Simulated Effects Of
PGP Encryption On Network Performance

Stephanie Podtburg
Microcomputer Studies Department
St. Cloud State University
post9801@stcloudstate.edu

Abstract

Network security is a major concern in the computer industry. Many different encryption techniques and programs have been created and implemented. PGP encryption is a popular security tool. PGP encryption is utilized in many e-mail programs to protect data transmissions. Another common use of PGP is data encryption within private networks. The building blocks for PGP are IDEA, a public key technology, and RSA, a private key technology. Both of the technologies implemented create overhead in the form of processing time, increased network traffic, and may increase network downtime.

I have simulated the effect of utilizing PGP encryption technology on a hypothetical network. My simulation involves the use of PGP and RSA to encrypt text documents of varying lengths. The documents were transmitted across a network while their effects on the network were traced using tcpdump, a packet sniffer built into Linux RedHat 6.2. Once the data collection was completed, I examined the data using SAS programming techniques. The results of the SAS analysis were then utilized in a network simulation using Comnet III. I have included simulations based on 10 MB/S Ethernet, 100 MB/S Switched Ethernet, and a network with an OC-12 backbone similar to ATM.

Introduction

In the past few years, many software engineers have integrated the use of PGP encryption in their e-mail and file security programs. You can find PGP encryption technologies in programs such as Eudora and Outlook Express. Versions are available for Macintosh computers, Unix, and Windows operating systems allowing users to communicate across platforms. E-mail encryption is PGP's most common use. E-mail and network hackers have created an environment where computer users feel the need to encrypt any documents they transmit across networks or store on disks accessible by other users.

What effects will the frequent use of encryption have on network efficiency? The link between PGP and RSA is unavoidable due to the integration of the RSA algorithm into the PGP algorithm. RSA requires several mathematical operations for key generation and data encryption that may overload networks and computers that are approaching maximum utilization limits. Constant calculations of keys and encryption tables take processing time away from other necessary functions such as entering and retrieving files from file servers, calculating business related statements, and general computing activities. Many companies continue to utilize shared Ethernet networks with legacy software and hardware components. Are these networks able to survive computer downtime, overflowing buffers, and delayed transmissions due to extensive use of encryption techniques? Will automatic encryption using PGP create situations where a network is unable to efficiently serve the needs of the business?

What is PGP Encryption?

PGP encryption is a unique combination of public and private key encryption algorithms created as freeware by Phil Zimmerman in 1991. It has since been purchased by Network Associates, Inc [1]. PGP, Pretty Good Protection, has been improved upon many times since its release in 1991. PGP's current format follows its original structure. PGP is based on four main modules: a symmetric key based on IDEA, a public-key from RSA, an MD5 one-way hash algorithm, and a random number generator used to create the encryption keys [2]. Newer versions of PGP encryption software offer options such as Diffie-Hellman key generation, DES, and Triple DES for greater versatility and security. By offering more options within the program, PGP has become even more secure than it was with the original four components. Integrating all of these algorithms creates a layered environment that is both secure and portable.

The basic scheme for PGP e-mail packet is shown in figure 1[3].

RSA encrypted symmetric key & header	DataGram encrypted with IDEA

RSA header	IDEA encrypted datagram with header & symmetric key encoded with RSA

Figure 1: PGP Encrypted Packet

This form of packet is also referred to as an RSA envelope. The theory behind RSA envelopes is that the data is encrypted using the less secure symmetric key encryption to save on the processing time required by the more secure RSA public key encryption. IDEA uses a 128-bit key to transform a 64-bit message into a 64-bit result. The main advantage to using RSA

envelopes to send the IDEA encrypted message is decreased demand for processing time and other computing resources. The common key length for RSA is 512 bits. The only part of the packet that will be encrypted using RSA is the IDEA symmetric key and data header. Each packet for the message will contain the IDEA key to open that packet [4]. The savings in computing time and power is resources that will be available for other necessary tasks. Another advantage to this method of encryption is that the symmetric key needed to open the encrypted message may change throughout the session if the data is very sensitive and session time is extended. Changing symmetric keys in the middle of the session offers more security in lengthy transmissions due to the fact that potential hackers will need to decrypt several different keys to decrypt the entire document.

PGP takes RSA envelopes two steps further. The MD5 one-way hashing algorithm is used for digital signatures [5]. The use of digital signatures creates a higher trust level between the sender and receiver. Digital signatures are difficult to forge and allow alert receivers to distinguish between a document that was intercepted and one that arrived at its destination intact. Random sampling from the keyboard or other input devices also creates lower probabilities that the same key is used for encoding two different keys. Random sampling from input devices is more random than seed generation. The same patterns of key strokes or mouse motions would be required to create identical keys.

All four components in PGP create a secure environment to store and share data. Intruders would be required to break into at least three different security devices to read encoded data. Newer versions of PGP also offer options that appear friendlier to less experienced users. One option is to send data using the ASCII radix-64 format. ASCII radix-64 is a printable form of the PGP encryption scheme [6]. This option is handy when sending e-mail, but less necessary for file storage. Less experienced users may find the ASCII radix-64 format more friendly in appearance than the normal PGP non-printable encoded format. Other current options are point and click implementation.

Data Compression

PGP is unique in another way. PGP encrypted text is compressed prior to encryption [7]. Many encryption algorithms expand file size. Datagrams are compressed by the PGP software before they are run through the encryption algorithm producing ciphertext. Ciphertext is often more lengthy than regular text. The RSA algorithm I used increased some files by nearly one hundred percent. The traffic and calculation overhead would likely slow transmissions on slower mediums such as modems and ISDN lines. PGP uses two types of compression: LZS and Deflate. While LZS and Deflate save resources on the slower medium, they have the opposite effect on faster mediums such as cable modems, DSL, T-1 and T-3 lines [8]. This is interesting. Compression routines produce a lot of overhead. The decrease in efficiency with higher speed medium may be due to the inability of the processor to keep up with the ability of the data lines to transfer data. Many networks would theoretically experience the opposite. Processors are quite often faster and more efficient than the lines over which they transmit data.

Table 1 shows the data collected on file sizes before and after encryption. The results were surprising. As a point of comparison, I used RSA to encrypt the text documents that were used in the simulation.

Table 1:

File Name Key Length	Plain Text	RSA enc	PGP RSA 1024	PGP RSA 1536	PGP RSA 2048	PGP DH 1024	PGP DH 1536	PGP DH 2048	PGP DH 3072
5kb.txt	5	13	2.6	2.45	2.75	2.47	2.6	2.73	2.97
96kb.txt	96	191	3.24	3.39	3.54	3.39	3.51	3.64	3.89
307kb.txt	307	799	4.77	4.8	4.99	5.07	5.2	5.32	5.57
1790kb.txt	1838	4795	14.6	14.5	14.6	16.3	16.4	16.5	16.8

File Size in Kilobits RSA = RSA Key DH = Diffie-Hellman

All the documents were created in text mode and then encrypted. The RSA algorithm more than doubled the size of the original file. By design, higher key lengths require more processing time and create larger proportional changes in file size. The PGP Diffie-Hellman key length of 3072 created the largest files. The largest file I used in my experiment was 1790 kilobits. After passing through the PGP algorithm, the file size decreased to 16.8 kilobits. These files were compressed in MSDOS to maintain a consistent atmosphere with the RSA algorithm. Encrypting the 1790-kilobit file was also encrypted in Windows 95. The result was a 17.2-kilobit file.

Utilizing PGP for file storage appears to save disk space. However, each person who needs to access the file must also possess the public key to decompress that file every time they have to access the file. People who are concerned about running out of disk space that have sufficient processing power to constantly decrypt files may see a benefit to storing files in PGP format. The drawback to this idea is that the 1790-kilobit file required 15 seconds to decode with a Pentium II processor. The major benefit to the file compression appears to be the decreased time required to form packets for transmission. Individual firms and users need to weigh the cost of the added security and increased transmission speed across media against the processing time required to open and close the file each time it is accessed.

Simulation

Many of the basic factors involved in PGP encryption raise questions relating to the cost of implementing an encryption scheme like PGP. PGP is loaded with overhead that would not be present without using encryption. However, network security and privacy issues have created the need for more forms of encryption that are both stronger and more universal. The simulation that follows answered many of these questions. No one answer is correct about whether or not a company should implement the frequent use of PGP encryption.

Step One: Data Collection?

As with many other problems, the first step in answering a question is to decide what is really needed to arrive at a conclusion. A number of ideas come to mind immediately. The major issues at hand are: processing time, transmission speed, network traffic, and other side effects. Processing time is difficult. Unix utilizes the ps -l command to evaluate processor utilization and how long a process has been running. Since this simulation was performed using windows based programs, times were manually logged. These times are recorded in Table 2. Time requirements were statistically similar for both PGP and RSA. Time differences were also insignificantly different between different key lengths with the exception of the largest file. Time increased

from eight to approximately nine seconds when the 1790-kilobit file was encrypted using the 3072-bit key. Results will vary based on processing resources built into different computers.

Table 2: Processing Times in Seconds

File Name Key	Time
5kb.txt	.5
96kb.txt	.75
307kb.txt	2.25
1790kb.txt	8.0

Transmission times and network traffic were logged through the use of tcpdump. Tcpdump is a packet sniffer that has the capability of listening to multiple ports while specifying the data to be collected. Transmission statistics were the most difficult to collect. Collecting transmission data requires root access to a computer running tcpdump, administrative rights on a computer capable of receiving e-mail or ftp transmissions, and rights to place files onto that computer. Due to security restrictions for network resources, these file transfers were completed between a personal computer running Windows 95 or 98 and a Linux server that was set up for this experiment. Ftp was the process used to simulate the movement of the encrypted and unencrypted traffic across the network.

A telnet session was established and logged to enable the tcpdump data collection. Ports 20 and 21 were monitored for ftp traffic on the Linux Server. The data collected on each packet were arrival time, and packet size, source address and destination address, and packet content.. Times and transmission speeds were also tracked using the ftp session. Transmission speeds vary due to network utilization. Several iterations of the transmission cycle are required to obtain accurate data regarding traffic patterns. A control session is also needed. To create the control session, I logged network traffic to all ports except ftp, and telnet. Data collected from the control run will represent background traffic not generated by the encryption data gathering. This data was then analyzed to arrive at a single expected transmission speed per packet. Estimated packet sizes were also obtained based on the control run.

All these data sets will be utilized in the creation of system parameters for the individual simulations. One key factor in the data gathering if you use the ftp method of simulating e-mail traffic is to specify the binary data type. PGP encrypted data is in an unprintable, binary format. If the data type is not set to binary for the transfer, the ftp session will result in the transferring of many empty data lines. Your transfer rates will not be correct.

Step 2: Analyze Raw Data

Once the data was collected, it was imported into SAS. SAS programs were used to read in the data and analyze it. Interarrival rates were calculated by calculating the difference between the arrival times of consecutive packets. Expected packet sizes were calculated by calculating the average packet size and its standard deviation. Expected interarrival rates were calculated based on an experimental distribution of the times collected. The SAS best fit procedures and transformations on the data did not produce a valid distribution. Therefore a distribution of intarrival times was created by calculating the probability associated with time intervals. The

expected packet size and the arrival distribution will be used to set the parameters for packet size and interarrival rates in the Comnet III simulation.

Step 3: Comnet Simulation

Once calculations have been completed to determine system parameters such as interarrival rates, packet size and processing times, the Comnet simulation may be executed. Figure 2 is a diagram of the simulated network. There will be three different networks involved in this analysis. The first network is a 10 MB/S shared Ethernet network implementing CSMA/CD. This network is subject to collisions resulting in exponential back-off's.

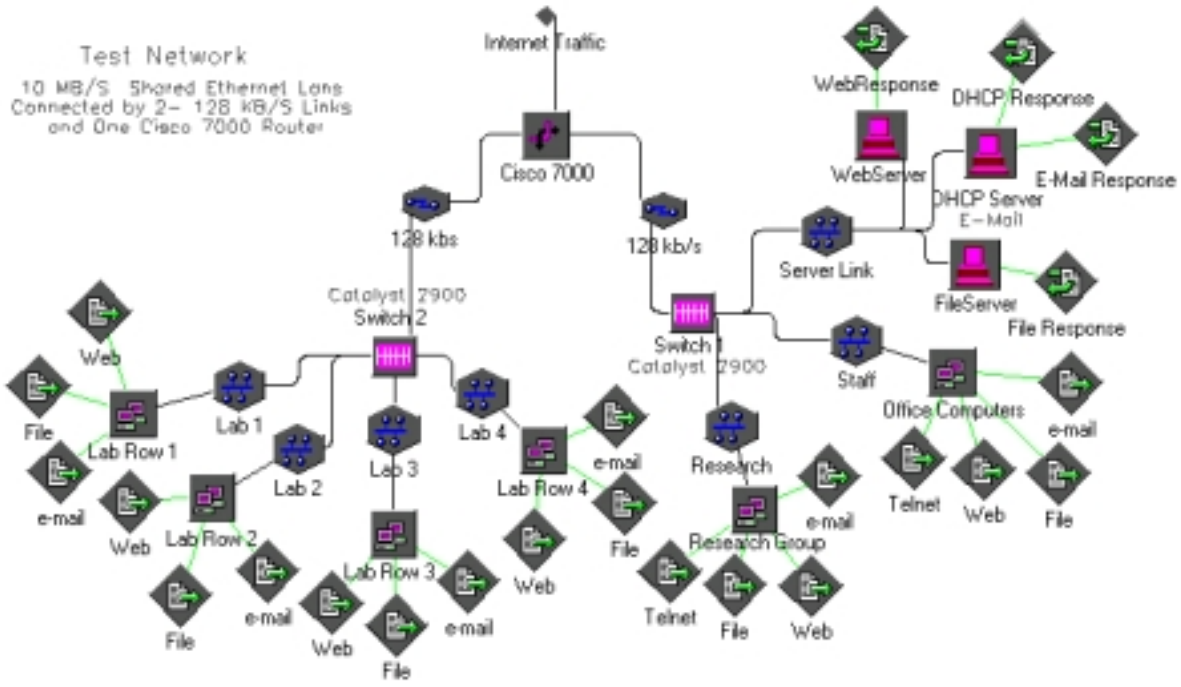

Figure 2: Network Diagram for Switched 100 MB/S Ethernet

Each of the three simulated networks is made up of a Cisco 7000 router handling all routing requests, both external and internal. Traffic then travels through one of two point to point connections leading to either a Cisco Catalyst 2900 switch or an Asante 10/100 Fast Ethernet Bridge. The physical differences between the three models are shown in Table 3.

Table 3: Physical Changes in Network Model

Model	Transmission Media	Switch or Hub
10 MB/S Shared Ethernet	128 MB/S	Asante 10/100 Bridge
100 MB/S Switched Ethernet	128 MB/S	Cisco Catalyst 2900
Simulated ATM	OC-12 for ATM	Cisco Catalyst 2900

Application differences are also taken into consideration. The major differences are the processing times for encrypted e-mail shown in Table 2, traffic differences based on calculations that alter the traffic demands proportionally with overall message size as a result of encryption methods and key length as shown in Table 1. Interarrival rates are altered according to the distribution functions created using the raw data gathered from a real network.

Items that remain constant are the four shared 10 MB/S shared Ethernet lab sections branching from Switch 2. Each lab section is made up of ten workstations with varying workloads. The Research Group also consists of ten workstations and may be reached by passing through Switch 1. The third computer group is Office Computers for staff members. This group is made up of 30 workstations. Finally, the Server Group consists of three servers. One server is used solely for Internet traffic. The second server is a DHCP server that is also the in-house e-mail server. The third server is the network file server. The DHCP server will probably take the heaviest hit, so it is equipped with dual processors. The File and Web servers are equipped with single processors. Traffic on the file server is mostly data traffic. The Web server receives many requests for graphic intensive traffic. Outgoing and incoming e-mail will also travel through the Web server.

E-mail traffic has been weighted to stay in house 90% of the time. The remaining 10% is split between office, lab, web, and research computers. In reality, most of the email would travel out to the Internet, but for the purpose of working the server to its maximum potential, I have routed most of the e-mail to the DHCP server that accommodates internal e-mail.

The first step in running the simulation is to run it based on the unencrypted document. The data collected in the control run is the most symbolic of real network traffic. It includes traffic of all types: file transfers, e-mail, ftp, telnet, and web traffic. This data will be run on all three models of the simulated network. Following that, the data collected from each of the encryption models will be run on all three network models.

Statistics to collect from each run include link, node, and channel utilization. Other necessary statistics are collision statistics, message delays, and link utilization by application. The statistics for link utilization by application are useful for determining how much of the traffic is actually due to e-mail traffic. Collision statistics report where the collisions occur, and the number of times the network had to try to resolve the collision situation. Node statistics will generate results based on the percent utilization of the individual processing nodes such as the lab machines. Link and channel utilization statistics describe how the traffic is passing through the network. This is extremely important in this simulation. Without actual traffic source and destination reports, you cannot see where the delays are created.

Step 4: Analyze the Results

The data in this simulation experiment suggest analysis through regression. The independent factors in this experiment are categorical, and the dependent factors are continuous. The results I have collected up to this point appear to support the idea that the encryption techniques applied in PGP are beneficial for slower network models. Faster processors and transmission media appear to be required for RSA encryption. The ATM simulation using OC-12 links and PGP encryption does indicate a need for faster processors and larger buffers to suggest efficient network operations with the frequent use of PGP encryption. RSA simply creates too much traffic and requires too much processing time to be considered a viable option for extremely

large files. RSA is terrific for smaller transfers such as passwords and short memos. Beyond that, I would suggest PGP encryption.

Conclusion

Compressed message content appears to compensate for increased demands for processing time. Line contention does not appear to be an issue with the shared Ethernet environment. This may be due to the delay created by encrypting the message combined with the presence of fewer packets waiting to cross the network. Circuit switching networks such as the 100 MB/S switched Ethernet network dedicate bandwidth once a connection has been established. For this reason, I conclude that higher processing speeds are required to accommodate for the ability of the high speed transmission lines to transmit data on demand during a given session.

The results of this simulation project lead me to believe that network security is a costly product. Secure transmissions affect productivity and efficiency. If a network is able to survive frequent uses of costly security, it may be a good idea. However, the business world is often reluctant to spend the money required to create a network that is able to maintain acceptable levels of productivity and efficiency. Perhaps a better suggestion is to educate computer users about guidelines for determining the difference between mission critical data and something that could safely be left in unsecured locations.

References

1. (1999, October). "PGP Freeware for Windows 95, Windows 98, Windows 2000 & Windows NT: User's Guide Version 6.5". [Computer Software]. Santa Clara, CA: Network Associates, Inc.

2. Heath, Jim. (2000, February). "How electronic encryption works and how it will change your business. Viacorp. Retrieved December 15, 2000 from the World Wide Web: http://www.viacorp.com/jim.html.

3. Comer, Douglas. (2000). Internetworking with TCP/IP: Principals, Protocols, and Architectures, Fourth Edition. New Jersey: Prentice Hall.

4. Heath, Jim. (2000, February).

5. Scott, C.., Wolfe, P., & Erwin, M. (1999). Virtual Private Netwroks, Second Edition. California: O'Reilly & Associates, Inc.

6. Zimmerman, Philip. (1994, October). "PGP user's guide, volume I: essential topics. PGP. Retrieved December 15, 2000 from the World Wide Web: http://www.PGP.com/FAQ.html.

7. (1999, October). "PGP Freeware for Windows 95, Windows 98, Windows 2000 & Windows NT: User's Guide Version 6.5".

8. (1999, October). "PGP Freeware for Windows 95, Windows 98, Windows 2000 & Windows NT: User's Guide Version 6.5".

9. (1997, September). Comnet III: Tutorial. [Computer Software]. California: CACI Products Company.

Acknowledgements

I would like to acknowledge the advice and assistance of Dr. Dennis Guster from the MicroComputer Studies Department at St. Cloud State University throughout the creation of this presentation.

I would also like to acknowledge the advice of Charles Hall, Network Administrator.