GOOSE Code Documentation

Introduction

The Generic Object Oriented Substation Event (GOOSE) protocol is part of the IEC 61850 standard, designed for real-time communication between Intelligent Electronic Devices (IEDs) in electrical substations. It enables high-speed, event-driven messaging for critical protection and control functions. This documentation outlines the usage of the GOOSE protocol code you have developed, including sending, receiving, and parsing GOOSE messages, as well as performing packet modifications.

System Requirements

To run this code, the following system setup is required:

- **Python version**: 3.x
- **Required libraries**:
    - `pyasn1`: For encoding and decoding ASN.1-based GOOSE messages.
    - `Scapy`: For network packet crafting and capture.
    - **Optional**: `Wireshark` for capturing and analyzing GOOSE packets.

Installation of Libraries

Install the required Python libraries using the following command:

```
pip install pyasn1 scapy
```

Additionally, ensure that the `goose` library is located in the parent folder of your project directory, as this is required for the code to function properly.

Usage Instructions

The GOOSE protocol code supports both **live capture** from network interfaces and **offline processing** of `.pcap` files. The usage of these two modes is explained below:

1. Live Capture

For capturing and processing GOOSE messages from live network traffic, use the following command:

```
python3 script_name.py --livecapture --output <output_filenam
e>
```

This command listens to network traffic in real-time, capturing GOOSE messages and saving the output to the specified file.

2. Offline Processing of PCAP Files

For processing previously captured packets from a PCAP file, use the following command:

```
python3 script_name.py --pcapfile <input_filename.pcap> --out
put <output_filename>
```

This processes the GOOSE messages from the PCAP file and writes the results to the specified output file.

> In both commands, replace script_name.py with the name of your Python script.

Manual Code Modifications

The GOOSE protocol code has been manually modified to enhance the handling of packet manipulation and message processing. These modifications are crucial for controlling specific GOOSE message fields and applying custom logic.

Modification Logic

- The **manual changes** are implemented between lines **270 and 333** of the code.

- These lines contain logic for **encoding, manipulating, and modifying** the fields of GOOSE messages, such as:
  - **Encoding of GOOSE fields** using ASN.1 encoding methods.
  - **Dynamic packet modification** using Scapy to allow real-time updates to fields like `stNum`, `datSet`, and `goID`.
  - Efficient handling of **packet injection** back into the network for testing or simulation purposes.

Ensure that any further adjustments are done within this section to avoid disrupting the core logic of the message processing pipeline.

Key Changes in the Logic:

- **Lines 270-290**: Responsible for encoding and preparing the GOOSE message components using ASN.1 encoding.
- **Lines 291-320**: Leverages the Scapy library to inject or capture GOOSE packets from the network.
- **Lines 321-333**: Implements packet modification features, allowing fields like `timeAllowedtoLive` and `stNum` to be dynamically changed during runtime.

Additional Notes

- **Wireshark Integration**: If needed, you can capture and analyze GOOSE messages using Wireshark. GOOSE packets can be filtered using the EtherType `0x88b8`.
- **Security Considerations**: Be cautious when using this code in a live substation environment, as GOOSE messages directly control real-time functions in electrical substations. Unauthorized manipulation of GOOSE messages can lead to serious operational consequences.
