

A Security Architecture for a Satellite Network Transport Architecture[†]

André Zúquete¹, Ana Simões²

¹ IEETA / Instituto de Telecomunicações / Universidade de Aveiro
Campus Universitário de Santiago
3810-193 Aveiro, Portugal
avz@det.ua.pt

² SkySoft Portugal, Av. Conselheiro Fernando de Sousa, n.º 19 - 12º
1070-072 Lisboa, Portugal
ana.simoos@skysoft.pt

Abstract

This paper presents the security architecture designed for SaNTA (Satellite Networks Transport Architecture). SaNTA is an architecture designed for accelerating TCP connections through satellite links. It uses a split architecture to overcome problems in the TCP congestion control mechanism when using satellite links. However, such split architecture cannot easily interoperate with secure communication protocols, that use an end-to-end paradigm. In this paper we present a security architecture for SaNTA using state-of-the-art security solutions: IPSec and SSL/TLS, as well as packet-filtering firewalls and NAT gateway mechanisms. This security architecture allows SaNTA to deal properly with end-to-end secure communication protocols, though not accelerating them, and to properly protect all traffic managed by SaNTA.

1 Introduction

SaNTA (Satellite Networks Transport Architecture) is an architecture initially designed to provide an efficient exploitation of TCP traffic over long-latency satellite links. For this purpose SaNTA breaks the traditional end-to-end paradigm and uses a split architecture where intermediate nodes impersonate end nodes for accelerating TCP acknowledgements (see Figure 1).

Split architectures like SaNTA raise many issues when security is addressed. First, most paradigms used in secure communication protocols, such as the ones implemented above the link layer (e.g., IPSec [11] at the network level and SSL/TLS [1] and SSH [10] at the session/presentation layers) enforce end-to-end security policies, thus preventing split architectures from impersonating end entities. Second, confidentiality enforced by some end-to-end secure communication protocols, namely when applied below the transport level, prevent split architectures from accessing transport headers. Therefore, dealing with security within split architectures is a complex matter that deserves special attention.

[†]Work supported by the European Space Agency (ESA) under Contract No. 15333/01/NL/ND

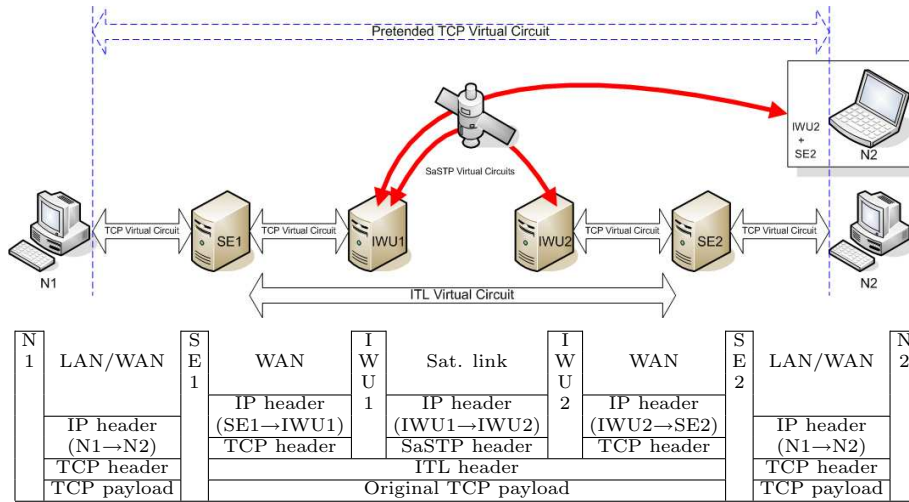


Figure 1: SaNTA TCP accelerating architecture and protocol stacking, breaking down a pretended TCP virtual circuit from N1 to N2 into four TCP virtual circuits and one SaSTP virtual circuit

The goal of this paper is to present the security architecture designed for SaNTA and the rationale for choosing the architecture. Dealing with security issues is a delicate task, as a single mistake may create a vulnerability in a complex security architecture. Therefore, the recommended approach in this area is to follow well-known security practices, to use well-studied secure protocols and to use highly-tested implementations of security solutions. Using public solutions is also advised because they have been and will continue to be under the scrutiny of a vast set of researchers and users.

Consequently, one important requirement in the design of the SaNTA security architecture was to use standard, mature and well-tested security-related technology. Namely, we decided to study the adoption of existing secure communication protocols (PPTP [3], L2TP [12], IPsec, SSL/TLS and SSH) for protecting communications and common firewall technologies, such as packet filtering of NAT (Network Address Translation [2, 9, 8]) transformations, for protecting SaNTA hosts.

This paper is structured as follows. Section 2 briefly overviews the purpose and the architecture of SaNTA. Section 3 overviews the security requirements of SaNTA infrastructures. Section 4 presents the basic functionality of SaNTA security. Section 5 presents the proposed security architecture for SaNTA infrastructures. Section 6 presents some aspects of the current prototype implementation of the SaNTA security architecture. Finally, section 7 draws some conclusion and shortly presents the main aspects of the proposed SaNTA security architecture.

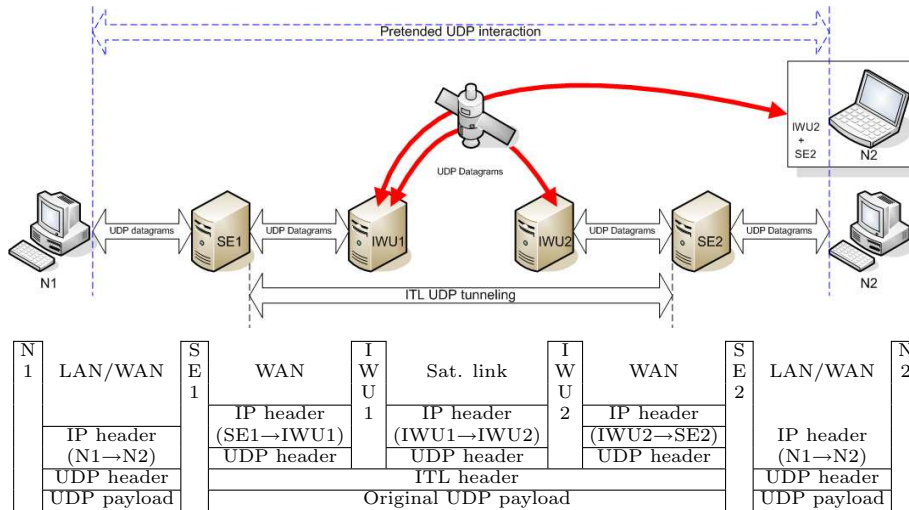


Figure 2: SaNTA UDP tunneling architecture and protocol stacking used for routing UDP datagrams from N1 to N2 through a specific satellite link

2 Overview of SaNTA

The congestion control mechanism of TCP assumes that delays are a symptom of packet losses caused by congestion. This is a simplistic congestion control model that deals poorly with network connections that encompass several physical links with different bandwidth and latency characteristics. Namely, satellite links introduce significant latency delays and, therefore, are likely to reduce the throughput of TCP by triggering its congestion control mechanism. Thus, a problem of the TCP congestion control policy is that it should be enforced along the path of TCP packets but is solely enforced by end systems.

The main goal of SaNTA (Satellite Networks Transport Architecture) was to provide an efficient exploitation of TCP traffic over long-latency satellite links [6, 7]. For doing so SaNTA uses a new transport protocol, ITL (InterTransport Layer), and a set of ITL relays for accelerating the acknowledge of TCP segments and avoiding the traditional management of TCP windows that assumes that data losses are due to congestion. For completeness, ITL also deals with other non-TCP traffic, such as UDP, but with no obvious advantage besides simplifying the management of traffic routing.

The SaNTA architecture is composed by two components that are ITL-aware (see Figures 1 and 2): Satellite Enhancers (SEs) and Inter-Working Units (IWUs). The SE is the end-user component of SaNTA. It acts as a gateway between normal TCP traffic, used by user applications and servers, and ITL traffic managed by SaNTA. The IWU is a core component of SaNTA that is closest to the satellite link. It acts as a gateway between SaNTA ITL traffic and satellite-specific transport protocols,

namely SaSTP (SaNTA Satellite Transport Protocol), a new transport protocol for satellite links that was also developed for SaNTA.

SaNTA components relay TCP connections that are routed into an SE until another SE. The relaying is managed by the ITL layer, standing above TCP. This layer uses TCP for relaying ITL-encapsulated end-user TCP segments and a simple UDP tunnel for relaying ITL-encapsulated end-user UDP datagrams. Other protocols were not considered in the original design of SaNTA.

Thus, a normal TCP connection between end nodes N1 and N2 and using SaNTA results in a series of TCP connections and one SaSTP connection along the communication path between N1 and N2 (see Figure 1). The component SE1 interacts directly with N1 on behalf of N2, thus impersonating N2. Similarly, SE2 interacts directly with N2 on behalf of N1. IWUs interact with each other through SaSTP, a satellite-link-specific transport protocol. The SaSTP runs over IP, itself supported on the satellite native layers 1 and 2.

SaNTA daemons running on SEs and IWUs use only one TCP port and one UDP port for creating TCP connections used by ITL and for tunneling UDP data handled by ITL.

From the strict point of view of traffic acceleration, SEs can be either Internet core routers or routing nodes belonging to end-networks. However, we will see that when security is considered, SEs should be part of end-networks. This topic will be addressed later on this paper.

3 Overview of SaNTA security requirements

A specific SaNTA infrastructure should be similar to a set of Internet core ISPs. Authorized end-nodes can use it to better explore a specific satellite link, but that should be as transparent as possible to end-nodes, applications and end-users. Transparent in this context means that end systems and persons do not have to change dramatically to benefit from SaNTA acceleration and security mechanisms. But it does not mean that end systems and persons should not be aware of SaNTA functionalities and requirements.

Therefore, in what concerns security aspects, SaNTA infrastructures:

- Don't need to provide any security mechanisms directly used by end-nodes and applications.
- Should not interfere with security requirements of end-nodes and applications.
- Should not introduce new vulnerabilities in the traffic they handle.
- Should be used only by authorized end-users or by authorized end-nodes/networks.
- Should not be abused by any Internet nodes.

Thus, in the rest of this section we will provide a first approach to the security aspects of SaNTA concerning the following three major issues:

1. Overcoming possible interferences of SaNTA activity with existing end-user protocols, namely security or security-related protocols, which can be used by SaNTA end-nodes.
2. Providing the proper protection of SaNTA traffic broadcasted by satellites for ground-level antennas managed by IWUs.
3. Providing the proper protection of SaNTA core components, SEs and IWUs, for avoiding abuses from other Internet hosts.

3.1 Interference of security mechanisms with SaNTA

In this section we will address the collision between security mechanisms that may be used by SaNTA end hosts/networks and the internal operation of SaNTA protocols and components.

SaNTA impersonates transport end-points, thus it needs to access and manipulate transport headers, namely TCP and UDP headers. Therefore, all end-to-end secure communication protocols that authenticate transport end-points or hide transport headers are problematic for SaNTA. Considering the most common secure communication protocols, this means that end-nodes N1 and N2 in Figure 1, or security gateways acting on there behalf:

- Should not use layer-2 secure VPNs (e.g., PPTP or L2TP) for encapsulating transport data flows between N1 and N2. Otherwise, SaNTA will not be capable to accelerate the securely encapsulated transport data flows.
- Should not use end-to-end IPSec confidentiality for hiding payload contents between N1 and N2. Otherwise, SaNTA will not be capable to inspect encrypted transport headers.
- Should not use end-to-end IPSec authentication, either with AH or with ESP, for authenticating the IP traffic between N1 and N2. Otherwise, SaNTA will not be capable to impersonate transport end-points.
- Can use any sort of security protocols above the transport layer, such as SSL/TLS, SSH and all the transaction layer secure protocols (e.g., PGP). SaNTA is capable of accelerating application-layer protocols over SSL/TLS (e.g. HTTPS) as it does with any other TCP data flows. The same happens with SSH secure connections, since they use TCP. TCP data flows encapsulated into SSH tunnels cannot be individually accelerated by SaNTA but that is not a problem.

On the contrary, authorization barriers enforced around end-nodes/networks, implemented by packet-filtering firewalls or NAT gateways, do not necessarily introduce new problems for SaNTA. In fact, SaNTA can even facilitate the exploitation of those systems as it encapsulates original TCP and UDP traffic into specific transport ports used by SaNTA components (SEs and IWUs):

- If both N1 and N2 are hidden behind a packet-filtering firewall, they can still be addressed by SaNTA IWUs provided that the closest SE

is also inside the protected perimeter and the SE incoming ports are accessible for a set of authorized IWUs.

- If both N1 and N2 are hidden behind a dynamic NAT gateway, they can still be addressed by SaNTA IWUs provided that the closest SE is also behind the NAT gateway and the SE incoming ports are statically forwarded by the NAT gateway.

Therefore, SaNTA can easily interconnect two networks behind very restrictive packet-filtering firewalls provided that SEs are inside the defense perimeter of end-networks. In this approach SaNTA can be seen as a sort of VPN, as all the traffic is securely sent to the same ITL port. Regarding NAT, the ITL tunneling (or encapsulation) can also be used to simplify the task of port forwarding (all that is necessary is to forward the ITL port). Otherwise, one would have to forward all ports addressed by incoming traffic.

As previously referred, SEs can either be Internet core routers or routing nodes belonging to end-networks. However, placing SEs inside authorized client networks simplifies the security management of the end-networks and the security management of SEs. But other architectures are also possible for dealing with SEs outside the defense perimeter of end-networks. For instance, authorized networks can use layer-2 tunnels between their gateway and the SE for bypassing the NAT mechanisms and ensuring a proper authentication of end-networks when accessing SEs.

Note that SEs aren't necessarily new machines to be placed on end-networks. In fact, SEs can be implemented on top of network gateways already existing on those networks. Furthermore, hosting SEs on gateways simplifies all the IP routing mechanisms: end-nodes don't need to route the traffic differently to reach a local SE, the SE is already in the path to the Internet.

Concluding, the best way to integrate client networks and SEs will always depend on the architecture and requirements of real client networks. But mechanisms used for perimeter defense of end-networks are compatible with the operation of SaNTA.

3.2 Overcoming interferences

As we saw in the previous section, SaNTA cannot work with layer-2 VPN traffic or IPSec traffic between end-nodes N1 and N2. Therefore, SaNTA must (i) handle properly such traffic using a best effort policy; and (i) should internally use security mechanisms for providing a similar protection, allowing end-users to securely abandon their end-to-end security in order to benefit from SaNTA acceleration.

For dealing with PPTP traffic and IPSec traffic, used either directly or within L2TP layer-2 VPNs, the simplest solution for SaNTA is to simply relay it, possibly encapsulating it inside UDP tunnels for keeping a coherent approach for dealing with SE's incoming traffic (from IWUs). Such tunnels are already used by other mechanisms, such as NAT-T [5].

For providing internal security mechanisms, thus allowing end-users to abandon their own end-to-end security mechanisms for using SaNTA acceleration features, a particular SaNTA infrastructure must:

- Be trusted by end-nodes N1 and N2. The trust issue is naturally solved by a contractual relationship between the end-nodes and the company selling the services of a particular SaNTA infrastructure.
- Be authenticated by end-nodes N1 and N2. This means that end-nodes should authenticate the closest external SaNTA component; such component may be an SE or an IWU (if the SE is located in their network and managed by their administration).
- Provide an IPSec-like security between each end-node — N1 or N2 — and the closest external SaNTA component — SE or IWU. This means that end-nodes, or some security gateway on their behalf, should manage IPSec secure associations between end-nodes and an external SE or IWU.
- Provide an IPSec-like security between SaNTA core components. One possible solution was to use IPSec between each pair of components (SE→IWU and IWU→IWU). However, this is not acceptable because of the overhead introduced (this is the technique used by Encore Networks' Selective Encryption Layers [4]). Therefore, within SaNTA one should implement a wider end-to-end security policy for protecting only the transport payload. Original transport headers do not need to be protected as they are replaced by ITL headers. However, the negotiation of ITL connections for handling original TCP connections, or ITL tunnels for handling original UDP datagrams, must be protected for avoiding traffic analysis between end-nodes.

A scenario that may deserve special attention is when SaNTA interconnects two LANs, or a LAN and a mobile host with satellite access, under the same security administration. In this particular case, SEs could properly impersonate hosts on the other extreme, allowing IPSec SAs to be negotiated between an end-host and the closest SE. Such SAs, however, could only protect (i.e., encrypt) the traffic between end-nodes and nearby SEs, since SaNTA components need to manipulate transport-level contents. Therefore, all the security mechanisms previously described would still be necessary, but end-nodes could transparently use IPSec and still benefit from SaNTA acceleration mechanisms.

3.3 Protection of radio-broadcasted traffic

Traffic radio-broadcasted by satellites for ground-level antennas can be protected in three different ways:

- By using link security in low-level satellite communication protocols;
- By using network layer security, such as IPSec, between the closest SaNTA components that explore the satellite link — the IWUs; or
- By relying on SaNTA SEs to properly address the security of traffic routed through satellite links.

All these solutions are compatible and can be enforced simultaneously but that may increase the overhead in the SaNTA computation and the latency of the overall communication. Thus, if SEs already deal with traffic

security issues, the best for performance is probably to rely solely on them for providing the adequate protection for the traffic radio-broadcasted by the satellite.

3.4 Protection of SaNTA from abuses

The SaNTA infrastructure must be protected from abuses, namely from unauthorized access attempts and from denial-of-service (DoS) attacks. This means that (i) SEs should only communicate with authorized end-nodes and with well-known IWUs and (ii) IWUs should only communicate with well-known SEs and IWU peers.

For enforcing this policy, SEs and IWU need to use a packet-filtering firewall and a strong authentication mechanism for authenticating IP peers. Such authentication should be implemented with IPSec authentication (using AH or ESP authentication). Higher level authentication protocols, like the ones used SSL/TLS or SSH, have several drawbacks in this scenario:

- They only handle TCP traffic, and not UDP. Therefore, they are not able to authenticate non-TCP traffic.
- They do not handle DoS attacks using IP spoofing, such as SYN flooding attacks.

4 Basic functionality of SaNTA security

Taking into consideration all the security aspects previously discussed, the SaNTA security should provide the following functionalities:

- Original TCP payloads encapsulated in ITL should be encrypted only once by SE's to provide confidentiality of data across internet networks and in traffic radio-broadcasted by the satellite. This can be done with SSL/TLS or SSH. However, there are two possible approaches: SSL/TLS/SSH on top of ITL or ITL on top of SSL/TLS/SSH.
- Information from original TCP headers used in the negotiation of ITL connections should be encrypted only once by SE's to provide confidentiality of original traffic across internet networks and in traffic radio-broadcasted by the satellite. This could also be done with SSL/TLS or SSH. Unlike in the previous case, SSL/TLS/SSH should be used below the ITL negotiation protocol.
- Original UDP datagrams encapsulated in ITL datagrams should be encrypted only once by SE's to provide confidentiality of original data and traffic across internet networks and in traffic radio-broadcasted by the satellite. The only existing solution for protecting UDP traffic is IPSec.
- SaNTA components, SEs and IWUs, should use packet-filtering firewalls to protect themselves from Internet hosts and restrict incoming traffic to authorized end-nodes and peer SaNTA components.

- SaNTA components, SEs and IWUs, should authenticate all incoming traffic with IPSec for properly enforcing peer authorization and preventing man-in-the-middle and DoS attacks. Such authentication has the side effect of controlling the integrity of IP datagrams and can also tackle IP replay attacks that could affect the integrity of communication flows.

4.1 Host authentication issues

It is advised to have a coherent and comprehensive policy for authenticating SaNTA components and computer nodes interacting with SaNTA extremes (SEs), namely capable of supporting multiple secure communication protocols, such as IPSec, SSL/TLS or SSH.

The authentication of computer nodes can be done in two different ways: (i) using shared secret keys or (ii) using asymmetric cryptography. IPSec can use either, SSL/TLS use the latter, SSH uses a blend of both — servers use the latter, client users can use either. Since IPSec, SSL/TLS and SSH are the candidate technologies for handling the security of SaNTA communications, the best approach is to use only asymmetric cryptography for all cases. Besides, asymmetric cryptography increases scalability.

A traditional issue of using asymmetric cryptography is dealing with Certification Authorities (CAs), certification chains and certificate revocations. We believe that in this case the problem can be simplified. In fact, SaNTA components belonging to the same satellite link provider can use a common root CA deployed and managed by the provider. SEs providing IPSec impersonation for end-hosts may use the already existing certification infrastructure that is used by the impersonated node (all that is necessary is to copy and asymmetric key for an SE).

4.2 Key management issues

Secure communication protocols have their own policies for handling key management or allowing client applications to enforce the most adequate policies. In this section we will shortly resume the policies allowed for each protocol and how they should be used by SaNTA.

Key management deals with the management of Key Encryption Keys, session keys and Perfect Forward Secrecy.

A Key Encryption Key (KEK) is a key that is used solely for distributing session keys; it is not used for bulk data encryption. KEKs may be shared, secret symmetric keys or public components of asymmetric key pairs held by session key receivers. In some protocols two types of KEKs are used: long-term KEKs, used for a primary authentication of peers, and short-term KEKs, used for repeated key distribution along the interaction of the peers.

A session key is a shared, secret symmetric key that is used for bulk data encryption and data authentication within a session; the exact definition of session varies.

Perfect Forward Secrecy (PFS) is a concept that describes the long-term security dependency of a (session) key from other keys (typically

KEKs). A key is said to have PFS if its secrecy does not depend on the secrecy of other long-term keys. The typical way to achieve PFS for session keys is to negotiate them with the Diffie-Hellman key distribution algorithm with ephemeral secret values. If, after the negotiation, the ephemeral secret values are discarded, then no long-term secret values involved in the negotiation of the session keys are maintained, yielding the PFS of the session key. Although PFS is usually desirable, some session key negotiation protocols use simpler protocols with KEKs and nonces for getting a new session key.

The next subsections provide a brief description of the items involved in the key management of IPSec, SSL/TLS and SSH, as well as the way they are handled.

4.2.1 IPSec

IPSec key management issues involve the management of IKE and IPSec SAs. IKE SAs are negotiated with Main Mode or Aggressive Mode and have a short-term KEK with PFS. IPSec SAs are negotiated with Quick Mode and have a session key that may or optionally have PFS. The Quick Mode can enforce PFS, by using ephemeral Diffie-Hellman private values in the session key negotiation protocol, or give up on PFS and use instead nonces for speeding up the protocol.

IKE SAs and IPSec SAs have a maximum lifetime that is administratively decided on their creation and may be renegotiated with IKE some time before expiring. IPSec SAs can expire after processing a maximum number of bytes, to avoid an excessive exploitation of a single session key, or can expire after a given time interval. In any case, an IPSec SA always expires after processing a maximum of 2^{32} datagrams, because of the counter used to detect replayed messages.

Considering what was said at the beginning of this section, IPSec may be used for authenticating traffic between SaNTA entities or for encrypting UDP payloads exchanged between SEs. For the first case, both IPSec SAs can have a long lifetime and IPSec session keys don't need to be refreshed with PFS, because they are not used to encrypt contents but just to compute Message Authentication Codes (MACs). For the second case, IPSec SAs should have a shorter lifetime, there must be a byte-count limit and PFS should be required in each Quick Mode renegotiation. The lifetime of IKE SAs may be long for both cases.

4.2.2 SSL/TLS

SSL/TLS key management issues involve the management of short-term KEKs (master keys) and session keys. The policy to govern these keys is totally under the control of client applications. In fact, interacting applications choose the way they want to negotiate master keys, the lifetime of the master keys, when new session keys are negotiated and their lifetime. Master keys can have PFS if negotiated with Diffie-Hellman and ephemeral secret values, but can be negotiated faster and without PFS using only a random value chosen by the client and sent to the server encrypted with the recipient public key. Session keys are always computed

with nonces and have no PFS.

Considering what is said at the beginning of this section, SSL/TLS may be used for encrypting ITL payloads and possibly ITL setup messages. Thus, master keys should be negotiated with ephemeral Diffie-Hellman private values, to ensure PFS. Session keys negotiated between SaNTA peers for encrypting ITL payloads or setup messages should not be used to encrypt large amounts of data.

4.2.3 SSH

SSH key management issues involve the management of session keys. These are computed with Diffie-Hellman secret values during the authenticated handshake protocol, assuring PFS. Session keys are not refreshed during the session. The connection protocol, that allows the tunneling of TCP streams on top of a SSH secure connection, does not negotiate or use any extra session keys. The lifetime of the session key of a SSH secure connection is the lifetime of the secure connection.

Considering what is said at the beginning of this section, SSH may be used for encrypting ITL payloads and possibly ITL setup messages. Thus, SSH sessions should have a limited lifetime for not over-exposing session keys.

5 SaNTA security architecture

This section describes the technological implementation of the SaNTA security architecture. This technological architecture implements the security policy presented in section 3 with the existing technologies discussed along this document.

5.1 Topological deployment

SEs should be part of the infrastructure of SaNTA clients and should be completely or as much as possible managed by them. This way, end-networks have more freedom to choose the right approach to route authorized traffic into SaNTA through a local SE. Internally SEs should be located where it best suits each network, but a natural location should be within an isolated DMZ or on top of a network gateway. SEs should be protected from inside or outside attacks to prevent network usage abuses involving satellite links and imputable to SEs' owners.

IWUs should be deployed and managed by satellite link providers. IWU should behave as Internet routers with SaNTA capabilities only for authorized SEs.

5.2 Authentication and access control

The authentication and access control between end-hosts and the closest SEs should be managed solely by the administrators of end-networks that SEs belong to. Each SE should interact only with an authorized set of hosts within the end-network it belongs to and with a set of authorized

peer IWUs in the external world. The authentication and confidentiality of traffic between SEs and authorized hosts in the same end-network should be defined and enforced by end-network administrations.

The authentication between SaNTA elements (SEs and IWU) can be enforced in two different ways:

- With asymmetric cryptography and X.509 digital certificates. Such certificates should be issued only by a self-certified CA managed by a satellite link owner and SaNTA provider and are enough for long-term authentication of a consistent set of SaNTA entities within several secure communication protocols, namely IPSec and SSL/TLS.
- With secret, symmetric pre-shared keys. Such keys can be used to authenticate IPSec peers (SEs and IWUs), but not SSL/TLS peers (SEs). However, the authentication of SSL/TLS peers is not critical as we already authenticate IPSec communications between them. But some care must be taken in the use of this authentication policy as badly chose passwords, that are the source of the symmetric pre-shared keys, can be subject to off-line dictionary attacks.

Note that only SaNTA entities certified by a common CA can interact between themselves. This naturally isolates the set of SaNTA entities capable of interacting between themselves, providing a natural authorization mechanism that requires no management. This advantage may partially compensate satellite providers for the burden of having to deploy and manage a CA.

The authentication of traffic between SaNTA entities is provided at IP level using IPSec SAs between pairs of SaNTA entities. Each IPSec SA should use only AH for ensuring authentication of origin, authentication of contents and replay protection.

Each IWU should only accept IPSec AH authenticated datagrams coming from authorized SEs or IWUs. All other traffic from any Internet host should be discarded using a packet-filtering firewall.

5.3 Handling TCP data flows

Handling TCP data flows involves the establishment of ITL virtual circuits between SEs and the encapsulation of TCP payloads into ITL “segments”. SaNTA needs to ensure the security of both operations and using SSL/TLS under ITL should be the preferred way to integrate both actions with an appropriate level of security (see Figure 3).

Putting SSL/TLS on top of ITL would not be a complete solution because it would not protect the setup of ITL virtual circuits. Therefore, using it would only duplicate efforts because something would have to be put under ITL to protect ITL negotiations. Therefore, using SSL/TLS under ITL seems the right approach to achieve both goals with the same kind of integration of technologies.

The choice between SSL/TLS and SSH is mostly derived from the design philosophy of both solutions. SSL/TLS was initially designed as a protocol, being latter made widely accessible to many applications through several libraries (e.g. SSLeay and OpenSSL). SSL/TLS was never

a standalone application, but instead a component to be used in many applications. On the contrary, SSH was from start a secure substitute of insecure remote session tools, such as telnet. Libraries for SSH came much latter and are not often used as SSL/TLS libraries for providing security for legacy application-level protocols. Furthermore, SSL/TLS was designed for handling several types of virtual circuits and to allow flexible key management policies for dealing with the security of each of them. SSH, on the contrary, has a rather simplistic key management policy for secure connections and secure tunneling appears more as an interesting feature (e.g. for tunneling back X11 client-server interactions e a remote session) than a well-established purpose. Therefore, SSL/TLS is more adequate for dealing with the security requirements of SaNTA regarding key management issues and should be easier to incorporate in SaNTA than SSH.

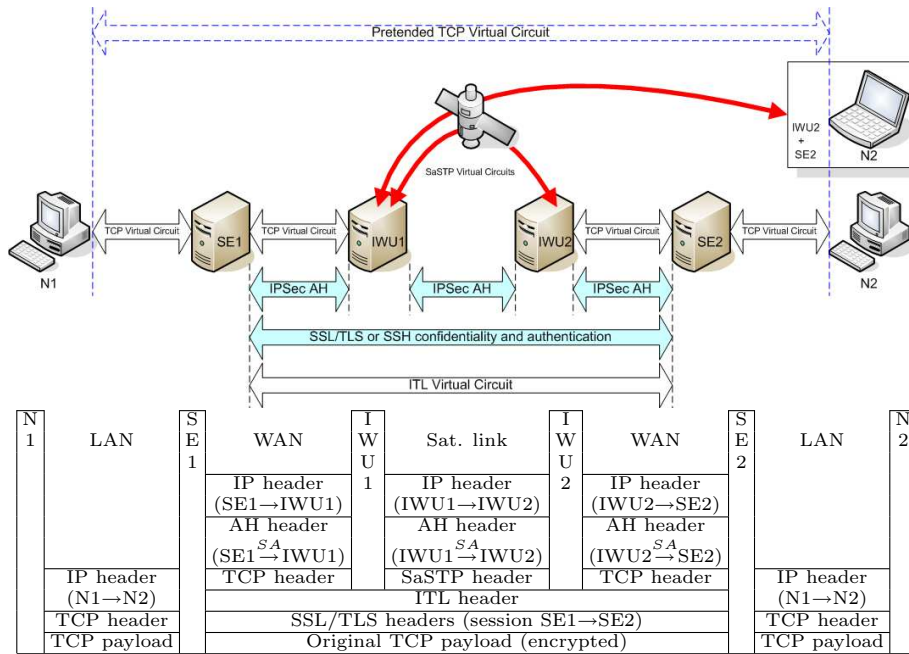


Figure 3: SaNTA security architecture and protocol stacking for handling TCP data flows from N1 to N2

5.4 Handling UDP data flows

UDP data flows are encapsulated into ITL UDP datagrams for properly routing them through a SaNTA infrastructure. Before such encapsulation UDP datagrams should be transformed using an IPSec SA between the local SE and the remote SE at the end-network (see Figure 4). The IPSec SA should use ESP with cipher and no authentication, as authentication is achieved using other mechanisms.

In our opinion using directly IPSec on the received UDP datagrams, before sending them for being tunneled by SaNTA, should be the lighter and most secure way to do this sort of secure encapsulation. Other possibilities, such as layer-2 encapsulation protocols, do not solve the security problem L2TP and introduces extra levels of tunneling that increase the latency and reduce the throughput of the UDP traffic through SaNTA. L2TP, for instance, would encapsulate original UDP datagrams into L2TP UDP datagrams and would use IPSec to protect the encapsulating datagram. This would represent a quite deep level of encapsulation — original UDP datagrams inside PPP inside UDP L2TP datagrams inside ITL UDP datagrams — and more computational effort would be necessary for encrypting L2TP UDP datagrams with IPSec than the original ones. PPTP, on the other side, is not secure enough for being considered for this purpose.

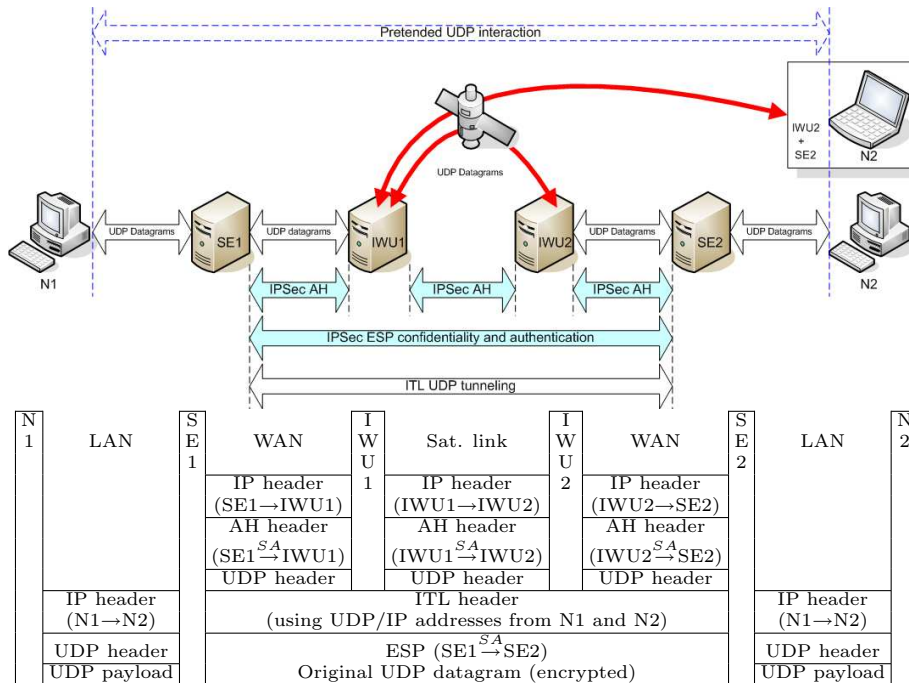


Figure 4: SaNTA security architecture and protocol stacking for handling UDP data flows from N1 to N2

5.5 Handling opaque data flows

The initial SaNTA architecture could not handle data flows other than explicit TCP or UDP data flows. Other data flows are a sort of “opaque data flows”, in the sense that SaNTA is not capable of properly accelerating or routing it. Example of data flows that need encapsulation are ICMP, IPSec or layer-2 VPNs between SaNTA end-networks. When the

architecture was revised for adding security mechanisms it was also added routing support for opaque data flows.

For properly handling the routing of opaque data flows using the existing SaNTA architecture, SEs should have some sort of front-end to client networks capable of encapsulating opaque data flows inside UDP or TCP to benefit from SaNTA services, namely routing and security services. Figure 5 shows how this can be done using UDP encapsulation.

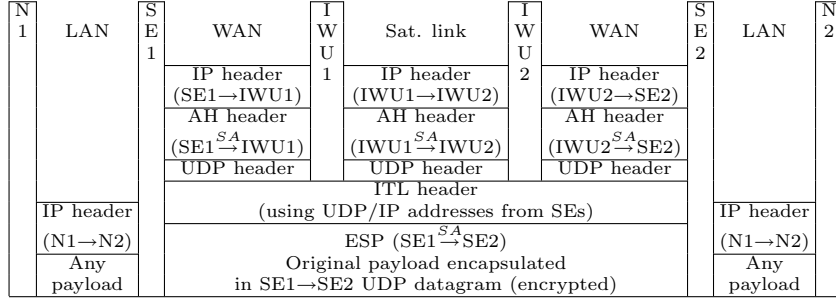


Figure 5: SaNTA security architecture and protocol stacking for handling opaque data flows from N1 to N2

5.6 Troubleshooting

All the tunneling mechanisms described before end up in preventing the use of some end-user ordinary diagnostic mechanisms to assess the proper operation of SaNTA. For instance, the UNIX `traceroute` tool cannot “see” SaNTA entities besides the first SE, because the diagnostic messages are encapsulated and are not directly seen by SaNTA entities. To handle this issue SaNTA should provide the normal behavior for this tool and other related tools.

5.7 Key management

All the session keys negotiated between SaNTA entities should have Perfect Forward Secrecy (PFS) for ensuring long-term security of the data exchanged through a SaNTA infrastructure. However, that has some computational costs as it requires the use of random, ephemeral Diffie-Hellman private values and the execution of Diffie-Hellman key agreement protocols. Therefore, some care must be taken to reduce the number of such negotiations.

For IPSec this is not a complicated task. IKE SAs are used only to negotiate new IPSec SAs, thus not exposing a lot IKE session keys. Thus, these keys and the related SAs can be used for long periods of time (e.g. one week). IPSec SAs are only used for producing MACs for IP datagrams (as we only use AH), therefore the session keys of IPSec SAs are not extensively exposed, allowing IPSec SA to be used also during for long periods of time (e.g. one or two days).

For SSL/TLS this represents a more challenging task. SSL/TLS sessions should be refreshed more often than IKE/IPSec SAs because they are used to encipher data, thus being more sensible to cryptanalytic attacks. SSL/TLS provides the means for refreshing session keys within one session, but without providing PFS. For providing PFS a new session must be negotiated and that may be costly. Nevertheless, the cost depends on the configuration of the SSL/TLS handshake protocol. In fact, SSL/TLS authentication of peers may be dropped and replaced by anonymous SSL/TLS handshake protocols, which are much faster than authenticated handshake protocols.

A possibility for properly dealing with the exact configuration of SSL/TLS key management is to push to end-users the choice of several parameters governing key management (events triggering key refreshment, fast key refreshment without PFS, slower key refreshment with PFS, etc.) and to use one SSL/TLS session per end-user.

6 Implementation

The security functionalities previously described for SaNTA are currently being implemented in a prototype version of the infrastructure. The prototype is based on Linux hosts (Fedora distribution, kernel 2.6) and we are using OpenS/WAN 2.3.1 for IPSec management and OpenSSL for SSL/TLS support.

In the implementation we decided not to change the Linux kernel nor both software packages previously referred, for facilitating the porting of SaNTA for future versions of them. But for fully implementing SaNTA security we add also to extensively use the functionalities of `iptables`, the Linux packet filtering/mangling tool, and the `libipq` library for dealing with IP packets with user-level processes.

The upgrade of SaNTA to include the described security mechanisms involved both the upgrade of the ITL protocol, to include the management of SSL/TLS sessions, an operational configuration of the Linux IPSec for dealing with SAs needed by SaNTA and a complete integration of both actions with several `iptables` rules and user-level SaNTA daemons.

In the current implementation we use SSL/TLS solely for protecting TCP payloads, therefore we only manage SSL/TLS sessions between SEs. In the future we will also provide the possibility to manage and use SSL/TLS sessions between adjacent SaNTA nodes for protecting ITL negotiations. Thus, the SaNTA ITL state machine, written in SDL (Specification and Description Language), is being updated to manage SSL/TLS sessions between SEs and to encrypt/decrypt TCP payloads encapsulated in ITL “segments” using specific source/sink BIOs (OpenSSL I/O abstraction objects).

Concerning IPSec, it is not at all easy to explore it as required by the SaNTA security architecture. In fact, IPSec implementations enforce IPSec transformations when packets are about to leave hosts, therefore it is not straightforward to add the ESP transformations presented in Figures 4 and 5. The same happens when packets with ESP transformations arrive at SEs, it is trivial to validate and remove the outer AH

transformation but not the inner ESP transformation.

In fact, for enforcing both transformations we used a complex packet flow engine using `iptables` rules, `libipq`-based SaNTA daemons and raw sockets. Just to give a idea, this is what happens when a IP datagram from N1 to N2 arrives at SE1:

- if it is a TCP segment, it is redirected by `iptables` into a local TCP port handled by a SaNTA daemon (using the REDIRECT target). The SaNTA daemon will handle the TCP connection with N1 and the local translation of TCP ports will be performed by the `conntrack` feature of `iptables`.
- if it is a non-TCP packet for forwarding from a local network through a SaNTA infrastructures, it is sent to a queue managed by a `libipq`-enabled SaNTA daemon (using the QUEUE target). If the packet is not an UDP datagram, then it first encapsulated into and UDP packet from SE1 to SE2. The original UDP headers are then recorded, the IP addresses are modified to have SE1 as source and SE2 as destination and the packet is sent using a raw socket. The UDP datagram is then transformed by an IPsec ESP SA negotiated between SE1 and SE2. The resulting ESP packet is captured by `iptables` when is about to reach the net (in the POSTROUTING chain) and sent again to the same daemon using the `libipq` queue. The daemon adds the ITL header, build from the saved UDP header, adds another UDP header with a SaNTA UDP port for tunneling and sends the resulting packet from SE1 to IWU1 using a raw socket. This time the UDP datagram is transformed by an IPsec AH SA negotiated between SE1 and IWU1. The resulting AH packet goes then into IWU1 and gets relayed with new IP, AH and UDP headers between SaNTA elements until reaching the SE2.

When the packet finally reaches SE2, the AH transformation is first automatically checked and removed by an IPsec AH SA between IWU2 and SE2 and the resulting UDP packet gets read by a local SaNTA daemon (using the UDP tunneling port used by SaNTA). This daemon removes the ITL header from the datagram, changes the source and destinations addresses to SE1 and SE2 and sends the resulting ESP packet into the loopback interface using a raw socket. This packet is then checked and decrypted by the IPsec ESP SA between SE1 and SE2 and the resulting UDP packet is sent to the same SaNTA daemon using the `libipq` queue. This process removes the UDP header or not, depending if it was added or not by SE1, and finally sends the original packet for N2 using a raw socket.

The previously described process for handling UDP packets is also applied to some IKE/ISAKMP UDP datagrams exchanged between SEs, as they need to be routed through SaNTA IWUs. But is not applied to IKE/ISAKMP UDP datagrams exchanged between adjacent SaNTA entities for negotiating IKE SAs or IPsec AH SAs.

Currently we have not yet integrated both IPsec and SSL/TLS authentications using X.509 certificates issued by a common CA managed by the satellite link owner. In this first prototype we are still using anonymous Diffie-Hellman negotiations for SSL/TLS and either pre-shared KEKs or

RSA public keys. But we plan to use a unique authentication paradigm, based on X.509 certificates, in a near future.

7 Conclusions

The goal of this paper was to present a security architecture for a SaNTA infrastructure. As we saw, SaNTA can be deployed in several different ways and different security requirements from SaNTA clients can lead to different implementations.

We think that the proposed security architecture for SaNTA, based on tunneling techniques, IPSec security and SSL/TLS security, is the most adequate one. It provides a high degree of security by using a set of well-established and free security technologies. SaNTA introduces some new challenges in the way such technologies are to be implemented but that does not imply any modifications in the paradigms and implementation of the referred security technologies.

We also think that the location of SEs in the internal network of end-users as several advantages to enforcing a proper access control to SaNTA services and does not create any vulnerabilities for the SaNTA providers (satellite link owners). Some security issues are still dependent on specific SaNTA deployment scenarios, such as the protection of SaNTA entities, SEs and IWUs, from unauthorized accesses and abuses. We also think that end-users should be able to decide upon some security management issues affecting only then, such as SSL/TLS session key management for protecting in-transit contents.

In short, the security solutions for SaNTA presented in this document can be summarized as follows:

1. **Topological deployment:** SEs are to be fully managed by end-networks; they should be located within an isolated DMZ or on top of a network gateway for allowing public access to peer IWUs and local access to authorized local users/hosts. IWUs will be managed by satellite link providers and act as Internet core routers for a set of authorized SEs.
2. **Authentication:** The authentication between SEs and local users/nodes is defined and enforced by the SE's administrators. SEs and IWUs may authenticate each other with X.509 public key certificates or with secret, shared keys. The former allows SSL/TLS authentication when establishing secure sessions for protecting ITL negotiations and contents, the latter doesn't. SSL/TLS authentication of peers (SEs and IWUs) can be dropped because they communicate over an IPSec authenticated path. IP traffic between SEs and IWUs is authenticated with IPSec in transport mode and AH (Authentication Header). Transport contents exchanged between SEs over SSL/TLS secure sessions is further authenticated with MACs.
3. **Access control:** The access control to SEs for local users/nodes are defined and enforced by the SE's administrators; the access control should consider protection policies and mechanisms to protect it from inbound/outbound attacks. SEs should interact locally only

with an authorized set of users/hosts and externally only an authorized set of peer IWUs. IWUs will only accept authenticated ITL traffic from peer IWUs and authorized SEs; all other traffic (besides routing traffic) should be denied or reduced to a minimum.

4. **Confidentiality:** The confidentiality of traffic and contents between SEs and local users/nodes is defined and enforced by the SE's administrators. IP traffic between SEs is hidden inside secure tunnels providing data and traffic confidentially:
 - Secure UDP tunnels for “opaque” data flows;
 - IPSec ESP tunnels for UDP interactions;
 - SSL/TLS sessions for TCP interactions.

All ITL negotiations between SEs and IWUs may be protected with SSL/TLS confidentiality for enforcing traffic confidentiality between end-nodes/networks.

5. **Key management:** By default all session key negotiations should ensure PFS, which means that they must use the Diffie-Hellman key agreement protocol with ephemeral secret values for deriving new session keys. A reasonable balance must be found for refreshing IKE and IPSec SAs in order to protect the overall security of a SaNTA infrastructure without great performance penalties. A reasonable balance must be found for refreshing SSL/TLS session keys in order to properly protect the confidentiality of ITL negotiations without great performance penalties. Some alternative policies should be made available for SEs to control the refreshing of SSL/TLS session keys that are used to protect the contents exchanged between end-networks.
6. **Cost and quality:** The proposed technologies — IPSec and SSL/TLS — are well defined, studied and used. They are also both widely deployed and freely accessible.

References

- [1] T. Dierks and C. Allen, *The TLS Protocol Version 1.0*, RFC 2246, IETF, January 1999.
- [2] K. Egevang and P. Francis, *The IP Network Address Translator (NAT)*, RFC 1631, IETF, May 1994.
- [3] K. Hamzeh, G. Pall, W. Verthein, J. Taarud, W. Little, and G. Zorn, *Point-to-Point Tunneling Protocol*, RFC 2637, IETF, July 1999.
- [4] Encore Networks Inc., *High Performance VPN Solutions Over Satellite Networks*, Tech. report, January 2004.
- [5] T. Kivinen, B. Swander, A. Huttunen, and V. Volpe, *Negotiation of NAT-Traversal in the IKE*, RFC 3947, IETF, January 2005.
- [6] E. Kristiansen, J. Neves, J. Brázio, G. Pagano, and S. Palazzo, *Rethinking the End-to-end Paradigm — The Missing Protocol Layer*, Proc. of the AIAA Int. Communications Satellite Systems Conf. (IC-SSC 2004) (Monterey, CA, USA), vol. 1, May 2004.

- [7] Erling Kristiansen, Afonso Nunes, José Brázio, and André Zúquete, *Satellite Network Transport Architecture (SaNTA)*, Proc. of the AIAA Int. Communications Satellite Systems Conf. (ICSSC 2005) (Rome, Italy), September 2005.
- [8] P. Srisuresh and K. Egevang, *Traditional IP Network Address Translator (Traditional NAT)*, RFC 3022, IETF, January 2001.
- [9] P. Srisuresh and M. Holdrege, *IP Network Address Translator (NAT) Terminology and Considerations*, RFC 2663, IETF, August 1999.
- [10] *Secure Shell (secsh) Internet Drafts*, <http://www.ietf.org/ids.by.wg/secsh.html>.
- [11] R. Thayer, N. Doraswamy, and R. Glenn, *IP Security Document Roadmap*, RFC 2411, IETF, November 1998.
- [12] W. Townsley, A. Valencia, A. Rubens, G. Pall, G. Zorn, and B. Palter, *Layer Two Tunneling Protocol "L2TP"*, RFC 2661, IETF, August 1999.