Running HIP-VPLS in infrastructure mode

Dmitriy Kuptsov
2024

Table of contents

CHAPTER 1

Introduction

CHAPTER 2

Background

In this section we are going to describe some background material. We are going to start with the cryptographic primitives such as symmetric key encryption/decryption algorithms and then move on to the discussion of the Virtual Private LAN Services and what kind of problems they solve. We will then conclude the discussion in this chapter with the basic information on Host Identity Protocol as it is in the core of the solution which we are discussing in this document. To make the background material more or less complete we are going to touch alternative Transport Layer Security (we will here mention why this protocol is not used in the core of our architecture and is only used for the control-plane communications between the HIP-Switches and the HIP-controller). With these final words we are going to conclude current chapter of this work.

2.1 Cryptography

Cryptography forms the bases for secure telecommunications nowadays. SSL, TLS, SSH, Tacacs+, IPSec, DKIM, DNSSec are only few well-known telecommunication protocols that use cryptography to prevent such well-known attacks as eavesdropping, tampering, denial of message origin, etc. Modern cryptography is based on the hard-core mathematics and non-trivial algorithms (such as random number generation, discrete logarithm problem, rings, fields, Euclidean algorithm, factorization of big numbers, etc.)

2.1.1 Symmetric cryptography

Symmetric key cryptography is just perfect for the data-plane traffic as it offers high-processing times (when compare to asymmetric key cryptography). As the name implies, symmetric key cryptography uses the same secret key to encrypt and decrypt messages. On one hand it is the main reason why these algorithms are so fast. On the other hand, and this is the main limitation of the type of cryptography: symmetric keys are hard to distribute and revoke without using more sophisticated symmetric key schemes.

As of today several symmetric key cryptography algorithms, such as Advanced Encryption Standard (AES), Tripple DES (3DES), and Twofish offer advantageous processing speed and sufficient security levels. In our prototype implementation of HIP-VPLS we are using AES with 256 bits keys to perform encryption and decryption of data-plane traffic. Moreover, since NanoPI R2S - hardware that we employ to run our Software Defined Network (SDN) code - has support for on-chip instructions to boost the encryption and decryption of arbitrary long message blocks. In other words we do perform AES operations directly in the Central Processing Unit (CPU) of the tiny computer. We are going to devote a separate section on the implementation of the hardware accelerated AES encryption and decryption routines by the CPU.

2.1.2 Asymmetric cryptography

Asymmetric key cryptography as the name suggests uses two separate keys to encrypt and decrypt the messages. Since the encryption uses big number exponentiations (such as RSA) and multiplications (such as EDDSA), as well as modular arthihmetics, the performance of these types of algorithms is considerably worse when compared to symmetric cryptography algorithms.

However, since one is allowed to expose public part of the key to anyone, and since this key is only required to encrypt the message and only person who holds the private part of the key (secret part of the key) can decrypt the message, efficient key distribution and revocation can be organized, at the cost of extra CPU cycles. Moreover, by encrypting the message with the private key, and then making decryption plausible only with a public key (exposed to everyone), digital signature schemes can be implemented at no hassle. Many distributed versions of signature/encryption also exist in the literature broadening the application landscape of this type of cryptography. In our work we are using RSA and ECDSA in HIP protocol to generate message signatures. Public keys are also used in HIP protocol to derive permanent identifiers for the HIP-switches.

2.1.3 Hash functions

Hash functions are one way mathematical functions used to generate the so called fingerprints of a message (MACs). In other words, given arbitrary long input message, a fixed size universly unique message (typically, 128, 160, and 256 bits) is produced. Ideally, it should be computationally impossible to reverse the function to find the original message (or image) given the hash (or fingerprint). Hash functions are used in digital signatures to compress the message befor signing it with public key cryptography algorithms. Modern, hash functions should guarantee that know collisions are possible for the hash function (in other words, it should be hard to find two distict images that will both produce the same hash code).

Keyed versions of hash functions are also widely spread. For example, Hash MAC (HMAC) is used in symmetric setting when both parties share the key. This type of algorithm is used for authentication and identification process. HMACs are typically used to protect the data-plane traffic from the forgery attacks.

2.1.4 Key exchange protocols

2.2 Virtual Private LAN Service

2.3 Host Identity Protocol

Internet was designed initially so that the Internet Protocol (IP) address is playing dual role: it is the locator, so that the routers can find the recipient of a message, and it is an identifier, so that the upper layer protocols (such as TCP and UDP) can make bindings (for example, transport layer sockets use IP addresses and ports to make a connections). This becomes a problem when a networked device roams from one network to another, and so the IP address changes, leading to failures in upper layer connections. The other problem is establishment of the authenticated channel between the communicating parties. In practice, when making connections, long term identities of the parties are not verified. Of course, there are solutions such as SSL which can readily solve the problem at hand. However, SSL is suitable only for TCP connections and most of the time practical use cases include only secure web surfing and establishment of VPN tunnels. Host Identity Protocol on the other hand is more flexible: it allows peers to create authenticated secure channels on the network layer, and so all upper layer protocols can benefit from such channels.

HIP relies on the 4-way handshake to establish an authenticated session. During the handshake, the peers authenticate each other using long-term public keys and

derive session keys using Diffie-Hellman or Elliptic Curve (EC) Diffie-Hellman algorithms. To combat the denial-of-service attacks, HIP also introduces computational puzzles.

HIP uses truncated hash of the public key as identifier in a form of IPv6 address and exposes this identifier to the upper layer protocols so that applications can make regular connections (for example, applications can open regular TCP or UDP socket connections). At the same time HIP uses regular IP addresses (both IPv4 and IPv6 are supported) for routing purposes. Thus, when the attachment of a host changes (and so does the IP address used for routing purposes), the identifier, which is exposed to the applications, stays the same. HIP uses a special signaling routine to notify the corresponding peer about the change of locator. More information about HIP can be found in RFC 7401.

CHAPTER 3

Architecture

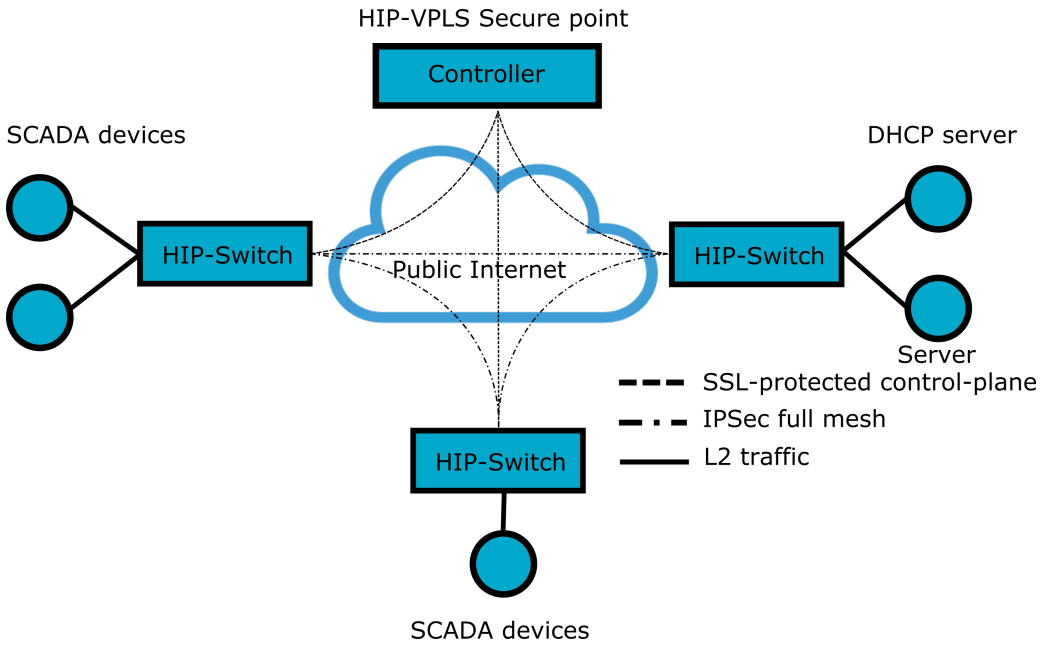

Figure 3.1: System architecture

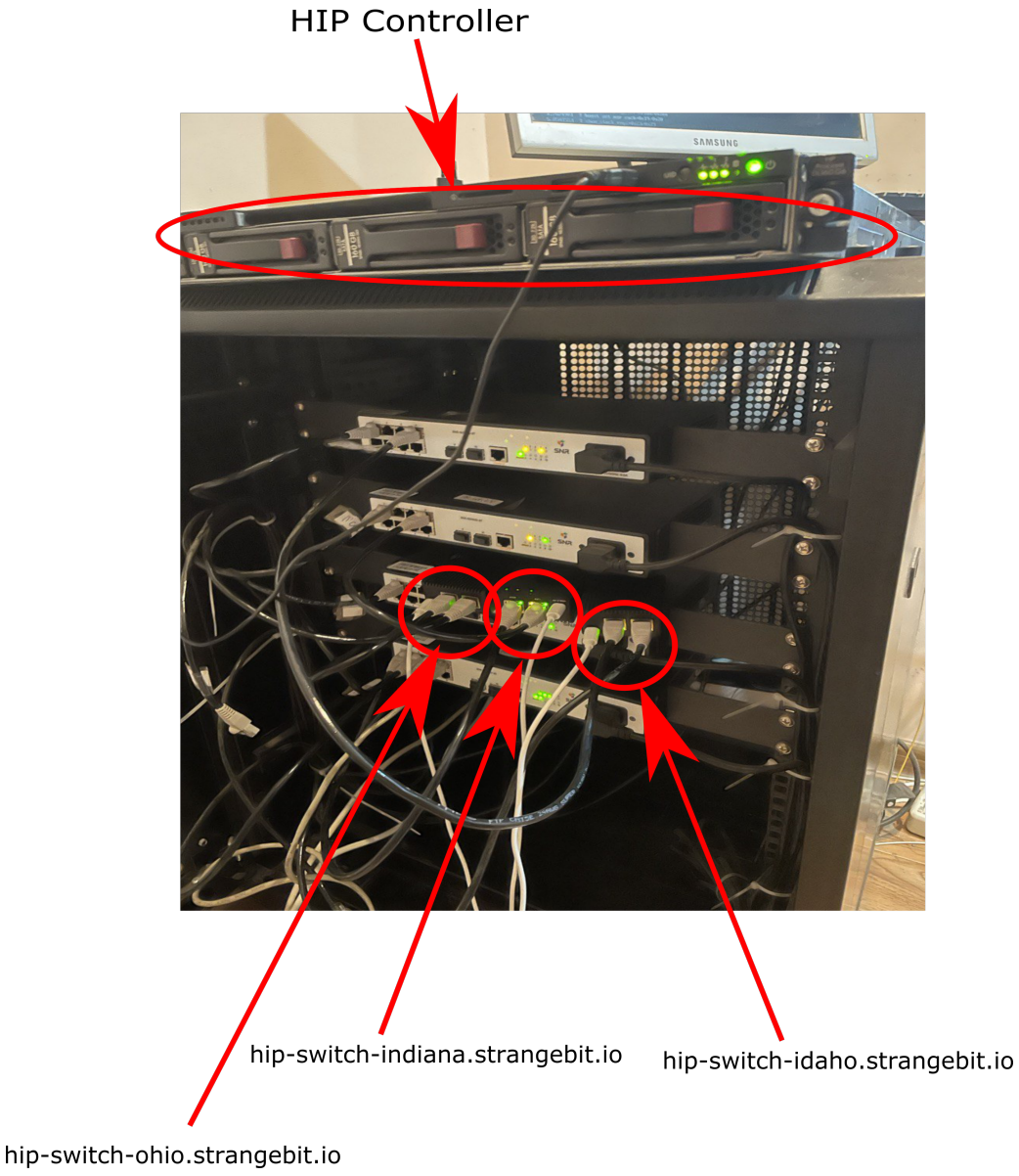

Figure 3.2: Testbed setup

CHAPTER 4

Proof-of-a-concept implementation

In this chapter, we are going to discuss the proof of a concept implementation of the HIP-VPLS (both HIP-switches and controller). In our work we have used the Python language as it offers simplicity at the cost of extra CPU cycles to do the job.

4.1 Accelrating AES with the hardware

4.2 Controller

4.2.1 HIP-switch registration

4.2.2 HIP-switch configration

4.2.3 HIP-switch MAC Access Control List

4.2.4 HIP-switch traffic shaping

Alhtough we did not implement this feature in HIP-switches and HIP controller, we believe that it is still valuable for the future research. For example, different hosts can be served differently (have more bandwidth) than some other hosts by using traffic shaping. For example, if some hosts in HIP-VPLS network send delay sensitive traffic, for example, curtain rules can be configured on HIP controller to

give needed advantage over other hosts in the network. We leave this for the future discussions and work.

CHAPTER 5

Performance evaluation

Although our goal was not to build production grade solution here we are still would like to discuss the performance (such as throughput) of our solution for two different settings: (i) without optimizations (such as hardware AES acceleration) and without code improvements; and (ii) with the improvements. As the metric we are going to use user obtained thoughput with iPerf tool.

Literature