Elastic virtual private cloud

Daniel Palomares, Daniel Migault, Hendrik Hendrik, Maryline Laurent

Elastic Virtual Private Cloud

Daniel Palomares
Orange Labs de France
Telecom
38-40 Rue du Général Leclerc,
92130 Issy-les-Moulineaux
daniel.palomares
@orange.com

Daniel Migault
Orange Labs de France
Telecom
38-40 Rue du Général Leclerc,
92130 Issy-les-Moulineaux
daniel.migault
@orange.com

Hendrik H
Orange Labs de France
Telecom
38-40 Rue du Général Leclerc,
92130 Issy-les-Moulineaux
hendrik.hendrik
@orange.com

Maryline Laurent
Institut Mines-TELECOM
UMR CNRS 5157 SAMOVAR
maryline.laurent@telecom-
sudparis.eu

ABSTRACT

Virtual Private Networks (VPN) are usually based on IPsec. However, IPsec has not been designed with elasticity in mind, which makes cluster of security gateways hard to manage for providing high Service Level Agreement (SLA).
Cluster of SGs must be handled, for example, ISPs use VPNs to secure millions of communications when offloading End-Users from Radio Access Networks to alternative access networks as WLAN. Additionally, Virtual Private Cloud (VPC) providers also handle thousands of VPN connections when remote EUs access private clouds.
This paper describes how to provide Traffic Management (TM) and High Availability (HA) for VPN infrastructures by sharing an IPsec context. TM and HA have been implemented and evaluated over a 2-node cluster. We measured their impact on a real time audio streaming service simulating a phone conversation. We found out that over a 3 minute conversation, the impact on QoS measured with POLQA is less than 3%.

Keywords

IPsec, IKEv2, context transfer, Virtual Private Cloud, High Availability, VPN Management, QoS, POLQA.

1. INTRODUCTION

VPN Security Entry Points are usually provided by one SG or two SGs for redundancy purposes. Such VPN architectures do not require *Traffic Management* (TM) mechanisms; and *High Availability* (HA) mechanisms can be performed either by re-establishing a session with the other Security Gateway, or by using HA mechanisms like ClusterIP, which enables transparent failover between SGs for End-Users.

This paper considers large VPN infrastructures with high SLA (Service Level Agreement) that cannot be provided by one or two Security Gateways. This VPN infrastructure requires the load to be distributed among multiple SGs in order to deal with up to millions of simultaneous VPN sessions. Additionally, such architecture requires to be highly scalable, which means nodes within a cluster should be added when resources are required at anytime.

We introduce TM to make possible the transfer of a VPN handled by an overloaded node to the newly added node, and thus increasing/decreasing the load among different nodes. Then, we introduce HA to provide node failover. When a node fails, its traffic is automatically taken in charge by another node, and in a seamless way for the EU.

The mechanisms TM and HA are considering two scenarios: offload and virtual private cloud.

- **Offload**: The aggregate smart-phone traffic in 2017 will be 19 times greater than it is today [3]. To manage this huge amount of mobile data, the operators are offloading their End-Users (EUs) from Radio Access Networks (RANs) to some alternate access network technology (E.g. WLAN). These offloaded EUs must maintain the same security level prior to the offload. Because WLAN are unreliable and untrusted compared to RAN, EUs set a VPN access the core network of the ISP. As a result, the ISPs have to deal with millions of VPNs with high SLA. These VPNs are handled by clusters of security gateways, whose management requires TM and HA.

- **Virtual Private Cloud (VPC)**: networks infrastructures of small and medium-size enterprises (SME) and individuals can be outsourced in the cloud. VPNs are used by the EUs to access their home's network or companies' in a secure manner. Over time, cloud infrastructures must be able to handle thousands of

EUs accessing their network at day time including rush hours. At this point, the providers must ensure scalability and high SLA. Thus, TM and HA features are required.

IPsec was originally not designed for dynamic and clusterized environments, but for an IPsec session to remain installed in the same device during an active VPN session. Evolution of services brought many requirements including mobility, multihoming, interface handover, TM and HA. Currently, MOBIKE is an extension that provides mobility features to IPsec. It enables a client and a SG to update the outer IP address of the VPN. However, the sessions are maintained between the same devices.

This paper describes TM and HA that move, seamlessly to the EU, the VPN attached to SG_a to SG_b, another SG of the cluster. Unlike mobility/multihoming, the two SGs concern different hardware. As a result, the whole IPsec context must be transfered from SG_a to SG_b. This operation remains transparent to the EU because both the SG_a and SG_b share the IP address of the cluster.

When an active VPN is transferred from one node to another, IPsec counters may happen to be desynchronized. In fact, an IPsec session has an associated *sequence number* intended to avoid replay attacks. This *sequence number* controls every incoming/outgoing IP packet protected with IPsec, and thus, becomes a very volatile and difficult value to remain updated among several nodes.

The key elements for TM and HA are to define an IPsec context and synchronize the counters associated to an IPsec session, avoiding stale values that might cause longer interruption while transferring a VPN session.

Throughout this paper, section 2 introduces some related work concerning IPsec facing mobility, context transfer and failover. Section 3 defines the two mechanisms intended to provide TM and HA for IPsec SGs. Following section 4, introduces the constraints of transferring an IPsec context between different nodes. Then, section 6 shows our experimental results with real implementation testbeds. Finally, conclusions and future works are given in section 7.

2. RELATED WORK

This section positions our work towards existing IPsec mechanisms, protocols and other publications:

MOBIKE [6] is a mobility and multihoming extension for the Internet Key Exchange protocol (IKEv2). Mobility makes possible to update the IP address associated to one of the extremities of a VPN. Multihoming makes possible to configure alternates IP address for the VPN session. These alternate IP addresses should be used in case the running IP address is not reachable anymore. With Mobility, the communication is established between two entities, the EU and the SG (or the Home Agent). These two entities remain the same before and after a mobility or multihoming operation occurs. Only the IP address of the EU is changed. On the other hand, with IKEv2/IPsec context transfer, the two entities before the context transfer and after the context transfer are different. In our case the SG is a different piece of hardware (device).

REDIRECT [5] is also an extension for IKEv2. It has been designed to redirect an IKEv2/IPsec session from one SG to another. The SG sends a REDIRECT message to the EU, indicating the new SG to attach. When the EU receives this message, it breaks the VPN and the associated Security Associations (SAs) established with the currently active SG and renegotiates all SAs with the new SG. Note that the EU is forced to renegotiate all the security parameters from scratch when being redirected to another SG. This may impact the EU due to network delays while establishing a new VPN towards another SG. REDIRECT does not consider VPN context transfers, which could actually ensure continuity of the VPN service and improve EU's experience.

Georgiades et al. in [7] exposed a theoretical case of study for homogeneous and heterogeneous security context transfer, avoiding renegotiation of all the IPsec parameters from scratch. The paper also discusses how a (IPsec) security context can be associated to an EU profile in RADIUS, Diameter. However, no details concerning the IPsec context are given. This study does not include any performance test or real implementation nor any simulation.

Allard et al. in [1] already addressed the transfer of an IKE/IPsec context. It proposes a MOBIKE extension in order to avoid collision of SA. For this matter, all the parameters for both IKEv1/IPsec and IKEv2/IPsec contexts are well identified. However, the motivation to perform a security context transfer was to make mobility in a Mobile IP environment faster. TM and HA differ from mobility, as the transfer is initiated by the EU and thus removes IPsec counter synchronization constraints (refer to section 4). Then, it also lightens the scalability constraints as EU are unlikely to perform massive transfers at the same time, which is the case of failover events for overloaded SGs in HA environments. Finally, in an implementation point of view, *Allard et al.* use IKEv1 in order to perform some tests, whereas our implementation is based on IKEv2.

3. ARCHITECTURE DEFINITION

This section introduces TM and HA mechanisms. Figure 1 illustrates the VPC architecture principle. EUs placed within an UNTRUSTED zone first establish a VPN with the cluster of SGs to access services in the TRUSTED zone. All the cluster members are configured under a single IP address (E.g. using clustering mechanism like VRRP [8], HSRP [2], ClusterIP [9], etc), however only one node can be responsible of a single VPN session for some EU.

Note also that how the load is distributed among the cluster members is out of the scope of this paper. Our proposal concentrates in providing elasticity to IPsec, so it can be dynamically transferred between physically different SGs.

3.1 Traffic Management Mechanism

Figure 2 illustrates the TM, which proposes dynamic management of VPN sessions. VPC providers interested in managing their VPN traffic within a cluster of SGs, can use TM to balance the load among different cluster members. For example, in the case where one specific SG is getting over-

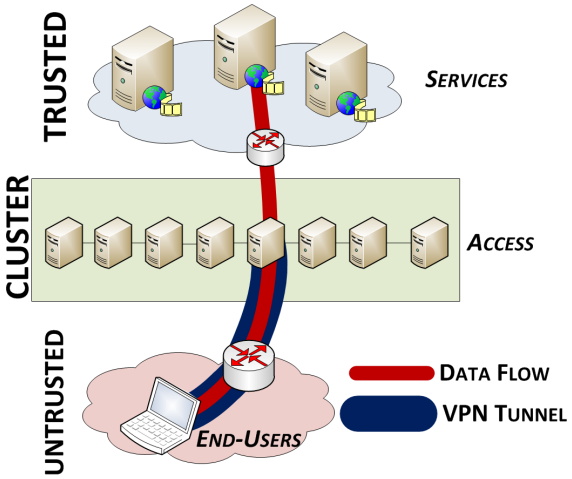


Figure 1: Virtual Private Cloud Architecture

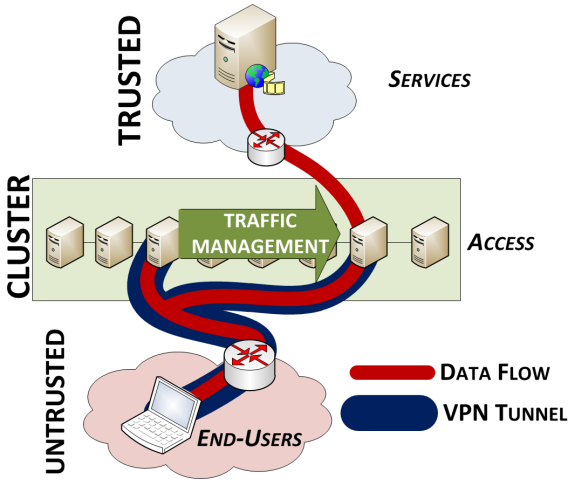


Figure 2: Traffic Management architecture for Virtual Private Clouds

loaded, the TM mechanism allows to transfer VPN sessions from the overloaded SG to some other nodes within the cluster. This avoids failures due to overloaded SGs and improves EU's experience.

In figure 2, an EU first establishes a VPN session with the VPC to access services in the TRUSTED zone. TM offers the possibility to move the tunnel from one SG to another within the cluster. All the information concerning the VPN must be transferred between SGs. More specifically, IPsec information must be transfered, which includes all the cryptographic material, together with the ID, traffic selectors, timers and counters associated. The key advantage of transferring IPsec information is to prevent the EU to re-establish a new VPN from scratch, with an additional authentication which would interrupt the VPN and the applications of the EU. Transferring the VPN or IPsec information reduces the interruption of the session up to an unnoticeable event. In fact our measurement shows that it has minimal impact on the QoS of the applications. Section IV details how to minimize the impact of TM on the applications. Note that the security model remains valid as the SGs have a trusted relation between each other.

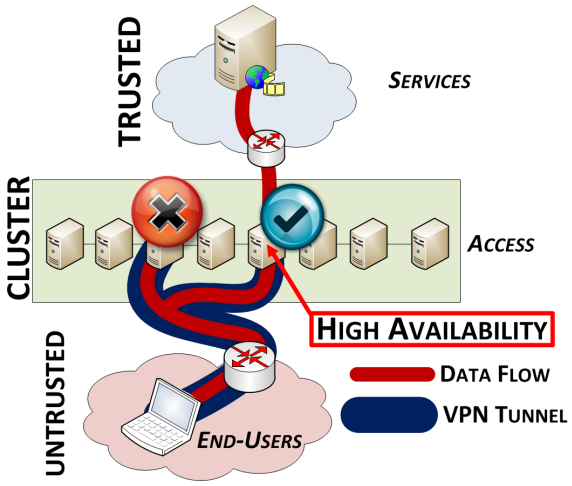


Figure 3: High Availability architecture for Virtual Private Clouds

3.2 High Availability Mechanism

Figure 3 shows how HA is provided for a cluster of SGs. VPC providers interested in providing failover, can use HA in order to ensure connectivity even in a case of a failure. In contrast with TM, failover is performed only when a failure occurs whereas TM can be launched at any particular time. As such, the nodes within the cluster must be capable to detect a failure among the cluster members and to take decisions about the distribution of the affected VPN tunnels among the remaining cluster members.

HA includes two functions in order to ensure connectivity: the **Heartbeat** and the **Sync** functions. **Heartbeat** checks the aliveness of all nodes within the cluster and alerts when a failure occurs. An active VPN is established between the EU and a SG. If this SG fails, then an alternate SG must take the VPN session in charge and become the responsible SG for that VPN. This requires clear mapping SG and alternate SG. This mapping can have different granularity. It can be defined on a per SG basis, that is to say each SG as an alternate SG takes the whole traffic of the failed SG. This means that the alternate SG will run with twice more traffic. For scalability reasons, we recommend that alternate SGs to be defined on a per VPN basis fashion. This allows to distribute the load of all affected VPN sessions among all other SGs, by taking limited fractions of the traffic. If the cluster is made of n SGs, then the overhead will be $\frac{1}{n-1}$ on all remaining SGs. Thus, prior to the failure, the **Sync** function periodically maintains the VPN session parameters synchronized among all the cluster members, whereas the **Heartbeat** frequently checks aliveness of all nodes.

Similarly to our TM architecture, there is a risk that VPN session counters (*Messages IDs* and *sequence numbers*) become desynchronized. How to overcome this issue is explained in detail in section 4.

4. CONTEXT TRANSFER CONSTRAINTS

The EU setting up a VPN, first establishes a secure signaling channel. This secure channel is the IKEv2 channel and is protected by the IKE_SAs. This channel is used to control and negotiate the IPsec_SAs. Each IKEv2 message includes a header with its corresponding *message ID*. Note that the specificity of IKEv2 is that it is an application that uses encryption and authentication through an IKE_SA to protect its messages. On the other hand, IPsec_SAs are implemented in the kernel space and thus cannot be easily accessed by the applications. However, because IKEv2 has a deep understanding of the IPsec it can derive most of the IPsec parameters stored in the kernel. As a result these parameters can be considered as being replicated in the user space. This is only valid for IKEv2.

The IPsec_SAs also have an associated counter called *sequence numbers*, which are increased whenever an incoming/outgoing IP packet is protected with IPsec. Duplicated usage of a *sequence number* is forbidden, providing anti-replay protection to the data flow. Note that these counters and all the cryptographic material are store within the kernel. IKEv2 applications only store the static IPsec_SA information within the application, but dynamic data is maintained by the kernel.

As a result, transferring a VPN session from one SG to another requires to transmit information about the IKE_SAs and the IPsec_SA. Keeping this information is the only way to ensure the continuity of the VPN service. However, not only the static data (i.e. the encryption/authentication keys, algorithms, timers, etc.) but the volatile data like *message IDs* and *sequence numbers*, are also involved in the transfer of a VPN session.

4.1 IKE_SAs constraints

Challenges and solutions concerning the IKE_SA when implementing TM and HA are:

- *Stale Value of Message ID:* during a VPN context transfer, it is possible that the newly responsible SG is not aware of the last IKE response sent by the cluster. If this ever happens, the *message IDs* used by the new responsible SG are stale. Actually, an IKE_SA needs to update its *message IDs* very often, because processing an exchange with higher ID value is not allowed. This is achieved by synchronizing the message ID counters very frequently or even immediately after each signaling exchange.

- *Unacknowledged Request:* it may happen that the new responsible SG is unaware of the last IKE request received within the cluster, thus the counters are stale. Receiving an unexpected *message ID* response would result in discarding the packet, leading to IKE_SAs destruction. The only way to reduce this risk, is to synchronize the *message IDs* after each exchange, however there is no possible way to completely remove this possibility. Although there are new standards that allow renegotiation of counters (i.e. RFC6311 in [11]), this would impact the EU's experience in terms of number of exchanges and delays to reestablish the session.

4.2 IPsec_SAs constraints

Challenges and solutions concerning the IPsec_SA when implementing TM and HA are:

- *Stale Sequence Number value:* whenever a SG takes responsibility for a given active VPN session, it may happen that the *sequence numbers* are out of date. This occurs when the newly responsible SG starts sending IPsec protected packets with stale *sequence number* packets. IPsec includes an anti-replay mechanism that rejects any packet with too low *sequence numbers*. In our case the anti-replay mechanisms makes the EU discard all incoming IPsec packets. Instead, note that IPsec standard allows to increase the value of *sequence numbers* at any time, even without preventing the EUs. Thus, the communication remains uninterrupted. However, *sequence numbers* are stored in the kernel of the system and change very quick. It is necessary to use kernel libraries that involve modification of such counters when updating these values.

5. TESTBED DESCRIPTION

This section describes our real implementation for TM and HA mechanisms. Our developments are based on StrongSwan 5.0, which is a complete OpenSource IPsec-based VPN Solution for Linux. Measurements are represented in graphs with a box-and-whiskers style, which is used to plot statistical data. For every measure (aprox. 50 samples per measure), the box-and-whisker plot indicates the smallest observation, the lower quartile, the upper quartile, the largest observation and the median.

Our testbeds are composed of three computers: two Dell laptops LATITUDE E4300 performing as client and server with intel Core 2 processors, 2GB of RAM, 100Mbps ethernet NIC, running on Ubuntu 12.04 LTS, and one Dell Desktop PRECISION T3500 PC with 4GB RAM intel Xeon, four 100Mbps ethernet NICs and running on Ubuntu 12.04 LTS as well. For performance measurements purposes, our VPC cluster is composed of two SGs, where each SG is built with a Virtual Machine running Ubuntu 12.04 LTS within the Dell Destkop PC.

Our TM testbed is represented in figure 4. The cluster is configured so that SG1 transfers a VPN context towards SG2 during an active VPN session. As the cluster is configured with a unique IP address, the TM is transparent to the EU and the communication is reestablished as soon as the session is transferred and installed on SG2. Our HA testbed is represented in figure 5. It includes the synchronization and aliveness functions: Sync and Heartbeat respectively (refer to subsection 3.2 for details). The cluster is configured to offer HA capabilities, like failover. It is built as a hot-standby set of SGs, where only one of the SGs is active at a given time.

5.1 Performance Measurements

The scenario we are considering is: initially, an EU establishes a VPN towards the VPC cluster (which is composed of two SGs). This allows the EU to reach the audio streaming server in a secure manner. During the VPN establishment, SG1 is always initially responsible of incoming connections.

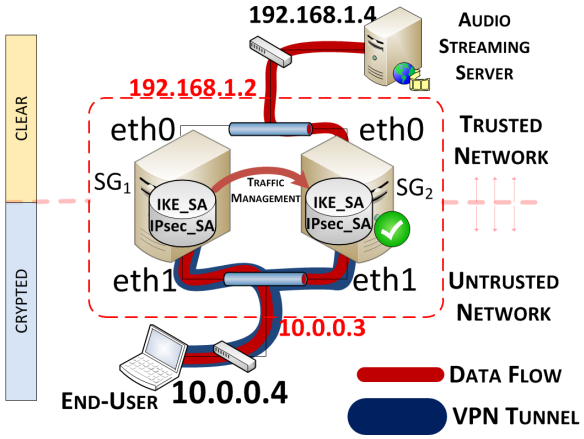


Figure 4: Virtual Private Cloud testbed for Traffic Management

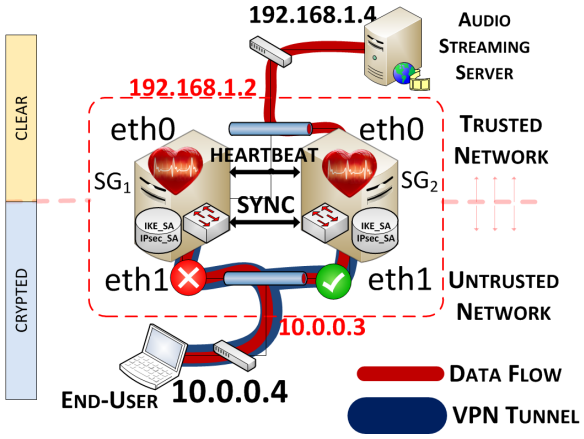


Figure 5: Virtual Private Cloud testbed for High Availability

Then, the EU receives an audio streaming with a duration of 8*sec*. Our performance tests are conducted either with HA (heartbeat and sync daemons activated) or with TM, as follows:

- HA tests: in the HA scenario, the Heartbeat and Sync functions take place every second. It means that, during an active VPN session, SG2 maintains synchronization of IKE_SAs and IPsec_SAs every second, and also detects whether SG1 is still responding. During the audio streaming of 8*sec*, we caused a failure in SG1. The Heartbeat function on SG2 makes detection and installs the VPN session that is previously synchronized through the Sync function. When the IKE_SAs and IPsec_SAs are installed on SG2, the VPN session is thus transferred towards SG2, becoming the new responsible entry point for the affected EU.

- TM tests: in the TM scenario, our test consists in transferring a VPN session from SG1 to SG2 during the audio streaming of 8*sec*. In contrast with HA, there is no failure in SG1 nor detection of interruption by SG2. During TM tests, we only perform a transfer of a VPN session and we measure the impact over the audio streaming.

We considered different parameters during our performance tests:

- RTSP vs. HTTP: we concentrated on two protocols for the audio streaming transmission, RTSP and HTTP. RTSP stands for Real Time Streaming Protocol, whereas HTTP stands for Hypertext Transfer Protocol. RTSP is based on UDP whereas HTTP is based on TCP, so we can measure the impact over connection-oriented and non connection-oriented protocols.

- CBC vs. CTR: in terms of encryption, we test two different encryption algorithms, AES128-CBC and AES128-CTR. Both encryption algorithms use AES128 as block cipher. However, different modes of operation are tested: Cipher-Block Chaining (CBC) and CounTeR (CTR).

- Bit-Rates: we tested three different bit-rates 8Kbps, 48Kbps and 96Kbps for each scenario. The idea is to measure the impact that TM and HA architectures have over different types of data bit-rate.

- POLQA: in terms of quality of service (QoS) measurements, we use *Perceptual Objective Listening Quality Assessment*, POLQA (see [10]). It is a relatively recent standard (2006-2011) for voice quality testing technology which is available under license. The performance evaluation and the impact of our developments is done with this software. In addition, POLQA requires a 8*sec* duration audio when performing evaluation of QoS, this is why we performed audio streaming with audio files of this length. Note that POLQA qualifies the audio file QoS from 1 (worst quality) to 5 (best quality).

6. EXPERIMENTAL RESULTS

This section presents experimental measurements for TM in figure 6 and HA in figure 7. Evaluation of the impacts of TM and HA is provided by measuring networking aspects and QoS aspects. Networking aspects consist in measuring how long the communication is interrupted, whereas QoS aspects consists in measuring through POLQA, how the QoS of an audio streaming service is impacted.

6.1 Traffic Management results

Figures 6a and 6b illustrate the total interruption time and the impact over QoS respectively. The following subsection aims to show the analysis of the results obtained through experimental measurements.

UDP services are more sensitive to TM than TCP

Figure 6b depicts how the audio streaming quality is downgraded when performing TM, even though UDP has a smaller network interruption time than TCP. In fact, POLQA estimated an average reduction of 35% for an HTTP audio streaming and 27.5% for RTSP audio streaming. However, this is not the case for 8Kbps bit-rate, which is not taken into account due to its bad quality transmissions and bad qualifications.

Encryption CBC/CTR has no impact on TM

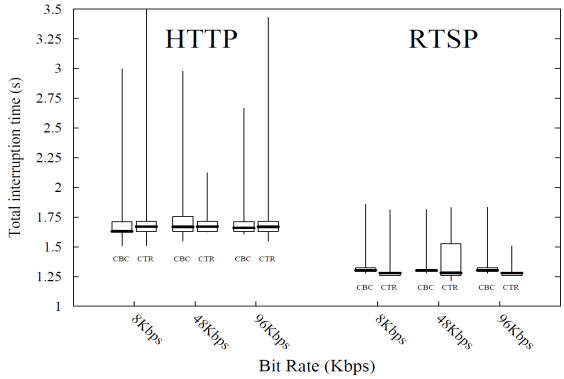
The impacts at the network layer are more significant for those applications based on connection-oriented protocols like TCP than non connection-oriented protocols like UDP. On the other hand, there is almost no difference between different bit-rates or between different encryption methods.

Interruption time is less for UDP than TCP

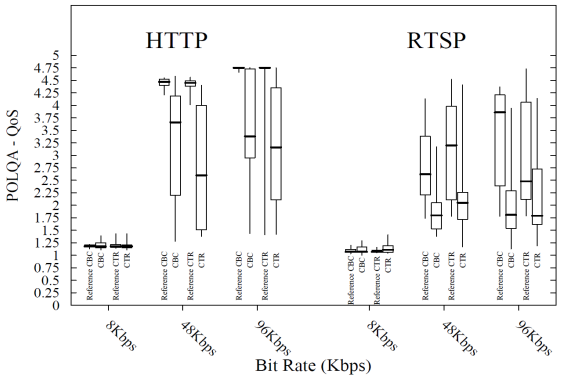
Figure 6a shows that audio streaming based on HTTP is interrupted for around $1.7sec$, even though some results give more than $3sec$, due to TCP retransmission wireless management. In contrast, RTSP is interrupted for around $1.33sec$ in average during TM. As RTSP runs over UDP, no retransmission of packets is done when losing packets, resulting in a quicker reestablishment of the VPN session.

TM's impact over VoIP applications

Melo et al. studied in [4] the duration distribution of more than one billion phone calls during the year 2010. The average time for a phone call resulted in $2min$. In a scenario like this, even if the impacts of our testbed are considerably high for an audio streaming with a duration of $8sec$, these evaluations are not representative for real phone call durations like in [4]. Thus, considering a $2min$ phone call length, the impact on the QoS becomes 2.33% and 1.83% for HTTP and RTSP respectively. This represents quite good results in terms of QoS.



(a) Network interruption time for TM

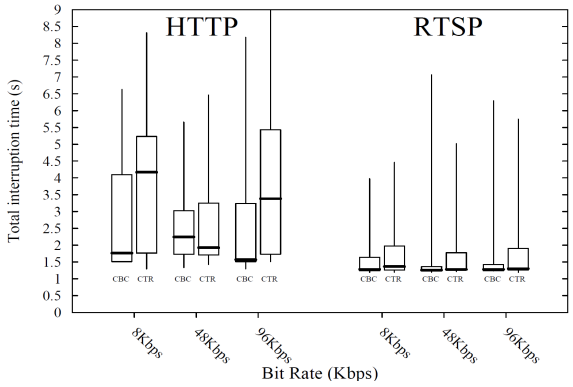


(b) QoS impact for TM

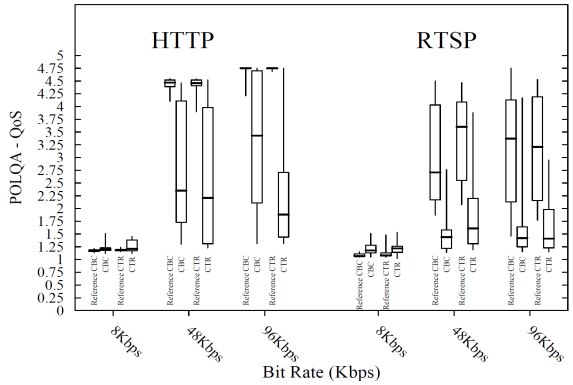
Figure 6: Performance measurements for *Traffic Management*

6.2 High Availability results

Figures 7a and 7b show the total interruption time and the impact over QoS for our HA testbed. The following subsection aims to show the analysis of the results obtained through experimental measurements.



(a) Network interruption time for HA



(b) QoS impact for HA

Figure 7: Performance measurements for *High Availability*

UDP is less impacted than TCP Services

We analyze the impact over the QoS with HA on figure 7b. POLQA estimated an average reduction of the QoS around 38% for an HTTP audio streaming and 29.75% for RTSP-based streaming. As for TM, 8Kbps registered the worst quality, and it was not possible to measure the impact for this particular bit-rate. The QoS is more impacted during HA than TM. Actually, HA takes more time to reestablish a VPN session due to the Heartbeat function, which adds an additional delay.

Interruption time is less for UDP than TCP

Figure 7a showed that HTTP audio streaming is interrupted in average $2.37sec$. However, some of the measures resulted in more than $5sec$ to get reestablished. This is due to TCP window retransmissions when packet loss happens. In fact, when performing a context transfer during an active TCP session, some packets are dropped since the VPN is not yet installed at SG2. The retransmitted packets are lost and will not be acknowledged. The server is retransmitting much later which results in more delays to reestablish the VPN session.

Additionally to this TCP issue, the Heartbeat module of the HA testbed introduces new delays for VPN session reestablishment. In the worst case, SG2 can take up to one second to realize that SG1 is not responding anymore. Once the failure is detected, SG2 installs the previously synchronized VPN session, becoming responsible of the affected EUs.

In contrast to HTTP, RTSP shows better performance during HA. The audio streaming is interrupted in average for a period of time of $1.51sec$. The main reason why RTSP performs better than HTTP is that it is based on UDP, which does not require retransmission of lost packets.

HA's impact over VoIP applications

Finally, estimation of the impact on the QoS should be considered for audio length longer than $8sec$. As cited in section 6.1, Melo et al. showed in [4], that an average phone call has a duration of $2min$. The impact of HA in $2min$ phone call is only 2.53% for HTTP and 1.91% for RTSP.

6.3 HA vs. TM comparison

Besides HA results, it is noticeable that TM has better performance than HA. This is due to the Heartbeat function additional delay. Actually, TM does not include failure detection through a Heartbeat, whereas HA scenario does. This detection can take up to one second when the Heartbeat is set to this value. Reducing the Heartbeat frequency might improve the HA overall results, however, for developments reasons, we can not reduce the Heartbeat beyond this value.

7. CONCLUSION

This paper proposes a mechanism that provides elasticity and increases reliability for IPsec-based Virtual Private Clouds. *Traffic Management* (TM) and *High Availability* (HA) mechanisms ensure the continuity of a VPN service within a same administrative domain or Virtual Private Cloud. In a SG cluster scenario, all nodes are reachable through a single IP

address which makes the transfer of a VPN session between SGs transparent for the EU.

Our results in terms of QoS, demonstrate that the impact of these mechanisms over a $8sec$ audio streaming are 35% (HTTP) and 27.5% (UDP) for TM and 38% (HTTP) and 29.75% (UDP) for HA. However, considering a phone call of $2min$ length, the impact is less than 3%.

Future works include the study of TM and HA between different administrative domains, where the transfer of VPN sessions occurs among SGs owning different IP addresses. A prototype using a mobility extension of IKEv2 called MOBIKE has been designed, and there are some ongoing developments. We also estimate that further investigations should consider designing an algorithm in order to distribute the load on clusters composed by more than 2 nodes.

8. REFERENCES

[1] Allard, F. *Le transfert de contexte : atout pour la mobilité et outil de réduction des coûts pour la sécurité*. PhD thesis, RSM - Dépt. Réseaux, Sécurité et Multimédia (Institut Mines-Télécom-Télécom Bretagne-UEB), 2009.

[2] Cisco. Hot standby router protocol (hsrp).

[3] Cisco. Cisco visual networking index: Global mobile data traffic forecast update, 2012–2017.

[4] de Melo, P. O. S. V., Akoglu, L., Faloutsos, C., and Loureiro, A. A. F. Surprising patterns for the call duration distribution of mobile phone users. In *ECML/PKDD (3)* (2010), J. L. Balcázar, F. Bonchi, A. Gionis, and M. Sebag, Eds., vol. 6323 of *Lecture Notes in Computer Science*, Springer, pp. 354–369.

[5] Devarapalli, V., and Weniger, K. Redirect Mechanism for the Internet Key Exchange Protocol Version 2 (IKEv2). RFC 5685 (Proposed Standard), Nov. 2009.

[6] Eronen, P. IKEv2 Mobility and Multihoming Protocol (MOBIKE). RFC 4555 (Proposed Standard), June 2006.

[7] Georgiades, M., Wang, H., and Rafazolli, R. Security of context transfer in future wireless communications. In *Wireless World Research Forum (WWRF)* (Toronto, Canada, Nov 2004).

[8] Hinden, R. Virtual Router Redundancy Protocol (VRRP). RFC 3768 (Draft Standard), Apr. 2004. Obsoleted by RFC 5798.

[9] Keene, R. Clusterip linux kernel module.

[10] POLQA. Perceptual Objective Listening Quality Assessment - POLQA.

[11] Singh, R., Kalyani, G., Nir, Y., Sheffer, Y., and Zhang, D. Protocol Support for High Availability of IKEv2/IPsec. RFC 6311 (Proposed Standard), July 2011.