TrustBreak: Detecting & Ejecting Malicious PEs in secure VPLS networks

Introduction

The demand for safe and efficient communication across networks is becoming crucial in today's digital era. Traditional Internet Protocol (IP) addresses serve two primary functions: identifying devices on a network and determining their location. However, this dual role can create security challenges and limit network management flexibility. The Host Identity Protocol (HIP) is utilized in this context. HIP separates these two roles by introducing a new identification layer, allowing devices to be recognized by cryptographic identities, making it harder for unauthorized users to gain access.

Consider a large organization that maintains office locations in multiple cities. To ensure all their offices can share information as if they were on the same local network, they can use Virtual Private LAN Service (VPLS) technology. VPLS, as shown in Figure 1, creates a virtual network that connects different locations seamlessly over the Internet, as if cables physically connect them. The combination of HIP and VPLS, known as Host Identity Protocol-based Virtual Private LAN Service (HIPLS), comes into play to make the VPLS network secure. HIPLS ensures that only authorized devices can join the network, and it allows these devices to communicate securely across different locations[1].

In operational settings, such overlays depend on group keys shared among provider-edge (PEs) nodes. A realistic threat model includes misconfigured or malicious participants that inject invalid contributions or attempt traffic manipulation. HIP with a multi-party exchange (HIP-MBX) supports verifiable group key establishment: each participant’s contribution can be checked, enabling precise identification and exclusion of a faulty node without reinitializing the entire group.

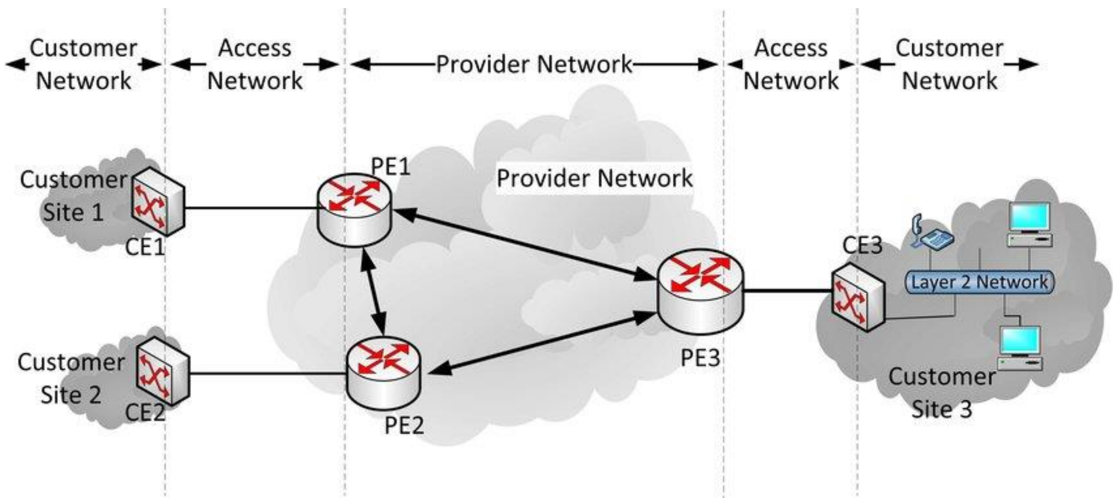
Project Description:

This project will consider and evaluate malicious-participant scenarios within a HIPLS testbed, detect faults via HIP-MBX verification, trigger limited rekeying, and quantify dynamics such as detection latency, rekey time, and transient data-plane impact. The outcome is an empirical assessment of fast fault isolation mechanisms that sustain

[1] For real-world scenarios, visit https://www.tempered.io/

availability and security in settings where reliable, protected communication is critical (e.g., healthcare and finance).

Figure 1: VPLS Network



Project Objectives:

1. **Simulate and detect a malicious node in HIPLS**
    - Extend the existing HIP-MBX protocol to simulate malicious behavior during the group key agreement phase, such as injecting invalid cryptographic contributions or tampering with protocol messages.
    - Develop detection mechanisms that allow honest nodes to verify received contributions and accurately identify misbehaving participants within the group.
    - Enhance the protocol to detect and respond to malicious nodes attempting to disrupt the key agreement process and simulate network behavior under such adversarial conditions.
2. **Implement Fast Fault Isolation and Rekeying:**
    - Design and integrate a recovery mechanism that isolates malicious nodes without restarting the entire key exchange process. Ensure that rekeying is localized, minimally disruptive to the network, and capable of maintaining secure communication among the remaining trusted participants.
    - Simulate large-scale network conditions with varying numbers of participants (e.g., 20–100) and measure key metrics such as detection time, rekey convergence time, and packet loss during rekeying.

Project Deliverables:

1. **Source Code:**
   - Implementation of detection, fault isolation, and partial rekeying logic integrated into the HIPLS testbed.
2. **Technical Documentation:**
   - User manual for deploying and testing the system in different network scenarios.
   - Developer documentation outlining the integration of detection mechanisms within the HIP-MBX protocol and instructions for modifying node behavior.
3. **Simulation Results and Analysis:**
   - Comprehensive report detailing detection accuracy, rekey performance, and impact on network stability.
   - Evaluation of system behavior under both normal and adversarial conditions, including discussion of trade-offs, scalability, and fault-recovery efficiency.

Requirements:

- Completing the Computer Networks and Distributed Systems course (TDTS04-TDTS06-TDTS11 or any related course) is essential.
- **Advanced** Python programming skills (HIPLS and mininet were written purely in Python).
- Solid understanding of network security fundamentals, including cryptographic algorithms and protocols such as Diffie-Hellman (DH), RSA, AES, HMAC, and IPsec.
- Competence in socket programming and working with low-level network protocols for implementing and testing communication between hosts.
- The existing codebase consists of approximately **10,000–15,000 lines of code**, including HIP integration, virtual LAN setup, and multi-party key exchange logic. This classifies the project as **medium-scale**, requiring structured understanding and modular contributions rather than full system development from scratch.

Readings:

- Host Identity Protocol (HIP): Towards the Secure Mobile Internet: https://www.wiley.com/en-us/Host+Identity+Protocol+(HIP)%3A+Towards+the+Secure+Mobile+Internet-p-9780470772904
- Primer on Host Identity Protocol: https://www.ida.liu.se/~TDDE21/info/primer-host-identity-protocol-whitepaper.pdf
- https://www.sciencedirect.com/science/article/pii/S1389128621002814
- https://github.com/strangebit-io/hip-vpls