Implementing Trust-to-Trust with Customer Edge Switching

Raimo A. Kantola, *Member, IEEE,*
Department of Communications and Networking
Aalto University
Helsinki, Finland
raimo.kantola@tkk.fi

***Abstract* — A Network Address Translator allows hosts in a private address space to communicate with servers in the public Internet. There is no accepted solution for an arbitrary host from the public IP network to initiate communication with a host in a private address space although attempts have been made to create one. This paper proposes the replace NATs with a more comprehensive concept we call Customer Edge Switching (CES). Customer edge switching assumes connection state on the trust boundary between the user and the core networks. The connection state is managed by implicit signaling. The state gives means for the private network operator to apply elaborate access control to packet flows arriving from the Internet to the private network. CES is a way of moving from the end-to-end principle to the trust-to-trust principle advocated by Dave Clark.**

***Keywords* — Network Address Translator, NAT traversal, trust, user identity.**

I. INTRODUCTION

For a long time, the Internet community could not make up its mind on the issue of end-to-end or accepting the use of middle boxes such as NATs. The latter were adopted by the industry because they met a need of extending the IP address space by private addressing and hiding customer networks from the Internet. IETF took a long time to recognize NATs and started to explore how to traverse NATs only after peer-to-peer applications had started making fun of doing so.

Now, after the first failed attempt by IETF, there is a new description of NAT behavior and the IETF has gone on to create recommendations on how to actually traverse NATs [1]. The result is the Unilateral Self Address Fixing architecture (UNSAF). The architecture consists of the STUN protocol for spying what the NAT is doing between the user's device and the global Internet and solution documents such as Interactive Connectivity Establishment (ICE) and SIP Outbound that describe how to make use of STUN in particular kinds of applications. The architecture assumes that applications will want to exchange information about the IP addresses and ports NATs have assigned for the user for the duration of a session.

The UNSAF architecture requires a STUN client in the hosts and STUN servers to be deployed in the global network but it leaves the NATs themselves as they are. UNSAF uses IP addresses for identification. It clutters applications with code that has nothing to do with the task of the application. It scales poorly particularly for mobile hosts. It does not help in deploying servers in private address space and thus hampers the user innovation potential.

An application on a user device such as a mobile terminal that wants to be reachable needs to maintain a NAT mapping by a keep-alive mechanism that wakes the device at regular intervals and keeps the NAT state alive. If the NAT applies the most typical policy of address and port dependent filtering, each application on the device needs to send its own keep-alives. For being able to use a single mapping in the NAT, UNSAF proposes to deploy a network based TURN server for relaying all traffic to and from the host leading to high cost.

Examples of applications that users may want to activate on their mobiles are voice over IP clients, www-servers and mobile peer to peer applications. If even one such application is activated, it drains the mobile's battery quickly. An example how the keep-alive process has been optimized is by implementing SIP registrations on a preprocessor rather than the main processor on a Nokia mobile phone. When first used this helped to increase battery lifetime from a few hours to less than 24h.

Compare this to the location update process for circuit switched telephony on mobiles. For CS telephony the mobile usually stays reachable for several days without the need of recharging the battery. The difference in battery lifetime can be explained by the difference in complexity. It is not feasible to burn on a small piece of silicon all the keep-alive functions required by all applications for packet switched services in mobile devices. Due to simplicity and uniformity of the CS architecture, reachability of the mobile at all times is efficiently implemented by the location update and paging process and their synchronization with the sleep-wake-up cycle of the mobile. In case of circuit switched mobile services, reachability is a low level function upon which an application like telephony can rely. In case of packet access to mobiles, reachability is a matter for each application.

The purpose of this paper is to propose an architecture that makes hosts in private address space reachable globally. The architecture does not require any keep-alive signaling nor registrations from a host. This is optimal for a mobile host that sleeps most of the time for battery saving while wishing to stay reachable and while there is no useful application activity.

The rest of the paper is organized as follows. Section II discusses related work. Section III sets the requirements. To set the stage for the solution, Section IV discusses the difference between routing and switching. The solution is described in Section V. The issues of deployment are addressed in Section VI, Section VII compares our proposal with an IETF endorsed protocol and finally Section VIII concludes.

II. Related Work

NAT traversal has been studied by other researchers. Most recently Miyazaki [2, 3] has explored some new solutions. We summarize the proposed solutions in Table 1.

TABLE 1: NAT TRAVERSAL INITIATED BY A GLOBAL NODE

	Changes required in			
Solution	GN	PN	NAT	Other
STUN	No	Yes	No	STUN Server
AVES	No	No	Yes	Waypoint Server DNS
NAT-f	Yes	No	Yes	DDNS
P2P/UDP	Yes	Yes	No	Master Server
Extended RSIP	No	Yes	Yes	RSA-IP Server DDNS
DPRP	No	Yes	Yes	
NATS	No	Yes	Yes	NTS Server

GN – Global Node/host, PN – Node in a Private Network, NAT – Network Address Translator

Many solutions take the attitude that hosts must adapt to work with NATs possibly with the help from some servers. A typical solution introduces switching state in a server and some *explicit signaling* for maintaining that state. Switching takes place on the IP layer. Several of the solutions suffer from additional configuration information that needs to be maintained.

IETF is working on the Locator/ID Separation Protocol [10] for the purpose of tackling the scaling issues in the current Internet. We will compare our solution to LISP in Section VII.

All the research papers that we have been able to find concentrate on the technical problem of traversing a NAT box. We argue that this misses an important point. A private network administrator has a legitimate need to hide its network and thus protect it from attacks. We believe that it is as important to address this need as it is to traverse the NAT.

III. Requirements

Let us set the requirement that a host having a private address can send messages to hosts that also are in their own private address spaces. Examples where this can apply are hosts with private IPv4 addresses and hosts with 802.1 MAC addresses that also are not globally unique. Moreover, the solution should assume that the hosts are mobile and may roam foreign networks. Let us formulate this idea of communication using private addresses into a clear principle. The principle is that *a private network shall not publish any routing addresses to the public domain and equally, the public domain shall not publish its addresses to private networks*. The purpose for the restrictions is enhancing trust. We illustrate this in Figure 1.

Each domain in Figure 1 has an independent address space and is protected by a trust boundary against inbound traffic.

The solution shall not require any keep-alive signaling from a host. It makes user devices reachable by default due to network capabilities. It is preferable that all those network capabilities reside in regular network nodes rather than in any add-on servers. Naturally, the host needs to keep some level of attachment to the network using a layer 2 protocol. The minimum is that the host can be found using paging like in the case of incoming CS telephone calls. Synchronizing the network attach –process with the sleep-wake-up cycle of the mobile will be a responsibility of the lower layer design.

It is desirable that the solution does not introduce any new protocols for global reachability. It should rather re-use existing protocols possibly with new data types. The solution may define new functions. It is undesirable that the solution requires changes in hosts because there are too many of them. A perfect solution is such that if a stakeholder invests into it, it can immediately benefit irrespective of what the other stakeholders are doing.

On an incoming packet from the global network not belonging to an existing session or flow, the private network shall have reasonable means of making the decision whether the packet is legitimate or not. The private network should be offered a range of tools for helping to make that decision. Collectively, these tools can be called *packet access control* (PAC).

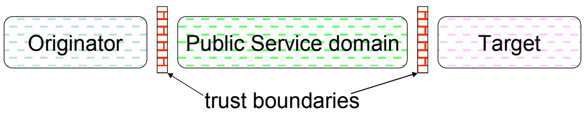


Figure 1: Communication across trust boundaries

We assume that the core network can be based on IPv4, IPv6 or 802.1 and its variants such as 802.1ah.

IV. Switching and Routing

IP routing uses global routing addresses that are carried in each packet. IP routing is a background process that dynamically creates the routing tables in the network nodes. The network state is not tied to user connections which can be short in duration.

In the IP architecture, the idea was that all communicating nodes and devices have a globally unique address. This premise has gradually eroded and user devices often have just a private address. According to the current prediction, the global and regional pools of unallocated IPv4 addresses will be depleted by September 2012 [4]. NATs allow reusing address space by hosts and thus have extended the lifetime of IPv4 considerably.

Switching is a method of data transfer that uses local identifiers and creates and makes use of state per connection, session or flow between users. A session may carry several flows. The mapping between the local identifiers for the incoming and outgoing legs of a connection through a node is stored in the connection state. In circuit switching we do not have any identifiers in the data units that are switched. In packet switching we carry some identifiers in the switched data units. Examples are ATM, label switching, NATs and many types of NAT traversal servers in the Internet. Sometimes, like in the case of NATs, we talk about “translation”. Sometimes, instead of connection state, we talk about a *cache* like in LISP. Here, for clarity we have adopted a different terminology.

The limit of scalability of signaling as a means of setting up connections comes from the size of the signaling flow that is needed for the service as compared to the size of the service payload itself. Routed networks scale best for traffic consisting of frequent short flows. All state is managed by a background process while packet forwarding makes use of the state but does not create any state for the connections or flows between users.

All networks need routing. Routing may use static configuration information, distributed protocols or a centralized path computation system. For the purpose of managing the connection state, switching needs either a management system or signaling. A management system scales to long lasting connections to a limited number of destinations. Signaling scales to connections that are from seconds to hours in duration and can reach Billions of destinations.

Signaling can be out-of-band or in-band. For example, NAT uses *implicit in-band signaling*. Such signaling is embedded in the most typical client server request-response message pattern. When a NAT sees a packet, not belonging to an existing flow, from the private address space, it deduces that a mapping to a global address is required. Having created the mapping entry, it modifies the packet and sets an invisible time limit for the mapping. The value of the time limit is based on static configuration and at best the application can try to spy or guess what the value is. Due to wrong guesses, applications fail sometimes. The advantage of the implicit signaling method adopted in NATs is that it scales well for short flows. This is important for data traffic.

A NAT has to recalculate IP header checksums and TCP/UDP checksums. For better integration with the protocol stack of the host, the timeout for the connection state may be protocol dependent and even fine-tuned for particular protocol states.

A NAT that has no state for a packet coming from the global address space will destroy the packet, because it has no idea to whom to send it in the private address space[1]. The only suitable field in the protocol headers that can identify to whom to send the packet is the TCP/UDP destination port but this overloads the port semantics and is rarely used. A single protocol layer (IPv4) that is supposed to provide global connectivity is not sufficient for dealing with both global and local addresses.

To conclude, NAT is a switching solution that creates a connection state based on implicit signaling and has been overlayed onto a network that was supposed to have no connection state at all. Moreover, usually connection state can be created only when the initiator is in the private address space.

V. Customer Edge Switching

The solution proposes to replace NAT by a new mechanism called customer edge switching. On the address space and trust boundary we replace the NAT box with a device we call a Customer Edge Switch (CES).

A. Identifying users and applications on hosts

To create a basis for protecting the private network from malicious attackers instead of publishing private network addresses in some configuration database such as the DNS, we publish or derive entities that we call *user identity tags (uit)*. These identify users or applications on hosts in a private addressing space. There are several ways of deriving the identity tags. One method is that *Uits* are derived from suitable names such as FQDNs or SIP URIs and suitable other parameters using an algorithm that the private network administrator can choose. The idea is that *uits* are uniformly distributed random values valid for a period determined by the user network administrator. By changing one or several parameters for its domain, the user network operator can

[1] It is possible to assign an IP port to a particular application server on a particular host and create NAT traversal state for an inbound flow on arrival of the initial request message from the public net. This method has limited scalability, creates a configuration burden, is error prone and is rarely used.

invalidate all old *uits* of its users for new connection attempts. We assume that users are roaming and that the names we use to generate their *uits* can not be chosen by the visited network but that they have been chosen by the home network.

We see two alternatives to random identifiers: (1) introducing deterministic user identifiers and (2) assigning globally unique addresses to CES boxes. We argue that deploying a completely new ID schema giving deterministic, scalable and globally unique IDs to all hosts would be burdensome, would increase OPEX and would lead to high protocol overhead per each packet. A clear alternative in option (1) is using 32 bit IDs carried in IPv4 address fields. This may be done in different ways. One of them is embedded in LISP [10]. It has scalability limitations. In an alternative 32-bit ID schema, the IDs would be locally significant and allocated by the visited network. This would have the drawback that an ID would change sometimes during roaming. The second option would mean that the operator community makes an attempt to broker trust among the user community.

Here, we take the position that due to existence of malicious senders it is better to concentrate on supporting all means that a receiver can think of for protecting its network and users. We believe that the benefits from our approach will be increased by combining it with an end to end Carrier Grade transport network.

The first question that arises is how long the random values need to be. We can seek the answer from the birthday paradox (see for example [5]). This is the question of the form: what is the probability that two people in a room have the same birthday if the room has N people. We can turn this into a form: for a given probability of a clash p of two identifiers, how many users we can bring to one CES node provided that the *uit* length is m?

We argue that it is sufficient to require that we can build CES nodes serving a few million users. A CES node is dealing with a set of users that includes its own users and their communicating parties. We assume that each user on average communicates with less than 5 remote parties at any given time. Moreover, it is sufficient to require that the probability of *uit* clash p is less than 1 in one million provided that we can detect most clashes. For comparison, let us recall the individual call failure requirement in telephony networks: ITU-T (Q.542) requires that the probability of premature release of a call is less than 2×10^{-5} at any one minute interval.

Clash detection can be based on *uit* dependent filtering akin to address dependent filtering in NAT. This means that a CES may look at the pair of two *uits* of the communicating parties in a session instead of looking only into one *uit* at a time.

Figure 2 demonstrates the number of *uits* that can come together in a CES as a function of the number of bits to represent the *uit*, provided that $p < 10^{-6}$.

The figure demonstrates that with $m > 60$ we can build reasonable CES systems. Moreover, the figure demonstrates that IPv4 will not be sufficient to carry random *uits* instead of IP addresses. The only important exception that we can think of, is identifying users in a home network behind an xDSL modem or similar. For nodes serving subscribers in public networks, using address dependent filtering for 32 bit addresses does not provide a sufficient safety margin either.

By combining the placeholders for an IPv4 address and a MAC address of the normal 6 octets, we can carry a random *uit* of up to 80 bits that is sufficient for our needs. By defining a new ethertype, we could stretch the DA/SA fields of an Ethernet frame into any desired length.

We assume that private networks use the normal protocol stack of IP over 802.1. These packets are tunneled over the core network that may use routed IPv4, IP/MPLS label switched paths from edge to edge, variants of 802.1 or IPv6 for forwarding. The Customer Edge Switch is responsible for creating the switching state for user flows, modifying the packets using the state and mapping the packets into and from the core forwarding method.

Figures 3 and 4 demonstrate the outbound CES process in the originating customer network and the inbound process in the target customer network.

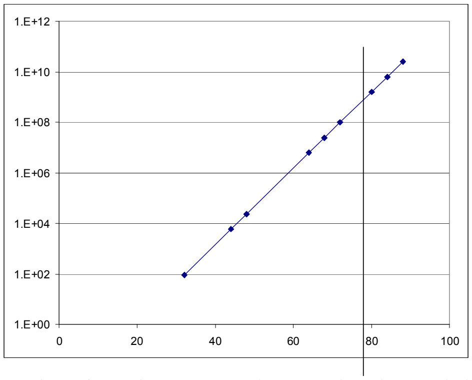


Figure 2: Number of *uits* that can come into a node, when probability of *uit* clash is 1/1000 000 and the length of random *uit* is on the horizontal axis.

The outbound and inbound CES processes make use of some directory service. For example these can be implemented by extending the DNS. The required resources information for a domain reachable through a CES in DNS would include the following:

```
domain | hashing algorithm | constant |
length of hash | cache timeout |other
parameters to ask and from whom| etc...
```

B. *Outbound CES*

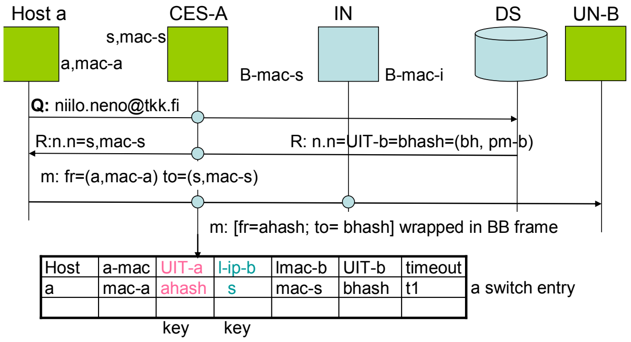
A CES resides in a user network (UN). It can be a consumer or a corporate network or an access network owned by an operator.

A client starts communication with a DNS query. It is routed through the CES box that picks up the response, i.e. the CES contains an enhanced DNS proxy. Figure 3 shows that the response contains target's *uit* split into two parts. If the directory service is implemented using DNS, it would be more natural to respond with the full resource record and let the CES calculate the target's *uit*. Alternatively, the *uit* generation algorithms could be deployed on an ID server that a CES can be configured to use. This would be useful for deploying new generation algorithms. In our solution, it would be natural that the ID server is managed by the home network.

Instead of returning the original response to the host, CES will map the target's *uit* to a local IP-address (s) and local MAC address (mac-s) that it allocates for the communication with the target. CES can use its own mac address as mac-s and allocate the IP addresses to remote parties from the private address space it owns or even from the global IPv4 address space provided that the concerned user network is connected to the global net only through the CES.

The 3rd message in Figure 3 is the initial message to the target. The host sends it to the CES with IP destination address set to *s* because that is where the host believes the target to reside. CES modifies the IP source address and source mac (SA) to carry the two components of the source *uit*. The source *uit* can be generated either on user or service names and ports or allocated temporarily by CES-A. CES sets the IP destination address and destination mac address to carry the target's *uit* that it finds stored in the communication state due to processing the former DNS response. The packet is forwarded to the provider edge node and wrapped in the backbone frame or packet. The outer frame contains the addresses needed for packet forwarding to the egress node.

We have simplified the figure by omitting the details of functions of the Ingress Provider Edge node (IN). It can also use switching to map user traffic to provider tunnels.


Host a
CES-A
IN
DS
UN-B
s,mac-s
a,mac-a
B-mac-s
B-mac-i
Q: niilo.neno@tkk.fi
R:n.n=s,mac-s
R: n.n=UIT-b=bhash=(bh, pm-b)
m: fr=(a,mac-a) to=(s,mac-s)
m: [fr=ahash; to= bhash] wrapped in BB frame
a switch entry
key
key

Host	a-mac	UIT-a	l-ip-b	lmac-b	UIT-b	timeout
a	mac-a	ahash	s	mac-s	bhash	t1

Figure 3: Outbound CES process.

Figure 3 shows the state created in the outbound CES. It stores the private source IP and mac address, the initiator's *uit* (we omit how to generate it), the target's *uit* and local addresses used to represent the remote target. The timeout works in a similar manner as in a NAT.

C. *Inbound CES*

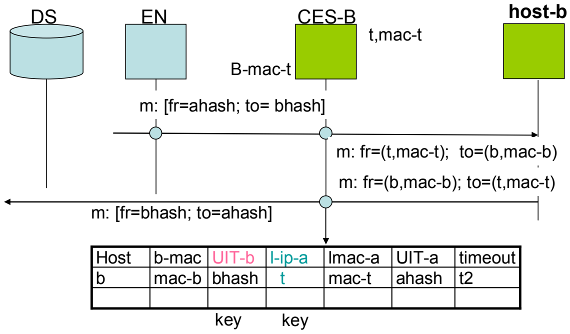
Upon reception of a message to *uit*-b, guided by a policy, the inbound CES (CES-B) creates state. CES-B allocates an IP (*t*) and a mac address (mac-t) to represent the source host from its local pool of addresses. CES-B has a method to map *uit*-b to its IP address. Local routing or bridging will resolve the target's mac address.

Once again, in Figure 4, we have omitted the details of functions of the provider egress node (EN). They can be defined in detail once the exact addressing in the core network is in place.

D. *Protecting a private net from attacks*

An inbound CES can use many methods to establish that an incoming packet is legitimate. These include but are not limited to the following:

(1) Checking that a connection state entry exits.
(2) Checking that the origin network is trusted by the target network.
(3) TCP/IP SYN-packet validation.



Host	b-mac	UIT-b	l-ip-a	lmac-a	UIT-a	timeout
b	mac-b	bhash	t	mac-t	ahash	t2

Figure 4: Inbound CES.

(4) Storing information about the frequency of different types of connection attempts based on any fields in the frame that tries to establish a new connection.
(5) Storing information about connection attempts or e-mail messages that the users have claimed to be unwanted i.e. SPAM.
(6) Generating a cookie in SCTP transport protocol.
(7) Monitoring the transport capacity used by an incoming flow.
(8) Authenticating user *a*.
(9) Challenging the originator with a puzzle that will use computing cycles.
(10) Employing user scanning detection.
(11) Employing firewall rules managed by the user.

Methods (1) and (2) are obvious. Method (3) is implemented in existing advanced L2 switches and works as follows. On reception of the first packet in an incoming TCP session, i.e. the SYN packet, CES-B, instead of forwarding it to user *b*, generates an acknowledgement on the TCP/IP level. If a second packet is sent by the corresponding user *a*, CES-B has established that the user *a* is not faking its address. This makes it more difficult to launch a denial of service attack against *b*. This is a well known counter-method against DOS attacks and here it can be reused in a new context.

Using the stored information and method (4) it is possible to rate limit incoming connection attempts. It is also possible to extend the method to raise the suspicion level of networks UN-A or hosts in UN-A if they seem to generate too many connection attempts. A sufficiently high suspicion level launches administrative action or leads to denying all further connection attempts from the too active client or the network of the client. An additional reactive protocol between Customer Edge Switches could push the deny status closer to the offending correspondent user. If the remote end does not seem to follow such a protocol, CES-B may choose to deny all communication with such an originating network.

Method (5) generates a SPAM filter into CES-B. The SPAM filter will block e-mail that matches the filter. SPAM filtering information is user specific and users may want to keep that information private. This clearly testifies that the functions we are describing here are best placed in a device managed and owned by the users. Likewise, a CES may choose to use any kind of private method or information to achieve its policy goals of securing its local network.

Method (6) makes use of the Stream Control Transport Protocol (SCTP). Instead of passing the first message to user *b*, CES-B takes care of the first phase of association establishment according to SCTP.

Method (7) allows monitoring the capacity used by different flows. This is useful for UDP flows and rogue TCP flows that look like TCP but do not abide by the congestion control algorithm in TCP for the purpose of attacking UN-B or *b*.

Method (8) authenticates user *a* using any authentication method.

Method (9) will push some of the cost of communication to the sender. It may be sufficient to make spamming unprofitable or at least a less attractive business.

Employing user scanning detection (10) detects an originator that tries to send messages to *uits* that do not exist or have been made invalid in the target network. The attacker may have collected *uits* from previous communications but the *uits* have been changed in the target network. The CES may decide not to accept any traffic for a time from a source that has previously tried to scan its users or it may apply other penalties to the connection attempts from that source.

Moreover, the target user may choose managing firewall rules (11) that may, for example, specify which IP ports are allowed and which are denied.

So, far we have avoided deploying new servers. If we take the view that networks can not be trusted, we better publish only the minimal information in DNS. This would be the address of an identity server for the domain managed by the domain administrator. A CES would consult the Id Server to obtain the target's *uit* adding one message pair into the basic message pattern. The Id server would have access to things like the algorithm for *uit* generation and any additional data that is used in the process. The Id Server could also provide *uit* to address and *uit* to name translations for the domain.

E. Summary and Proof-of-concept Prototype

CES hides the user network from the core and the remote user networks by not revealing any private network address information. A CES reuses existing IP and MAC protocol address fields to carry user identity tags that are generated from names such as FQDNs or uniform resource identities in combination with a number of parameters that are stored per resource record or per domain. These parameters improve randomness of the *uit* values and also void the idea of collecting *uits* and reusing them later for attacks. A local CES behaves much like a remote end NAT. A typical application that may even be prepared to traverse a NAT using something like UNSAF, will not notice a CES at all. From the application's point of view, the remote end resides in the same local network as the initiator of the communication. However, applications that send IP addresses on a control channel (e.g. FTP) to the remote party may be broken by the proposed mechanism. A remedy is to send FQDNs instead of addresses.

In a Python prototype, we implemented, ICMP, Telnet, http and SSH traversed a simple CES without difficulty. The prototype demonstrated that applications such as FTP or SIP require that CES must look up the protocol and use a more complex state machine than the one of CES-B that can be drawn up from Figure 4.

We gave a list of examples of methods for establishing that a communication is legitimate. The methods cross protocol layers. The list demonstrates how we can leverage the connection state we propose to have in CES in order to enhance trust.

VI. Deployment

A. Impact on protocol stack

User generated IP packets are carried usually over Ethernet in User Networks. CES or the provider edge tunnels packets over the core protocol. In case of 802.1ah in the core, we have IP over MAC-in-MAC. In this case, the fact that we carry *uit*'s does not need to create any additional protocol overhead per packet as compared to the usual way 802.1ah is used. In case, the core is an IPv4 or IP/MPLS network, CES creates an overhead of about 34 octets per packet. This may cause some issues but for 100Gigabit/s core networks that are soon to emerge this should not be a real problem. If not so, there is a choice of getting rid of IP as a routed protocol in the core.

B. *Ramping up*

The challenge of deploying Customer Edge Switching is that full benefit of the method depends on both ends of the communication deploying it. The problem can be alleviated in at least two ways. First, operators can host CES proxy services for user networks that have not yet deployed their own private CES functions. Due to space limitation we can not elaborate on this in this paper. The second approach is that CES is implemented as an extension to NAT software. Such a CES would behave like a NAT in communication with destinations that do not have CES resource records in DNS and like a CES towards destinations that have a CES DNS resource record.

VII. Comparison with LISP

The Locator/ID Separation Protocol is an experimental solution in the draft state in IETF. Like Customer Edge Switching, LISP [10] proposes to use tunneling for packet delivery over the network core. Similar to our solution, LISP proposes to use Routing Locators as routing addresses in the core network. Unlike CES, LISP is tied to a core that uses IP for forwarding. LISP defines a 32-bit End-point ID (EID) that is in practice carried in an IPv4 address field in the packets created and consumed by hosts. In LISP, both End-point IDs and routing locators need to be globally unique. Considering that only 32 bits are available for the ID and that there are some 1.7B users in the Internet already today this is a serious scalability limitation. LISP uses some explicit signaling to set-up switching state in the ingress and egress nodes. CES uses only implicit signaling for setting up the state targeting to better scalability for short flows.

LISP sacrifices architectural clarity for the sake of trying to maximize compatibility with existing host software. For example End-point identifiers are sometimes used as routable addresses. The current LISP draft does not discuss private IP addressing nor how the protocol helps to protect the destination networks from unwanted traffic. To generalize, LISP does not rely on any particular trust architecture.

LISP sees the locator/ID separation as a matter of scalability and packet delivery. Our approach is driven by the needs of all stakeholders in trust and the needs to tackle unwanted traffic. This need is especially important to mobile hosts because host based firewalls are not appropriate for battery powered devices that use limited bandwidth shared media channels for connectivity. Our approach integrates the issues of locator/ID split and NAT traversal and is optimized for battery powered devices.

VIII. Conclusions

Customer Edge Switching makes nodes in private networks reachable from other private networks using only private addresses and identifiers that do not need to be globally unique. Unlike the recommended solution by IETF namely UNSAF, CES scales to mobile hosts letting them stick to their sleep-wake-up cycle and thus preserve battery power while being reachable by default for any packet applications.

We have implemented a proof-of-concept prototype of customer edge switching [9]. We used it to test the logic and the application interactions. The prototype showed that most applications work fine but that applications transmitting IP addresses on a control channel to a remote end will need support from application specific state machines in the Customer Edge.

The idea of customer edge switching fits nicely together with an Ethernet core network such as being implemented in the ETNA project [7]. The idea can also work, although with the cost of some protocol overhead, with IP core networks. When applied to IP based core, our solution is similar to LISP but relies on a clearer semantics of the Identifiers. An advantage compared to LISP is better reuse of the 32 bit IP address space and as a result, CES scales to a larger number of hosts.

Customer Edge Switching as a concept is independent of the forwarding method that is adopted in the core. For the hosts connected to a user network that is attached to the global net only through CES, the approach changes all IP addresses into private addresses. As a result, the problem of IPv4 address exhaustion goes away. Private addresses can be reused as many times as one wants. Consequently, CES removes the need to deploy IPv6. Also, this is unlike LISP.

Customer Edge Switching makes connection state an important and legitimate building block of the end-to-end architecture. Lawrence G. Roberts who designed the first routers, is advocating the idea that a router should switch flows in addition to performing the routing functions for the first packet of the flow [8]. CES goes further by isolating networks on the two sides of a trust boundary. Roberts' work shows how traffic can be better managed based on flow state.

CES can be seen as an implementation of the idea of Trust-to-Trust stated by Dave Clark [6]. CES puts connection state on trust boundaries. Based on the state, elaborate methods of packet access control for tackling DOS and DDOS attacks and other malicious connection attempts can be deployed.

One might object to the idea of connection state for all communication or to the fact that application specific state machines are needed. However, in this respect CES only does the same as NATs and Firewalls. Unlike, NATs and Firewalls, CES makes connection state a legitimate and essential part of the Internet architecture.

A. Future work

We plan to implement a comprehensive prototype of customer edge switching and explore its use in context of different core networks and intermediate steps of deployment. Also, the best and most secure ways to translate names to *uits* and *uits* to addresses as well as interactions with all kinds of applications need to be tested.

Acknowledgment

This work was partly supported by the EU funded, FP7 strep project ETNA Ethernet Transport Networks, Architectures of Networking (project nr 215462) [7].

References

[1] Behavior Engineering for Hindrance Avoidance (behave), http://www.ietf.org/html.charters/behave-charter.html, referred June 30th, 2009.

[2] Miyazaki, Yutaka & Suzuki, Hidekazu & Watanabe Akira. 2007. A Proposal for a NAT Traversal System that Does Not Require Additional Functions at Terminals. IEEE TENCON 2007. Graduate School of Science and Technology, Meijo University.

[3] Miyazaki, Yutaka & Suzuki, Hidekazu & Watanabe Akira. 2008. Proposal of a NAT Traversal System Independent of User Terminals and Its Implementation. IEEE TENCON 2008. Graduate School of Science and Technology, Meijo University. ISBN 978-1-4244-2408-5.

[4] http://www.potaroo.net/tools/ipv4/index.html

[5] http://en.wikipedia.org/wiki/Birthday_problem, referred June 30th, 2009.

[6] David Clark, MIT Communications Futures Program, Bi-annual meeting, May 30-31, 2007, Philadelphia, PA.

[7] http://www.ict-etna.eu/index.html.

[8] Lawrence G. Roberts, A radical new router, IEEE Spectrum, July 2009.

[9] L. Virtanen, Communicating Globally Using Private IP Addresses, M.Sc thesis, Comnet/TKK, 2009.

[10] D. Farinacci, V. Fuller, et.al, Locator/ID Separation Protocol (LISP), draft-ietf-lisp-05.txt, Sep 29, 2009.