Scalable VPNs for the Global Information Grid

Sam Small, Andreas Terzis, Fabian Monrose
Computer Science Department
Johns Hopkins University
Email: {sam,terzis,fabian}@cs.jhu.edu

Bharat Doshi, Antonio De Simone
Applied Physics Laboratory
Johns Hopkins University
Email: {bharat.doshi,antonio.desimone}@jhuapl.edu

***Abstract*— Virtual Private Networks (VPNs) are the preferred mechanism for securing sensitive traffic crossing public networks. Traditionally, configuration of VPN gateways has been done manually. However, static configuration of gateways is particularly problematic within the context of the *Global Information Grid* (GIG), the next-generation network of networks developed by the US government. For one, GIG VPNs are expected to consist of tens to hundreds of trusted networks, which is an order of magnitude greater than current deployments. Moreover, trusted networks that essentially comprise of units in the field (e.g. army companies or ships) need to be seamlessly connected to the GIG even while they are *mobile*. Our goal in this paper is to address the lack of scalability and support for mobility that exists in current VPNs. We do so by providing a dynamic routing protocol which VPN gateways use to securely advertise prefixes of their internal network to peering gateways. Our initial results show that our protocol can sustain VPNs comprising of roughly one thousand gateways with only moderate overhead.**

I. Introduction

Virtual Private Networks [19] (VPNs) are the preferred mechanism to provide security for sensitive network traffic crossing networks such as the Internet. A VPN connects geographically remote networks through a series of virtual links (also known as *tunnels*) between peering VPN gateways. These VPN gateways lie at the border of trusted networks and the public network and encapsulate, encrypt, and authenticate data on behalf of internal hosts. Today, such tunnels are manually configured when the virtual network is established.

VPNs play a central role in the Global Information Grid [21] (GIG), the combined network-of-networks being developed by the US government to support the communication needs of the defense and intelligence communities. The GIG architecture can be viewed as having two main components, namely trusted edge networks and a large backbone core consisting of a combination of both trusted and untrusted network segments. In order to achieve privacy and integrity of the data crossing the backbone, edge networks use VPN gateways to encrypt outbound traffic.

The scale of the GIG, in size and global reach as well as in the diversity and services to be supported, is approached by only the very largest commercial enterprise networks. But even the largest enterprises do not face the special problems the GIG must solve: by the nature of network-centric warfare, the GIG will include a large number of wireless and mobile nodes as well as mobile networks on ships, aircrafts and mobile ground forces. While backbone links will be high-capacity and nearly error free, the links at the tactical edge will be bandwidth-limited and will suffer from errors and outages that will make them intermittent over a broad range of timescales. The combined effects of the sheer scale and the dynamic behavior of the VPNs envisioned for the GIG motivate our work.

In this paper we propose to address the aforementioned issues by replacing the static configuration of VPN gateways with a dynamic routing protocol. Specifically, each gateway in the VPN automatically discovers the list of its peers and establishes secure links with each of them. Through these tunnels, the gateway advertises the list of internal network prefixes that it is able to reach, and peering gateways update their configuration tables accordingly. Subsequent packets destined to the internal networks are routed through the VPN gateway that advertised those networks. In addition to removing the configuration burden, our routing protocol naturally supports network mobility. A moving gateway can simply re-advertise its internal networks from its current attachment point to the backbone.

Paper Organization: The rest of the paper is organized as follows. In Section II we describe the GIG and how it motivates and influences our system model. A high-level view of our proposal is given in Section III, followed by a more specific description in Section IV. We evaluate our design in Section V. Related work is presented in Section VI, and we conclude with closing remarks in Section VII.

II. Background

The Global Information Grid (GIG) is the network-of-networks being developed by the US government to support the communications and information systems needs of the defense and intelligence communities. Key network domains to be deployed under the GIG umbrella include a

very high capacity IP-Optical terrestrial backbone called the GIG-BE (GIG Bandwidth Expansion), high capacity satellite backbone called Transformational Satellites (TSAT), and ground, sea, and air based wireless tactical networks using software defined radios (Joint Tactical Radio Systems or JTRS). These networks, coupled with legacy satellite, wireline terrestrial, and wireless land, sea, and air-based networks will provide the GIG network infrastructure. While all component networks of the GIG will be built using standards based IP technology, it also requires adapting, and possibly, extending existing IP technology.

A fundamental goal of the GIG is to support a broad spectrum of users, and so it will carry information that spans the full spectrum of security levels, from fully public information up to Top Secret. The architectural construct that supports this capability is the GIG *black core*. The term, black core, refers to any common network segment in a delivery path that has been determined unfit for transporting unencrypted traffic generated by edge networks (we refer to these edge networks as *red networks* or *enclaves*). Outbound plaintext traffic is encrypted at the edge of red networks by IPsec VPNs so that only encrypted (hence unclassified) information is carried across the black core (see Figure 1). We envision that the edge will move closer to the hosts and hence the number of red networks will grow from a few hundreds up to tens of thousands and even to a million or more. Moreover, it is likely that some of these networks will have multiple connections to the black core, for example using parallel terrestrial and satellite links. The black core will include all of the JTRS domains, GIG-BE and current terrestrial wireline domains, TSAT and many current satellite domains, and will act like the public Internet infrastructure. Red enclaves will be small campuses, single building, single LAN, or even single hosts that connect to the black core via gateways.

The black core architecture means that VPNs are a central part of the GIG: any communication across the black core requires a security association so the information may be encrypted and any entity communicating across the GIG must be behind a VPN gateway. Notice that for each pair of red enclaves that wish to communicate a VPN tunnel must be established between them. This requires that the IPsec VPN gateway at the source enclave know the black core address of the destination enclave (in general, each enclave can be represented by an address prefix). To make all enclaves available to each other, each IPsec VPN gateway needs to be configured with a mapping between black core addresses and enclave prefixes in addition to maintaining key parameters for each enclave.

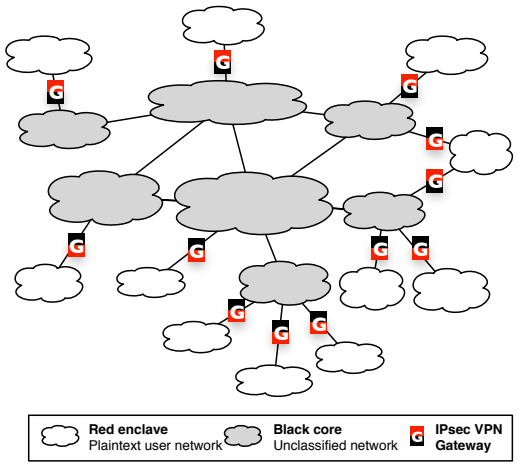

Fig. 1. An outline of the GIG network infrastructure. The black core consists of a number of common network segments that carry encrypted traffic generated by IPsec VPN gateways located at the edge of red enclaves.

III. System Overview

In this section, we provide a high-level overview of the VPN routing protocol, as well as a brief description of its different components. We provide a more in-depth discussion on each of these components in Section IV.

As an illustrative example, consider the topology given in Figure 2, which contains a black core with a number of enclaves, R_1 through R_4, each assigned a set of private address prefixes P_i, and having one or more VPN gateways denoted as V_i. Now suppose that an entity in R_1 needs to communicate with another in R_3. Traditionally, this requires that R_1's VPN gateway, V_1, be statically configured with an association that states that traffic destined for a P_3 address is tunneled via V_3, the VPN gateway at R_3.

We address the issue of manual configuration of enclave routes by having a VPN address prefix routing protocol run among all VPN gateways connected to the black core. In this way, each VPN gateway advertises its private prefix set to its peers, and prefix routing tables are generated and maintained automatically. This automatic configuration is achieved by having each enclave first discover (i.e, when the gateway first connects to the black core) its peers through a well-known network *rendezvous point*, and then securely trade prefix information with each peer. The exchanged information is then used to maintain a dynamic routing table similar to what is done in traditional multi-homed networks.

IV. System Details

A. Peer Gateway Discovery

Clearly, before a VPN gateway can advertise the list of its internal prefixes it has to first discover its peers.

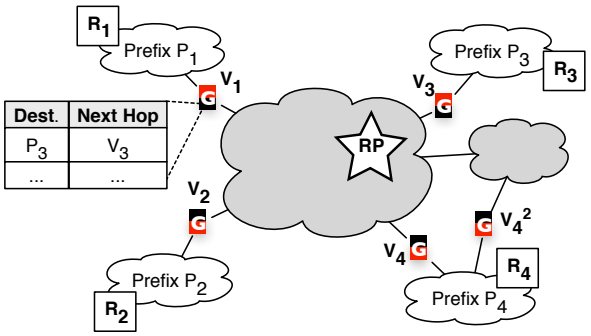

Fig. 2. A high-level model of our system. Each enclave R_i has an address prefix set P_i and connects to the black core through one or more VPN gateways V_i. A black-core addressable *rendezvous point* (RP) provides peer discovery to each enclave. After discovery, peers advertise prefix sets to each other and insert entries into their prefix routing tables.

This functionality is achieved by using a straightforward scheme for gateway discovery whereby gateways register themselves with a well known rendezvous point (RP) as they come online. Upon successful registration (see Section IV-A.1) each gateway receive a list of the other gateways attached to the core.

Notice that the rendezvous point serves as a level of indirection. This arrangement has numerous benefits; for one, by not directly disclosing the addresses of the VPN gateways, the gateways are not bound to a specific network location and may even choose to be only periodically available. An up-to-date state of the network topology is maintained by having gateways periodically pass soft-state updates (i.e., keep-alive messages) to the rendezvous point. Moreover, to accommodate variations in network latency and capacity, the period length is specified by each VPN gateway when it first contacts the RP. If the rendezvous point has not received an update from a particular gateway for more than δ consecutive periods, then that gateway is dropped from the RP's state table.

1) Gateway Authorization: As security and scalability are high priority goals in the GIG architecture, the rendezvous point must be robust to attack or failure and should not leak any information about tunneled traffic. Therefore, to ensure that only authorized parties gain access to the network, the rendezvous point only responds to authenticated requests [1]. To achieve flexible and transparent authentication between gateways and the RP, we use a Kerberos-like [10], [20] protocol.

Due to the potential scale of the GIG, protocol performance is a genuine concern, and so our authentication protocol uses hybrid cryptography to keep computational overhead low. Public-key operations are used with pre-deployed certificates signed by a certification authority to establish mutual trust with the rendezvous point [2]. This essentially "bootstraps" system-wide authentication for a VPN gateway. Subsequently, symmetric-key operations are used to secure the majority of any remaining protocol messages. The notation for message interaction is given in Table I, followed by the protocol itself in Table II.

In our scheme, the traditional services provided in Kerberos by the key-distribution center (KDC) and ticket-granting service (TGS) are provided by the rendezvous point. However, unlike the standard protocol, messages (1) and (2) occur over a private and mutually authenticated channel (e.g. SSL [5]). Privacy and trust are relevant in these initial steps to prevent the rendezvous point from disclosing the set of VPN gateways currently online, represented in our protocol by the set $\mathcal{R}$.

One noteworthy property of message (2) is that the rendezvous point can choose to enforce some form of access control by revealing only a subset of $\mathcal{R}$ to certain VPN gateways (likewise, a VPN gateway may choose to initiate prefix advertisement with only a subset of the VPN gateways disclosed in $\mathcal{R}$). This property is especially useful in the context of the GIG because it allows the rendezvous point to logically separate enclaves based upon traffic disclosure levels (e.g. maintain separate rosters for enclaves that transmit unclassified, secret, or top secret information) and additionally allows enclaves control over their degree of connectivity if constrained by network resources.

After a new VPN gateway has received $\mathcal{R}$ in message 2, it can request authentication tickets for any member of the set. Consider the case where a VPN gateway wants to establish communication with all members. For each VPN peer $r_i \in \mathcal{R}$, the new VPN gateway contacts the rendezvous point and requests communication credentials (i.e. tickets in Kerberos parlance). These credentials are passed on to the peer described by r_i and verified (messages 4 and 5). From this point forward, the two VPN gateways trust the authenticity of each other and can begin route advertisement (message 6). These steps are repeated for all $r_i \in \mathcal{R}$ (messages 3 through 6).

At this point, the new client and its peer VPN gateways share pairwise session keys, $K_{c,s}$, which are used to encrypt traffic (e.g, prefix information) transmitted between nodes (it may be possible to use a simplified IKE parameter negotiation protocol at this point). We show in Section V that since the gateway authentication protocol is mostly based on symmetric key cryptography, its performance

[1]For availability and load-balancing, multiple redundant and synchronized RPs may be used.

[2]It is assumed that a public-key infrastructure is preexistent in the GIG. Issues regarding the availability and resiliency of certification authorities are tangent to the topics covered here.

Msg. #	Principals	Message	Description
1	$c \rightarrow RP$	$c =$ "$vpn@node_0$"; $s_0 =$ "$rp@node_1$"; n_0	*Request roster from RP*
2	$RP \rightarrow c$	$n_0; \mathcal{R}; K_{c,RP}; T_{c,RP} = \{c, s_0, a, t, D_{srvc}, K_{c,RP}\}_{K_{RP}}$	*Return roster,* $\mathcal{R}$
The following message are repeated for each selected $r_i \in \mathcal{R}$.			
3	$c \rightarrow RP$	$s_1 =$ "$peer@node_2$"; $n_1; A_{RP} = \{c, a, t, r_i\}_{K_{c,RP}}; T_{c,RP}$	*Request ticket for* r_i
4	$RP \rightarrow c$	$\{n_1; K_{c,s}; T_{c,s} = \{c, s_1, a, t, D_{srvc}, K_{c,s}\}_{K_s}\}_{K_{c,RP}}$	*Ticket granted for* r_i
5	$c \rightarrow s$	$A_s = \{c, a, t\}_{K_{c,s}}; T_{c,s}$	*Contact* r_i *with authenticator*
6	$s \rightarrow c$	*Secure interaction*	*Begin prefix advertisement*

TABLE II

STEPS REQUIRED TO AUTHENTICATE TO A VPN GATEWAY AND ESTABLISH A VPN TUNNEL. MESSAGES (1) AND (2) OCCUR OVER A MUTUALLY AUTHENTICATED SSL CHANNEL. MESSAGES (3) THROUGH (6) ARE REPEATED FOR EACH r_i.

Abbrv.	Meaning
$\{m\}_K$	Encryption of the message m under the key K.
K_p	Principal p's private key
$K_{p,p'}$	A session key shared between p and p'.
RP	Identifier for the rendezvous point
c	Client identifier (e.g. new VPN gw), of the format "client@node"
s	Service identifier, of the format "service@node"
$\mathcal{R}$	Set of VPN gateway records $r_1 \ldots r_n$
n_i	Randomly generated nonce
$T_{a,b}$	Ticket generated for service between principals a and b.
A_{srvc}	Authenticator generated by a client to match T_{srvc}.
D_{srvc}	Duration of a ticket for the service $srvc$.
a	Identifier for c (eg. IP address).
t	The current time.

TABLE I

INDEX OF SYMBOLS USED TO DESCRIBE TUNNEL ESTABLISHMENT PROTOCOL IN TABLE II.

overhead remains negligible.

B. Route Announcements

Once the tunnels among peering gateways are established, each gateway participates in an inter-VPN routing protocol based on the Routing Information Protocol, RIP v2 [13]. At each gateway, a routing table is constructed that contains entries indexed by destination with next-hop and cost pairs. In our case, destinations are internal enclave prefixes, next-hops contain the black core address of VPN gateways, and costs are used to signify preferred links on multi-homed VPN routers.

The protocol then works in the following manner. Each VPN gateway sends the list of the address prefixes in its local private (red) network to its peers. A gateway receiving an advertisement from peer R_j for prefixes $p_1, p_2, \ldots p_n$, updates its *correspondence table*, mapping $p_i, i \in [1, \ldots, n]$ to the IP address of R_j. At that point, packets destined to p_i are encapsulated at the local gateway with the IP address of R_j and forwarded through the black core. When the encapsulated packet finally arrives at R_j it is decapsulated and forwarded to its final destination. We assume here that each VPN gateway learns the list of internal prefixes through the intra-domain protocol (e.g. OSPF [14], IS-IS [4], or RIP [18]) running inside the local red network.

We note that the inter-VPN routing protocol we use differs from RIP in a significant way; in RIP, each gateway periodically resends its routing table to its neighbors. Instead, we only send keep-alive messages unless changes occur to an internal prefix list. In our case, the routing table need not be sent (periodically) as the protocol takes place over a reliable TCP channel.

C. Network Mobility and Multihoming

While *mobility support* has traditionally been used to convey the idea of provisioning for uninterrupted network access to individual end-hosts moving between networks, mobility in the GIG setting is very different — in our case, entire *networks* are mobile (e.g. a navy ship). Thus, when an enclave, along with its VPN gateway, moves to a new location the enclave's VPN gateway re-registers with the RP and re-announces its prefixes to the other VPN gateways, thereby restoring connectivity to the core. If the mobile network has multiple links to the GIG (e.g. one terrestrial and one satellite link) one of which remains connected while the network is moving, the dynamic routing protocol shifts traffic to the secondary link during the move. This way the service interruption is minimal during the periods when the primary network link is disconnected. This same technique can be used to support automatic fail-over to backup links if a stationary multi-homed VPN gateway loses connectivity through its primary link.

As mentioned briefly in Section IV-B, we achieve fail-over, multihoming, and mobility by using a cost metric in prefix advertisements as part of the inter-VPN routing protocol. First, a multi-homed VPN gateway registers all

external interface addresses with the RP. Subsequently, the multi-homed VPN gateway advertises the private prefixes through both interfaces but with different link metrics. A number of different metrics can be used such as the link's delay or the link's bandwidth. Peer gateways that receive both advertisements choose the path with the lowest metric. If a mobile enclave is not multi-homed, the VPN gateway will request that all peers remove its routes from their tables and unregister with the VPN. When this enclave reconnects to the core, it re-registers and advertises prefix information from its new black-core address.

Note, however, that connecting to the GIG is not as easy as simply "plugging in" to the core; un-authenticated attachment would be problematic given that sensitive data traverses the core. Therefore, to safely accommodate mobile enclaves, we introduce an additional entity to the network called a *docking gateway*. Docking gateways are special edge routers that control access to the network for mobile enclaves. Mobile enclaves connect to available edge networks after relocating and discover docking gateways using a DHCP-like mechanism.

Enclaves and docking gateways mutually authenticate using the Needham Schroeder Public Key Protocol [12]. Unlike the authentication protocol described in Table II, this public-key protocol is meant only to provide authentication between docking gateways and encalves, and uses only public-keys and certificates previously generated offline by the certification authority.

After authentication takes place, the docking gateway provides, to the enclave, network access to the GIG at-large. If mutual authentication fails, the docking gateway simply drops all enclave-generated (gateway) packets.

V. Evaluation

We evaluated our VPN gateway protocol both analytically and with a custom network simulator. Specifically, we consider three primary cost components: network setup, maintenance, and route updates (e.g. enclave relocation). To better evaluate our design, we created a prototype implementation of our VPN system using a network simulator, Simnet [8]. Developed at Johns Hopkins University, Simnet is a freely-available discrete-event network simulator designed specifically for analyzing network security protocols. We chose to use Simnet in lieu of other available network simulators due to its modular and extensible design that allows for fast prototyping of intricate protocols and run-time topology control.

The design of our simulations are intended, at a high-level, to mirror the network tiers of the forthcoming GIG topology (for reference, see Figure 1).

A. Network Setup

Recall that bringing a VPN gateway online involves contacting and authenticating with the rendezvous point, and then advertising its enclave's address prefixes to other VPN gateways attached to the core. This means that the number of messages required to bring a VPN gateway online is order $O(n)$ where n is the number of VPN gateways already online and participating in route advertisement. Similarly, the authentication protocol used for guaranteeing privacy and authentication requires order $O(n)$ symmetric-key operations to bring a VPN gateway online.

Compared to a public-key based solution, our symmetric-key authentication protocol scales well. In Table III, we present the computational cost for the last VPN gateway to come online in various sized networks and authenticate with the rendezvous point and all VPN gateway peers [3]. This cost does not reflect factors such as network latency. The results show that our authentication protocol continues to perform well with negligible cost as the number of VPN gateways in the network increases, while mutual authentication with asymmetric-key operations results in palpable delay at even 256 gateways. The primitives used for these experiments are HMAC-SHA1 [16] and AES-CBC [17](256-bit keys) for symmetric-key operations and DSA [15] for asymmetric signing and verification (1024-bit keys) [11], [2].

# VPN GWs	Symmetric	Asymmetric
64	7.6e^{-5} *sec*	1.8 *sec*
256	3.0e^{-4} *sec*	7.3 *sec*
1,024	1.2e^{-3} *sec*	29.1 *sec*
8,192	9.7e^{-3} *sec*	3.8 *min*
16,384	1.9e^{-2} *sec*	7.7 *min*
32,768	3.8e^{-2} *sec*	15.5 *min*
65,536	7.8e^{-2} *sec*	31 *min*

TABLE III
A COMPUTATIONAL TIME COMPARISON BETWEEN OUR SYMMETRIC-KEY BASED AUTHENTICATION PROTOCOL AND PUBLIC-KEY AUTHENTICATION IN VARIOUS SIZED NETWORKS. THE TIMES GIVEN REPRESENT THE COMPUTATIONAL TIME NEEDED FOR THE LAST GATEWAY IN NETWORKS OF VARIOUS SIZES TO PERFORM ALL CRYPTOGRAPHIC OPERATIONS REQUIRED FOR AUTHENTICATION.

B. Maintenance

After the VPN gateways come online and advertise prefix lists to each other, soft-state keep-alive messages are used

[3]These values are derived from benchmarks taken using OpenSSL [1] on a PowerPC G4 running at 1.33 GHz with 768 MB of RAM. Specialized hardware-based VPN gateway will perform cryptographic operations quicker than the values presented here, though the trends will be the same.

for two reasons: to keep rendezvous point membership accurate and up-to-date, and to purge stale VPN peer associations. The overhead to maintain a quiescent network (i.e. no new prefixes, gateways, or disruptions) is dependent only on the gateways' soft state periods.

Although the messages themselves are small, the effects of different timer periods upon network utilization can be quite significant depending on the size of the VPN topology. The reader may recall from Section IV that update periods can be negotiated between pairs of VPNs and the rendezvous point depending on network characteristics; for our experiments update periods are set uniformly at each gateway.

Period	No. Updates	Max BW (Bytes/sec)
30 *sec*	35,940	450
60 *sec*	17,984	225
120 *sec*	8,990	113
180 *sec*	5,992	75
240 *sec*	4,492	57
360 *sec*	2,996	38

TABLE IV

SIMULATION RESULTS FOR THE NUMBER OF SOFT-STATE UPDATES SENT DURING A ONE HOUR PERIOD WITH DISTINCT UPDATE PERIODS IN A NETWORK WITH 24 VPN GATEWAYS.

Table IV shows the number of network messages caused by soft-state updates in a simulated network with 24 VPN gateways for a duration of one hour at 30, 60, 120, 180, and 360 second intervals. The simulation values vary slightly (average delta of -17 seconds) from the expected value because VPN routers come online in tandem rather than simultaneously in our simulation. Maximum bandwidth reflects the aggregate network overhead caused by soft-state updates in bytes per second. Observe that even at a relatively fast update interval of 30 seconds, the total network overhead is only 450 Bytes /s to maintain VPN connectivity in the network.

Given the above results we can calculate the expected number of network messages for an arbitrary number of VPN routers in the topology. As shown in Figure 3, the network burden is significantly different for topologies of larger sizes with different update periods. Naturally, the choice of update periods reflects a tradeoff in the increase of network traffic versus reaction time to network changes. Even so, in a network with 1000 VPN gateways our results indicate that uniform timers with periods ranging from 30 to 360 seconds yield aggregate network overhead ranging from 6 Mb/s to 0.5 Mb/s; given the available bandwidth in the core, any of these periods can be considered an appropriate choice.

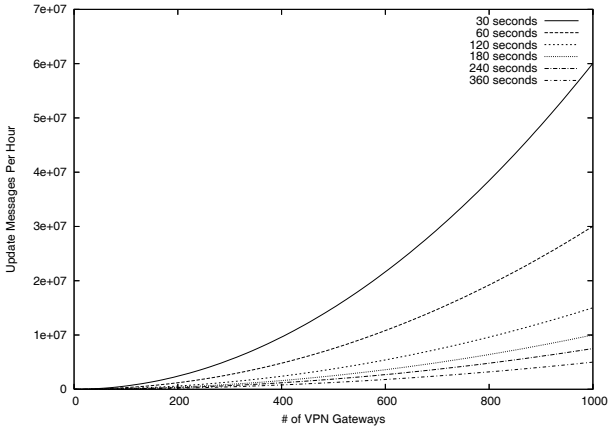

Fig. 3. This figure depicts the total number of soft-state updates traversing the network in a one hour period for different network sizes at distinct timer periods.

C. Route Updates

The last component we evaluate is the overhead incurred by prefix route updates. The updates can be triggered by a number of events such as having an enclave move, change its preferred route, or modify its prefix set. Modifying a prefix set or changing a preferred route requires that an enclave's VPN gateway generate an update message for each peer gateway and therefore produces order $O(n)$ messages. On the other hand, mobility of an enclave with respect to its VPN gateway, requires route updates both before and after relocation, resulting in twice as many messages. In addition, gateway relocation incurs a small additional cost in order to authenticate with a docking gateway (this cost is negligible, consisting of only three public-key verifications).

VI. RELATED WORK

Baker *et al.* propose a method for translating private network addresses to addresses advertised by gateways in the public network [3]. The proposed method uses a hash function to translate the host portion of a private IPv6 address to another 64-bit number advertised to the public network by the VPN gateway of that private network. Peering VPN gateways receiving packets for that private address calculate the same one-way hash function to map the private address to the corresponding public address advertised by the original gateway. Our solution applies to IPv4 and IPv6 networks alike. Furthermore, rather than advertising additional routes on the public network, VPN gateways exchange routing information over secure channels, leaking no information about the private networks to the black core.

Two IETF working groups have established charters in areas relevant to our work as well. The Network Mobility Working Group [9] (NEMO) is exploring methods to manage the mobility of an entire network in the Internet. The mobile networks of interest to that working group are edge networks only, much like the enclaves we discuss, and do not carry transit traffic. In general, they handle mobile networks in a fashion analogous to mobile hosts in mobile IP [7].

On the other hand, the IKEv2 Mobility and Multihoming Working Group [6] (Mobike) is focused on specifying extensions to the IKEv2 protocol to add support for multihoming and mobility. The goals of this working group are similar to ours and members of Mobike are planning to publish a first draft in June 2005. However, peer discovery is a topic that is not addressed by Mobike thus far.

Other approaches to the problem have been proposed. One set of approaches involves, as in the approach discussed in this paper, each VPN gateway maintaining the list of prefixes it is fronting for. However, instead of VPN gateways exchanging plaintext routing reach-ability and information, they keep the information local and use multicast queries to find a peer VPN gateway every time a VPN gateway needs to identify a destination VPN gateway for a destination prefix. Other approaches are based on using servers to store prefix to VPN gateway mappings. VPN gateways query these servers to obtain the mappings as needed and update these servers whenever there is a change in local prefix configurations.

VII. Summary

Virtual Private Networks are a central part of the design for the Global Information Grid. In this paper we have proposed a system to automate VPN configuration and maintenance for large-scale VPN deployments. We introduce a peer discovery mechanism and routing protocol that advertises prefixes between peers to remove static configuration and reduce maintenance costs. Additionally, we announce prefixes over multiple links so that peers choose a route with the lowest metric to support multi-homed and mobile enclaves. Our initial results are encouraging, indicating that our protocol can scale reasonably well in the GIG to over a thousand VPN gateways. In future work we will explore methods to further reduce protocol overhead and improve traffic engineering and control.

References

[1] The OpenSSL Project. http://www.openssl.org/.

[2] J. Arkko, V. Devarapalli, and F. Dupont. Using IPsec to Protect Mobile IPv6 Signaling Between Mobile Nodes and Home Agents. *RFC 3766*, June 2004.

[3] F. Baker, P. Bose, and D. Voce. Routing in a Nested VPN. Internet Draft, draft-baker-nested-vpn-routing-00.txt, Feb. 2005.

[4] R. Callon. Use of OSI IS-IS for Routing in TCP/IP and Dual Environments. *RFC 1195*, Dec. 1990.

[5] T. Dierks and C. Allen. The TLS Protocol. *RFC 2246*, Jan. 1999.

[6] P. Hoffman and J. Arrko. IKEv2 Mobility and Multihoming Working Group, Internet Engineering Task Force. http://www.ietf.org/html.charters/mobike-charter.html, 2004.

[7] J. Ioannidis, D. Duchamp, and G. Q. M. Jr. IP-Based Protocols for Mobile Internetworking. In *Proceedings of ACM SIGCOMM*, pages 235–245, 1991.

[8] S. Kamara, D. Davis, L. Ballard, R. Caudy, and F. Monrose. An extensible platform for evaluating security protocols. In *38th IEEE Annual Simulation Symposium*, pages 204–213, San Diego, CA, April 2005.

[9] T. Kniveton and T. Ernst. Network Mobility Working Group, Internet Engineering Task Force. http://www.ietf.org/html.charters/nemo-charter.html, 2004.

[10] J. Kohl and C. Neuman. RFC 1510: The Kerberos Network Authentication Service (V5), September 1993.

[11] A. K. Lenstra and E. R. Verheul. Selecting cryptographic key sizes. *Journal of Cryptology: the Journal of the International Association for Cryptologic Research*, 14(4):255–293, 2001.

[12] G. Lowe. Breaking and fixing the Needham-Schroeder public-key protocol using FDR. In *Tools and Algorithms for the Construction and Analysis of Systems (TACAS)*, volume 1055, pages 147–166. Springer-Verlag, Berlin Germany, 1996.

[13] G. Malkin. Routing Information Protocol Version 2. RFC 2453, SRI Network Information Center, November 1998.

[14] J. Moy. OSPF Version 2. *RFC 2328*, Apr. 1998.

[15] National Institute of Standards and Technology. Digital Signature Standard. *FIPS 186*, May 1994.

[16] National Institute of Standards and Technology. Secure Hash Standard. *FIPS 180*, Apr. 1995.

[17] National Institute of Standards and Technology. Advanced Encryption Standard. *FIPS 197*, Nov. 2001.

[18] D. Pei, D. Massey, and L. Zhang. Formal Specificaiton of RIP Protocol. Technical report, UCLA CSD, Feburary 2003.

[19] C. Scott, P. Wolfe, and M. Erwin. *Virtual Private Networks, 2nd Edition*. O'Reilly, Sebastopol, CA, 1998.

[20] J. Steiner, C. Neuman, and J. Schiller. Kerberos: An Authentication Service for Open Network Systems. In *USENIX Association Winter Conference 1988 Proceedings*, pages 191–202, February 1988.

[21] United States Department of Defense. About GIG enterprise services. http://ges.dod.mil/about.html.