

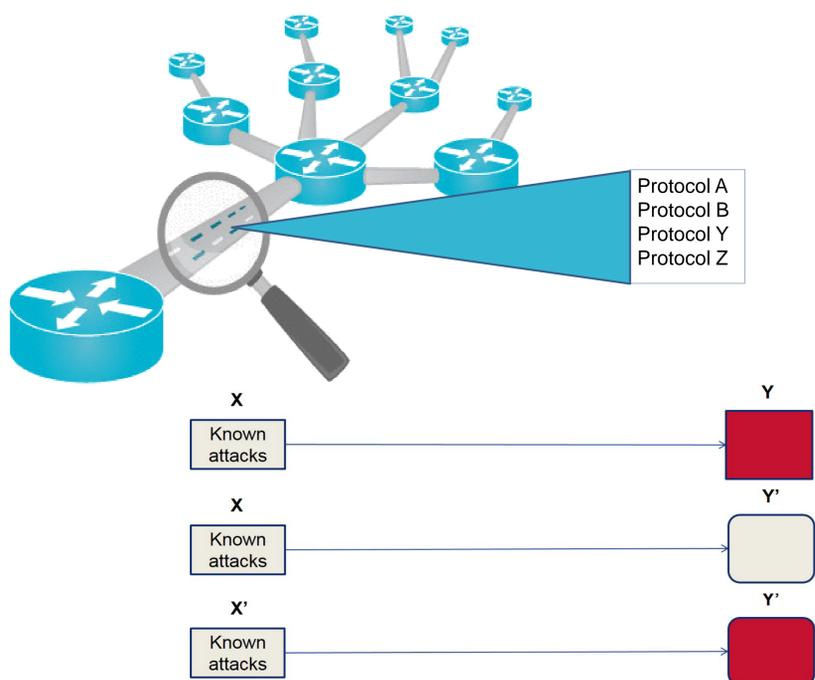
**S&T Campaign:**  
**Analysis & Assessment**  
*Assessing Mission Capability of Systems*

**Jaime C. Acosta, Ph.D**  
 (575) 678-8115  
 jaime.c.acosta.civ@mail.mil

**Caesar Zapata**  
 (575) 678-5764  
 caesar.zapata.civ@mail.mil

## Objectives

- Develop a tool that will automate the generation of network protocol models that can be used to support the analysis of systems that use these protocols.
- Enable further analysis of network protocols within stringent time constraints by automatically generating client software capable of communicating with them.



Analysts target protocols of interest and evaluate their survivability against a set,  $x$ , of known attacks. Variations in the effectiveness of such attacks have been observed across assessments. Often, these can be attributed to modifications within the target protocols.

## Challenges

- Determining message types and packet format.
- Extracting protocol behavior.
- Automating the process of translating protocol behavior to model and simulation code.

## Approach

- Leverage a dictionary of known message types and Wireshark's dissector library.
- Utilize Prospex's (Protocol Specification Extraction tool) inference module to extract network protocol state machine.
- Generate ns-3 protocol models and simulations based on network captures.
- Develop a pluggable software framework that enables analysts to generate code for multiple platforms.

## Progress

- Extracted protocol specific data (e.g., fields, field properties, message type, packet format, etc.) and the state machine for the ICMP protocol successfully from a network traffic capture.
- Leveraged the extracted protocol behavior to automatically generate an ns3 (Network Simulator Engine) model.
- Automated the process of creating an ns-3 simulation that utilizes the generated model.

This complex block illustrates the workflow from traffic capture to simulation. On the left, a terminal window shows the output of a protocol dissector (likely Wireshark) for an ICMP Echo (ping) reply. The output lists fields such as Type, Code, Checksum, Bad Checksum, and Identifier. In the center, a table shows the extracted 'Vocab' for these fields: Type (0), Code (0), Checksum (6fe9, f4e2, b3da, 75d2, 73cb), and Identifier (0). On the right, a state machine diagram shows states [0, 1] and transitions labeled with '0' and '1'. Below this, another terminal window shows the generated state machine code in a format like 'digraph message{ node0 [label="0", shape=circle]; node29 [label="0 8", shape=circle]; node0 -> node29 [label="8"]; node29 -> node0 [label="0"]; }'. The file is named 'labeledStateMachine.dot'.

Vocabulary extraction and state machine generation.

## Limitations

- Manual intervention is still required when selecting the vocabulary to use for fields exhibiting high entropy.
- Input files must be filtered to contain only one communication stream.
- Fixed length fields are assumed.

## Sought Collaboration

- Develop a flexible architecture that allows cross platform inter and intra packet algorithm development e.g., checksum and sequence numbers
- Test the model's ability to establish a connection and communicate with real network protocols.