# Behavioral and Performance Characteristics of IPsec/IKE in Large-Scale VPNs

Okhee Kim
Advanced Network Technologies Division
Information Technology Laboratory
National Institute of Standards and Technology
Gaithersburg, Md 20899, USA
email: okim@nist.gov

Doug Montgomery
Advanced Network Technologies Division
Information Technology Laboratory
National Institute of Standards and Technology
Gaithersburg, Md 20899, USA
email: dougm@nist.gov

**ABSTRACT**

Cryptographic network security services are essential for providing secure data communication over an insecure public network such as the Internet. Recently there has been tremendous growth in the requirements for, and use of, secure *virtual private networks (VPNs)* to interconnect enterprises with business partners, traveling staff, and remote office locations.

*Internet Protocol Security (IPsec)* tunnels have become one of the most widely adopted means to build secure VPNs between sites and individual computers. To date, most IPsec VPNs are statically configured and are of moderate scale. To facilitate future large-scale VPNs with potentially rapidly changing memberships and varying security policies the industry must move to the use of dynamic protocols for the establishment and managment of cryptographic keys and security policies. Adopting such on line management systems will ease the administrative burden associated with VPN instantiation and operation.

In this paper we examine the relative performance characteristics and dynamic behavior of large scale VPN environments based upon IPsec and the *Internet Key Exchange protocol (IKE)* version 1 [1]. We introduce the *NIST IPsec/IKE Simulation Tool (NIIST)* and use its detailed, packet level, simulation models to characterize the performance impact of varying: key management scenarios, security association (SA) policies and management parameters, cryptographic algorithms, and implementation options in IPsec/IKE suites. Our results highlight the significant performance impact of subtle IPsec/IKE implementation and policy decisions on the overall performance and behavior of TCP based applications in large scale VPNs.

**KEY WORDS**

Network Security, IP Security Protocols, Internet Key Exchange and Management, Virtual Private Networks, Performance Analysis, Large-scale Networks.

[1] version 2 of IKE is currently under development by IETF

## 1 Introduction

The *Internet Protocol Security (IPSec) suite* [1, 2, 3] was designed in the IETF to provide network security services such as confidentiality, data origin authentication, data integrity, and anti-replay to protect datagrams in the Internet. To enable future growth and manageability, VPNs are evolving from the use of static IPsec tunnels and are begining to adopt dynamic key and policy management systems.

To date, little focus has been given to thoroughly understanding the detailed behavior and performance impact of the interacting IPsec/IKE suite of protocols in very large scale VPNs. Characterizing the performance dynamics of such environments can provide users valuable insight in the design and management of scalable IPsec VPNs.

To address these issues we have developed the *NIST IPsec/IKE Simulation Tool (NIIST)* [4], a detailed, packet level, simulation model built as an extension to the *Scalable Simulation Framework (SSF)* [8]. We have used NIIST to characterize the performance impact and detailed behavior of various IPsec VPN scenarios, including: various key management policies, security policy and management parameters, choice of cryptographic algorithms, and implementation options in IPsec/IKE suites. Our results highlight the significant performance impact of subtle IPsec/IKE implementation and policy decisions on the overall performance and behavior of TCP based applications in large scale VPNs.

In this paper, we present simulation results to characterize the relative performance impact of dynamic key management under varying SA granularity (e.g., *per-site* vs *per-host* tunnel granularity), re-keying strategies, and cryptographic algorithms (e.g., 3DES, AES, HMAC_SHA1). We also discuss results that characterize the relative performance impact of distinct IPsec services, network topologies, and implementation options. The remainder of the paper is organized as follows: Section 2 gives a brief description of the IPsec protocol suite and Section 3 briefly describes the NIIST models. In Section 4, we discuss the detailed experiments and methods designed to analyze the behavior and relative performance characteristics of interacting security protocols. Section 5 gives the analysis and observations of the experiment results. Finally, Sections 6 gives our conclusions.

## 2 Overview of IPsec/IKE

The IPsec protocol suite is a set of protocols that provide security services at the network layer through the use of tunnels between security gateways or individual hosts. IPsec services are provided by two security protocols: the *Authentication Header (AH)* [6] and the *Encapsulating Security Payload (ESP)* [7]. The *Internet Key Exchange Protocol (IKE)* [2] provides dynamic cryptographic key exchange and management functions. The protocol mechanisms used by IPsec are independent of specific cryptographic algorithms and network topologies, thus allowing complete flexibility in the selection of cryptographic algorithms to address differing security requirements on a flow-by-flow basis.

AH provides data integrity, data origin authentication and optional replay protection for IP datagrams. ESP can provide confidentiality, data integrity, data origin authentication, anti-replay, and limited traffic flow confidentiality. Both AH and ESP support two types of encapsulation modes: transport mode and tunnel mode. In transport mode, IPsec is used between the original source and final destination hosts. In tunnel mode, IPsec is commonly used between two intermediate *security gateways (SGs)* that protect the communications of one or more hosts residing behind them, or between a security gateway and a remote host.

All IPsec services require the pre-establishment of *security associations (SAs)* between the SGs, or hosts, that will serve as end points of the IPsec tunnel. An SA is the negotiated, shared set of state information necessary to enable one-way communication between two IPsec peer entities (e.g., chosen cryptographic algorithms, security protocols, keys, traffic filters, etc). Two SAs are required to support bi-directional communication between two peers (one in each direction). IPsec allows the user to control the granularity of SAs (e.g., port-based, protocol-based, host-based or site-based tunnels) depending on the local security policy.

IPsec implementations contain a *security association data base (SADB)* that stores the current set of established SAs. The contents of the SADB are negotiated by the IKE protocol. In addition, the *security policy database (SPD)* specifies what IPsec services are to be provided to distinct IP traffic flows and in what mode of operation. IPsec modules check each inbound/outbound IP packet against the SPD to determine if IPsec services are required. Packets requiring, or already containing, IPsec headers are checked against the SADB to ensure processing that is consistent with the negotiated SA.

The IKE [2, 9, 10] protocol provides automated cryptographic key exchange and management mechanisms for IPsec. IKE is used to negotiate security associations for use with its own key management exchanges (called Phase 1) and for other services such as IPsec (called Phase 2). This is a two stage process. First a Phase 1 (IKE) SA must be established, then the Phase 2 (IPsec) SA is negotiated using the services provided by the Phase 1 relationship. Phase 1 can be accomplished either in Main mode or Aggressive mode. Main mode provides identity protection. Aggressive mode can be used with reduced round-trips when identity protection is not needed. Both modes use the Diffie-Hellman (DH) public key exchange for creating shared secret keys. Phase 2 is used to negotiate SAs for other security protocols (e.g., AH and ESP) and is accomplished in Quick mode. Once phase 1 is established either party can initiate Quick Mode since the IKE SA is bi-directional. Phase 2 generates two SAs, one in each direction. A single IKE SA can be used to negotiate more than one IPsec SAs. When perfect forward secrecy is desired, phase 2 exchange can initiate a new Diffie-Hellman exchange for new and fresh keying material.

IKE supports re-keying established SAs after a negotiated period of time, or usage level (i.e., bytes sent/received). Both Phase 1 and Phase 2 SAs can be re-keyed and either end point can initiate the re-keying operation. Typically, key life times are specified in terms of an upper bound and a threshold (e.g., 2400 seconds and 95%). The threshold indicates when to initiate a rekeying operations, the life time indicates when the existing SA must be deleted.

## 3 NIIST and Models of Security Performance/Behavior

The NIIST framework provides detailed, packet level models of IPsec/IKE protocols and the tools to easily build large scale models of VPN scenarios. Our objectives in developing NIIST were to model the performance and behavioral impact of security protocols on overall distributed system performance. In particular, we model the impact of IPsec security management protocols, cryptographic processing delay and packet processing rules. NIIST supports the ability to parameterize a simulation model in terms of network topologies, security policies, IPsec/IKE parameters, performance of specific cryptographic transforms, and specific packet processing details.

The goal of NIIST is to enable relative performance characterizations of the impact of the variables above on end-to-end applications. NIIST provides instrumentation to enable one to conduct detailed analyses of IPsec/IKE performance and its effect on end-to-end protocols such as TCP/FTP. NIIST is not designed for evaluating the underlying security properties of these protocol suites, and as such, abstracts actual cryptographic techniques away to only model their impact on performance.

## 4 Experimental Environment

### VPN Topology

In the experiments reported here, we consider VPNs in which a security gateway connects a given site to every other site over IPsec tunnels (i.e., a full mesh of tunnel mode IPsec among SGs). We use a network configuration of N sites, each consisting of M hosts and a single SG. The hosts and SG at each site are connected by a 100Mbps LAN, the SGs are interconnected by a mesh of 1.5Mbps WAN links with a propagation delay of 50ms. The network MTU is assumed to be a constant 1000 bytes. A simple TCP-based client server application is used among the hosts.

In order to vary the workload presented to the VPN, and IPsec/IKE in particular, we examine three variations of this basic VPN configuration:

- Asymmetric Hosts *(asymmhost)*: a configuration where half the hosts in each site are clients and the other half of the hosts are servers;
- Asymmetric Networks *(asymmnet)*: a configuration where half the sites contain only hosts that are clients and the other half of the sites contain only hosts that are servers; and,
- Fully Symmetric *(fullsymm)*: a configuration where each host in every site acts as both a client and a server.

Another aspect of VPN topology is the distribution and granularity of the IPsec tunnels that logically interconnect its sites. As we noted above we are assuming a full mesh interconnectivity among the VPN sites. Within this environment, we consider security policies that result in two tunnel granularities: *per-site* creates a single IPsec SA for all the hosts behind a given SG; and *per-host* creates an unique IPsec SA, at the SG, for each communicating host pair.

### Workload Models and Experiment Parameters

Our study uses simple fixed size file transfers (i.e., FTP) over TCP as the VPN application. The parameters of this traffic model include the size of the file to transfer and the time/space distribution of FTP requests. Each application cycle, an FTP client wakes up, randomly chooses one target FTP sever, performs a FTP-get of a fixed size file, then goes to sleep for another random period.

While NIIST is capable of modeling large VPNs over long time scales, for the specific study reported in this paper, we chose to model a modest topology but with security policies and application workloads scaled so as to generate significant IPsec/IKE activity. In particular, we simulated 8 hours of activity in a VPN with 10 sites and 5 hosts at each site. For our application workload we considered two file sizes (1K Byte) and (1M Byte). The sleep time between client transfers was chosen from an exponential distribution with mean of 10 secs.

IPsec/IKE parameters were chosen so as to generate significant activity given the scenario above. In particular, all Phase 1 (IKE) SAs used a life time of 2400 seconds, while Phase 2 (IPsec) SAs were set to 800 seconds. We allowed either security gateway to initiate the re-keying negotiation, if required by the local security policy. But, for phase 2 re-keying, only outbound SAs are allowed to rekey.

For all experiments, we used perfect forward secrecy for Phase 2 and IKE SAs and IPsec SAs used the same cryptographic algorithms, if required. The IKE default authentication mode was a pre-shared secret key. The default SG policy was to drop all IP packets that required security, but that did not have an established SA. The performance of cryptographic algorithms used for the experiments were taken from [11] scaled to a 200Mhz clock cycle. In particular, our experiments used the following encryption and decryption processing rates: 3DES_CBC = 1481KB/s, HMAC_SHA1 = 15384KB/s, AES_CBC = 12500KB/s, DH group 2 delay = 0.1 sec.

These topology and workload models are not meant to model any specific (i.e., realistic) application or VPN scenario. Instead they are chosen to present a cumulative workload to the security gateways that is demonstrative of various issues in IPsec VPN performance. For example, the smaller files highlight the impact of IKE negotiations on TCP session establishment, while the larger files highlight the impact of IPsec security transforms on the TCP data. With cumulative workloads on the order of 100,000 TCP sessions and 60,000 security associations, this workload model results in SG activity indicative of much larger networks observed over much longer times.

Table 1 shows traffic load (i.e., TCP sessions and IKE/IPsec SAs) generated from the various topology and workload models. Due to space limitations, in this paper, we only present the results from a *fullsymm* network configuration for analysis and characterization.

### Performance Measures

The NIIST is capable of producing performance measures at various protocol levels and observation points within the VPN. At the SGs we focus on the dynamics of IKE behavior. The metrics of interest to this paper include basic counts of the number and type of SAs created (we distinguish between initial SAs and those created due to rekeying) and the number of IP packets discarded due to IPsec (no SA available).

For each type of SA (Phase1/Phase2, initial/rekey), we examine SA establishment latency, which is a measure of time taken to establish the SA as seen by the initiator. IKE SA latency is measured as the time taken from sending the first IKE message and to receiving the last (6th) message (requiring 3 round trips). IPsec SA latency (for an

| | *asymmnet* | | | | *asymmhost* | | | | *fullsymm* | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | *per-site* | | *per-host* | | *per-site* | | *per-host* | | *per-site* | | *per-host* | |
| | 1KB | 1MB | 1KB | 1MB | 1KB | 1MB | 1KB | 1MB | 1KB | 1MB | 1KB | 1MB |
| Appl.:<br># of sess | 70228 | 51482 | 70333 | 51452 | 84572 | 62078 | 84833 | 61622 | 140771 | 103793 | 137900 | 101382 |
| IKE:<br>Init req<br>Rekey req | 25<br>389 | 25<br>388 | 25<br>389 | 25<br>388 | 45<br>700 | 45<br>699 | 45<br>699 | 45<br>701 | 45<br>702 | 45<br>701 | 45<br>702 | 46<br>700 |
| IPsec:<br>Init req<br>Rekey req | 25<br>1214 | 25<br>1215 | 625<br>30048 | 625<br>29933 | 47<br>2283 | 48<br>2345 | 540<br>26147 | 540<br>26097 | 46<br>2237 | 46<br>2288 | 1125<br>54402 | 1126<br>57238 |
| Pkt drops<br>(noSAs) | 26 | 26 | 625 | 625 | 49 | 48 | 540 | 540 | 48 | 48 | 1131 | 1126 |

Table 1. TCP Sessions and IKE/IPsec SAs Created From Each Network Configuration using *ESP (AES_CBC+HMAC_SHA1)*

outgoing SA) is measured for the time taken from the initial IPsec SA request to receiving the corresponding message from IKE to add the established SA to the security association database (SADB). IPsec SA latency may also include Phase 1 set-up time if no IKE SA is available. When a corresponding IKE SA is available, IPsec SA set-up requires 2 round trips.

At the application layer/hosts, we examine FTP session throughput, session latency, the number of sessions that succeeded, and the total number of TCP retransmissions.

## 5 Selected Results and Analysis

In this section we present simulation results to characterize the relative performance impact of security policies, implementation options in IPsec/IKE suites, and dynamic SA establishment.

### 5.1 Impact of Security Policies

A security policy specifies what security services are required for specific inbound/outbound IP flows. The policy defines the level of protection required for the traffic. If a flow requires secure communication, the SA specifies the type of security services (e.g., authentication and/or encryption), the specific cryptographic algorithms (e.g, AES, 3DES, HMAC_MD5, HMAC_SHA1), and SA granularity (e.g, *per-host* vs. *per-site*).

To explore the impact these policy decisions have on the overall performance of VPN applications, we examined both per-site and per-host SA policies with the following protection levels: 1) Bypass: No IPsec required; 2) AH with HMAC_SHA1; 3) ESP with 3DES and HMAC_SHA1; 4) ESP with AES and HMAC_SHA1; and 5) ESP with AES and null authentication. Although null authentication should never be used, we have included it for the statistical comparison purpose.

Note that Bypass results in no IPsec/IKE operations and is the equivalent of operating in an open network, without any IPsec VPN services.

#### Effect of SA Granularity

Figure 1 shows the effect SA granularity has on overall end-to-end application performance. The results show that the performance of FTP in a VPN using *per-site* granularity is significantly better than that of *per-host*. The results for the small, 1KB, files are dominated by the impact of IKE negotiations. In this scenario we see from table 1, that the fullmesh, 1KB, per-site policy results in approximately 2,300 Phase 2 (IPsec) SA establishments, while a similar per-host policy results in approximately 55,500 Phase 2 establishments. This increased IKE activity results in more *noSA* packet discards (1131 vs 48) and increased wait times. For short sessions, Figure 1 clearly shows that the impact of the key management delays and induced packet drops far out weigh the impact of the choice of cryptographic transforms. As we will see in the next section, as the volume of data transferred increases, the choice of security services has a much more pronounced impact on VPN application performance.

#### Effect of Security Services

Figure 2 depicts the performance of FTP under security policies dictating varied IPsec security services. Here we note that the relative performance of the selected cryptographic transforms begin to dominate the overall performance of the application. Most notably ESP with 3DES performs worse, even at per-site granularities, than either of the AES based ESP transforms. Still, for a given IPsec transform, the per-site policies still result in improved overall performance (as high as 10%).

To understand the effect of IKE dynamics in greater detail, we examine Table 2. This table, derived from

| | AH(h_sha1) | | ESP(3des+h_sha1) | | ESP(aes+h_sha1) | | ESP(aes+null) | |
|---|---|---|---|---|---|---|---|---|
| | *per-site* | *per-host* | *per-site* | *per-host* | *per-site* | *per-host* | *per-site* | *per-host* |
| IKE: | | | | | | | | |
| Init SA delay (s) | 0.538 | 0.572 | 0.567 | 0.571 | 0.537 | 0.574 | 0.538 | 0.565 |
| Rekey SA delay (s) | 0.505 | 0.522 | 0.539 | 0.580 | 0.506 | 0.522 | 0.505 | 0.520 |
| IPsec: | | | | | | | | |
| Init SA delay (s) | 0.847 | 0.331 | 0.894 | 0.344 | 0.842 | 0.331 | 0.850 | 0.331 |
| rekey SA delay (s) | 0.403 | 0.461 | 0.428 | 0.452 | 0.403 | 0.459 | 0.402 | 0.460 |

Table 2. IKE/IPsec SA Establishment Latency in Various Scenarios

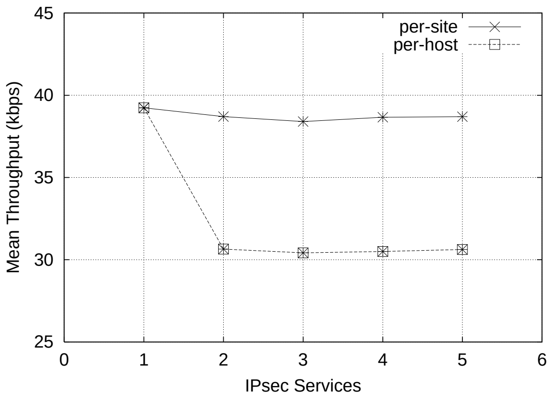


Figure 1. Effect of SA Granularity on FTP Performance (file size of 1KB). IPsec Service: 1=Bypass; 2=AH(hmac_sha1); 3=ESP(3des+hmac_sha1); 4=ESP(aes+hmac_sha1); 5=ESP(aes+null)

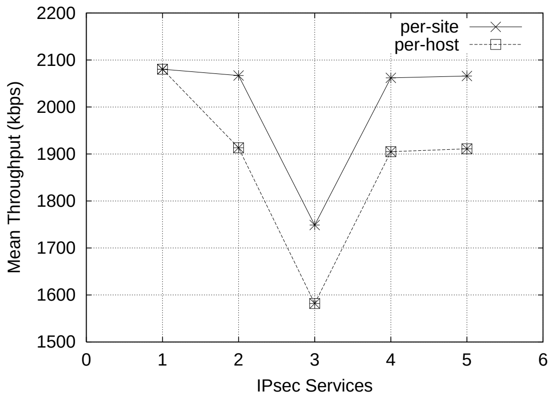


Figure 2. FTP Performance (file size of 1MB) using Various Cryptographic Algorithms. IPsec Service: 1=Bypass; 2=AH(hmac_sha1); 3=ESP(3des+hmac_sha1); 4=ESP(aes+hmac_sha1); 5=ESP(aes+null)

detailed NIIST instrumentation of IKE protocol models, shows detailed latency performance for the various phases of IKE SA establishment in each simulation scenario presented in this paper.

Examining this detailed IKE performance we note two things. First, by comparing entries for the same metric across differing IPsec security services, we note that IKE key management performance is not significantly impacted by the choice of security services being negotiated. Second, we note further detail underlying the impact of per-site vs per-host security policies. In particular we note that latency for the IPsec initial establishment (*Init SA delay*) (requiring 2 round-trip IKE message exchanges) is higher than that of IKE (requiring 3 round-trips) for *per-site*, but is lower for *per-host*. These results show the basic impact of IKE behavior in Phase 1 and Phase 2. Under *per-site* policies, every IPsec initial SA request must wait for an initial IKE (Phase 1) negotiation. Thus, the IPsec Init SA delays include both the time required to put the corresponding Phase 1 SA in place and the time required complete the Phase 2 negotiation. Since a single IKE SA between a pair of SGs can be reused to negotiate any number of Phase 2 SAs, we see that the corresponding IPsec Init SA delay for *per-host* policies is much better. The cost of establishing the Initial IKE SA is amortized across numerous IPsec SAs that reuse it and thus experience no Phase 1 delay. Examining the IPsec rekeying delays we see that per-site and per-host scenarios are very similar. Here, even under per-site policies, the Phase 2 rekeying almost always occurs over an existing Phase 1 SA since we set the Phase 1 lifetime to be longer than the Phase 2 lifetime. Looking back at Table 1 we can see the one-to-one correspondence between Initial IKE and IPsec SAs under per-site policies, and the ratio of IKE SA reuse under per-host policies.

## 5.2 Impact of IPsec/IKE Implementation Options

In this section, we look at some implementation details of IPsec/IKE that can have significant impact on overall VPN performance. The NIIST has the capability to model various key implementation details of IPsec/IKE protocol suites. We examine two such details below.

## Various Re-keying Techniques

The IKE specifications leave unspecified when an Initiator should switch over to transmitting on a newly rekeyed SA following transmission of the final IKE re-keying message. In this section, we illustrate the performance trade-offs inherent in four re-keying implementation strategies that have been observed in real implementations.

The IPsec re-keyed SA cut-over options examined include:
1) *DeleteMsg*: cut-over when an explicit Delete message for the old SA is received;
2) *fixed delay*: cut-over when either receiving inbound traffic with the new SA or, in the absence of incoming traffic, after a fixed amount of time (e.g, 30s) has elapsed;
3) *twice measured RTT*: cut-over to the re-keyed SAs after either receiving inbound traffic using the new SA or after twice the measured round trip time has elapsed; and,
4) *immediateUse*: cut-over immediately to using the re-keyed SAs.

Table 3 shows the relative impact of these implementation options on various aspects of IPsec/IKE and application performance. We note that the simplest implementation option, *immediateUse*, results in the poorest overall performance. While this option results in the least rekeying delay (zero additional wait in establishing the final rekeyd SA), the resulting number of dropped packets is an order of magnitude higher than the other approaches. These excessive drops are the result of the re-key initiator establishing the new SA (and deleting the old) as soon as the IKE Quick Mode 3rd message has been sent out. Unfortunately, the responder will continue to transmit on the old SA until this Quick Mode 3rd message is received.

While it is clear that *immediateUse* is a poor choice of cut-over strategy, we see that any of the other approaches that allow time for the final re-keying message to arrive at the responder, can produce reasonable performance. Clearly, use of the optional *DeleteMsg* from the responder produces the best performance because, it ensures that both the initiator and responder have complete re-keying before cutting over to the new SA. Even if this IKE option is not implemented, any techniques that delay long enough for the responder to complete re-keying is useful. Techniques that attempt to dynamically measure this wait time (e.g., *RTTD2*), perform better than those based upon a fixed upper bound wait time (e.g., *FixedDelay*). In particular, when re-keying occurs during period of no traffic, or in simplex paths, the fixed delay schemes must wait for much longer periods to complete re-keying.

The subtle behavior of IKE during rekeying has been documented as the source of numerous interoperability failures and performance problems during bakeoff testing of real implementations. In our experiments we noted that the re-keying implementation options studied produced a extensive range of behavior and performance results. Inconsistent choices among these options among peer SGs can result in re-keying failures, thrashing of initiator/responder roles, long periods of inconsistent SA state and SA black holes.

## IPsec Handling Options for the First TCP SYN Packet

In this section we investigate the implementation options for the handling of the first IP packet (e.g., TCP SYN message) when no SAs are available on the performance of TCP based applications.

The IPsec specification states that received IP packets are to be dropped when security policy requires IPsec but no appropriate SA currently exists. However, Figure 3 shows that application performance can be significantly improved through a simple implementation optimization of keeping the first IP packet (typically a SYN packet for a TCP connection) at the gateway and transmitting it to the peer SG as soon as the IPsec SA negotiation is complete. This can result in session latency performance improvement proportional to the initial TCP retransmission timeout.

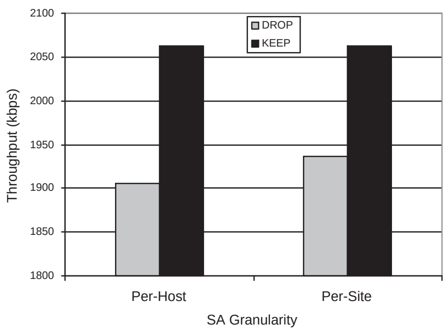


Figure 3. Impact of Handling Options on the Initial SYN Packets When No SAs Available. ESP with AES+HMAC_SHA1. File size of 1MB

## 5.3 Impact of Dynamic SA Establishment

While in previous sections we have examined the performance impact of all aspects of IKE behavior in great detail, the choice to use dynamic key management at all is a policy decision. While the primary considerations in this decision are security related, we examine the performance trade offs of using IKE key management for both Phase 1 and Phase 2, Phase 2 only (i.e., SGs have a pre-established, static IKE SA in place), and not using IKE at all (i.e., static Phase 1 and Phase 2 SAs). Figure 4 depicts the impact on application latency of these three scenarios: 1) Dynamic P1+P2 - dynamically created IKE SA followed by IPsec SAs; 2) Dynamic P2 - statically created IKE SA and dynamically created IPsec SAs; and 3) Static P1+P2 - manually preinstalled IKE SA and IPsec SAs.

| | DeleteMsg | Fixed Delay | RTTD2 | ImmediateUse |
|---|---|---|---|---|
| Application: | | | | |
| # of sess (1MB/sess) | 101382 | 99929 | 101419 | 101608 |
| Avg Thrput (kbps) | 1904.955 | 1900.938 | 1904.564 | 1896.463 |
| Avg sess. delay (sec) | 4.200 | 4.208 | 4.200 | 4.218 |
| # of retransmissions | 1126 | 1130 | 1145 | 2036 |
| IPSec: | | | | |
| Rekeying requests | 57238 | 56297 | 57296 | 57249 |
| rekeying SA delay (s) | 0.459 | 27.755 | 0.918 | 0.309 |
| pkts dropped (No SA) | 1126 | 1129 | 1145 | 19839 |

Table 3. Performance trade-offs in Phase 2 Re-keying Techniques, file size of 1MB, ESP (AES_CBC and HMAC_SHA1)

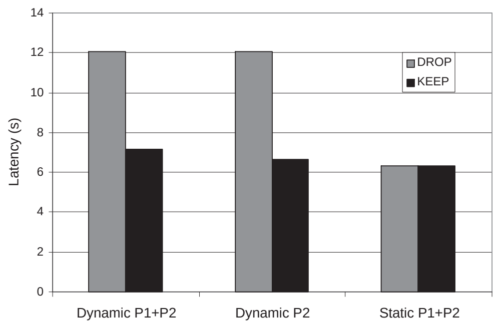


Figure 4. Impact of Dynamic SA negotiation on the FTP Performance (File size of 1MB)

Figure 4 depicts the impact of dynamic SA establishment on TCP application latency. Here we see that SG implementations that drop all packets pending SA negotiation (*drop*), suffer increased average latency. In this scenario, the additional latency is bounded by the TCP initial retransmission timeout. The performance of the *drop* policy is identical under both dynamic IKE scenarios because this retransmission timeout is longer than the IKE latency for both Phase 1 and Phase 2.

Looking at the *keep* implementation policy, we see that the overall latency impact of dynamic key management, for medium sized file transfers, is not significant. In the simple scenarios presented here, packet drop implementation details are more significant than policy decisions with respect to when to use dynamic key management.

## 6 Conclusions

In this paper, we have introduced the NIST IPsec/IKE Simulation Tool (NIIST), and presented results of its use to examine and characterize dynamic behavior and relative performance of VPN environments based upon IPsec/IKE technologies.

We have characterized the performance impact of dynamic key management and IPsec security policies under various network configurations. We have also highlighted the significant performance impact of dynamic key management and IPsec/IKE implementation details on the overall performance and behavior of TCP based applications.

While, clearly, the detailed quantitative results of such simulations are highly dependent the specific scenarios and parameters used, we believe that this work highlights several important factors that will significantly impact the relative / qualitative performance in any such scenario.

## References

[1] S. Kent and R. Atkinson, *Security Architecture for the Internet Protocol*, RFC 2401, November 1998.

[2] D. Harkins and D. Carrel, *The Internet Key Exchange (IKE)*, RFC 2409, November 1998.

[3] Sheila Frankel, *Demystifying the IPsec Puzzle* (Norwood, Artech House, Inc., 2001).

[4] http://www.antd.nist.gov/niist/

[5] C. Kaufman, Editor, *Internet Key Exchange (IKEv2) Protocol*, draft-ietf-ipsec-ikev2-11.txt, October 9, 2003

[6] S. Kent and R. Atkinson, *IP Authentication Header*, RFC 2402, November 1998.

[7] S. Kent and R. Atkinson, *IP Encapsulating Security Payload (ESP)*, RFC 2406, November 1998.

[8] http://www.ssfnet.org/homePage.html

[9] D. Piper, *The Internet IP Security Domain of Interpretation for ISAKMP*, RFC 2407, November 1998.

[10] D. Maughan, M. Shertler, M. Schneider, J. Turner, *Internet Security Association and Key Management Protocol (ISAKMP)*, RFC 2408, November 1998.

[11] N. Ferguson and B. Schineier, Helix: Fast Encryption and Authentication, *Dr. Dobb's Journal, Computer Security*, #354 November, 2003, 28-34.