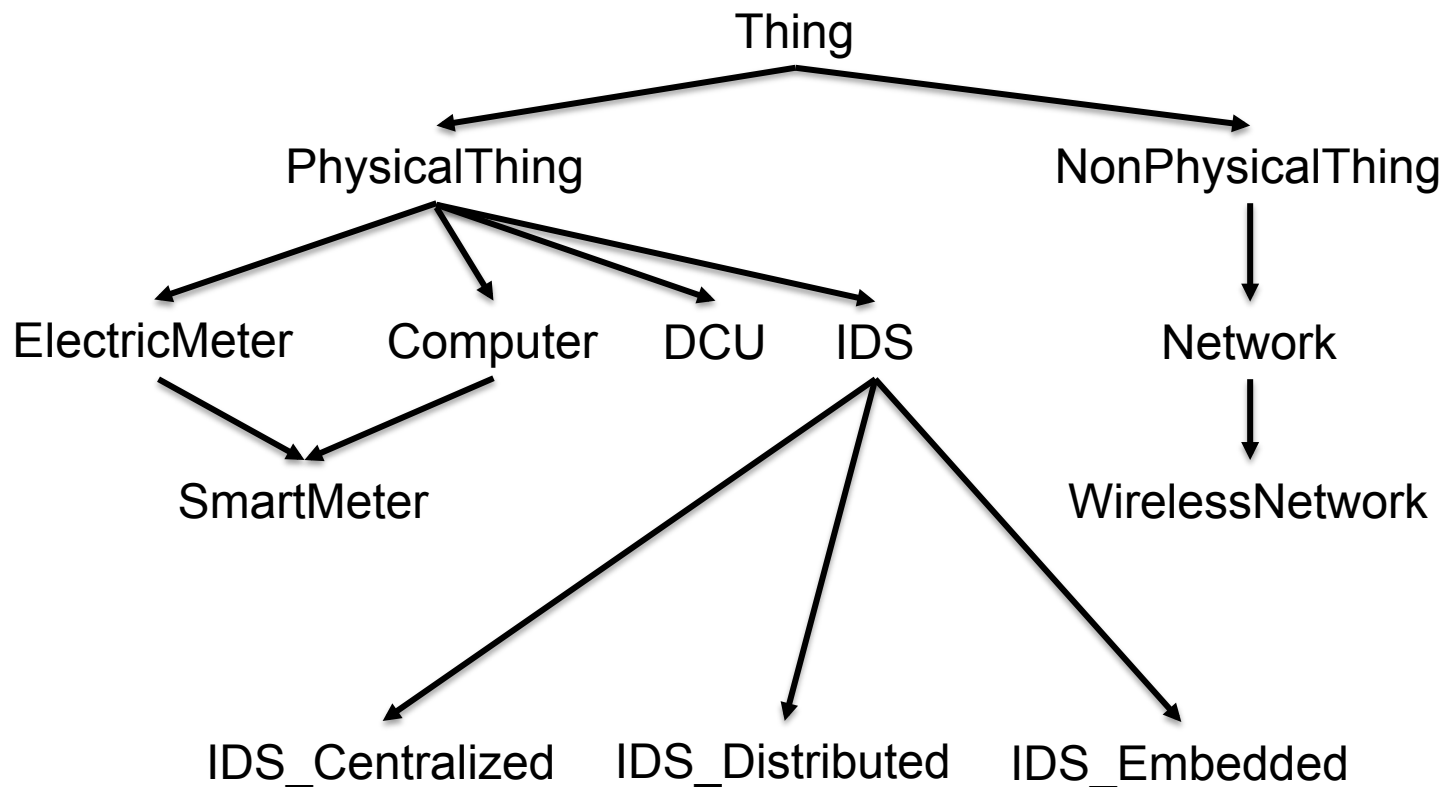
# Advanced Metering Infrastructure Case Study

## Define purpose of analysis

- Determine the cost-effectiveness of different intrusion detection systems (IDSes) in an Advanced Metering Infrastructure (AMI) network.
- In particular, compare
  - Centralized IDS,
  - Distributed IDS, and
  - Embedded IDS.

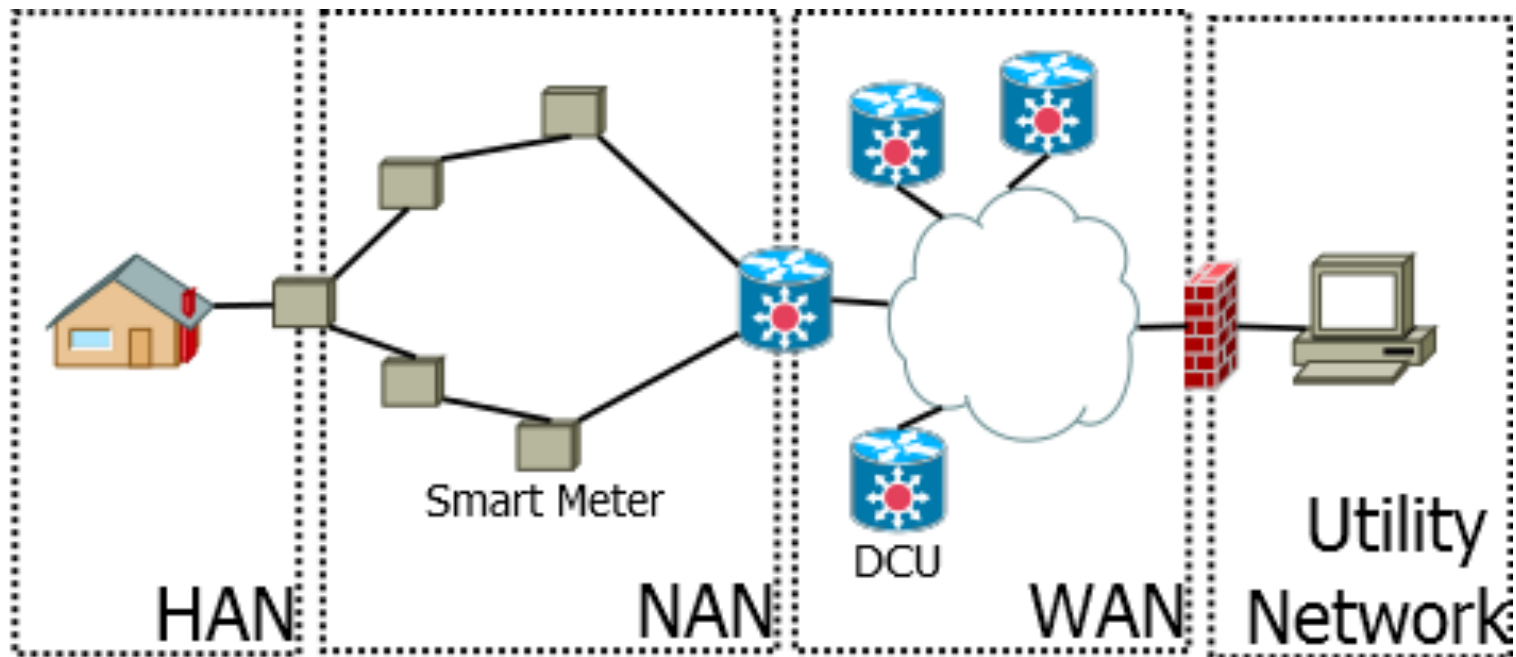
# Advanced Metering Infrastructure Case Study

## Define system components, relationship and attributes



# Advanced Metering Infrastructure Case Study

## Define system components, relationship and attributes



# Advanced Metering Infrastructure Case Study

## Model Attack Goals and Adversaries

- Compromise availability and integrity of the AMI, or steal electricity, depending on adversary.
- Adversaries include
  - Malicious customers
  - Insider
  - Nation-State
  - Terrorist

# Advanced Metering Infrastructure Case Study

## Select Metrics

- Damage to System
- Probability of Detecting Adversary
- Attack Path

# Advanced Metering Infrastructure Case Study

## Simulate Model

- Execute ADVISE models to determine how each of the three IDSes may fare when faced with attacks from the four adversary types, when compared on
  - Likelihood of detection,
  - Attack path chosen by adversary, and
  - Damage to the system.
- Calculate cost-effectiveness of each IDS

# Agenda

- Registration and Continental Breakfast
- Welcome
- Goals
  - Tool
  - Workshop
- Steps to Use ADVISE Meta
- Hands on Sessions
- Case Studies and Custom Ontologies
- Wrap Up

## Wrap Up

- General feedback
  - Was this tool useful?
  - How could you and your organization use it?
  - What areas need work?
- The Near Future
  - Improvements to ADVISE/Actor Model
  - Expanded Ontology
  - System to Easily Share Ontology Packages
  - Ontology Editor Improvements / Validation



# Thank You!

- Contact Info
  - [staff@mobius.illinois.edu](mailto:staff@mobius.illinois.edu)
  - <http://www.mobius.illinois.edu>