Internet SmartWare Ltd

Bench-Marking SmartGate

Establishing the Scaleability of V-ONE's SmartGate Security System

Peter Cox

July 24th 1997

1. Introduction

V-ONE's SmartGate is designed to provide a highly scaleable security solution to promote the development of Corporate Intranets, Electronic Commerce Systems and On Line services both on the Internet and on other public networks. The scaleability of SmartGate has been a major factor influencing its selection for a number of large-scale projects because the financial viability of those projects has depended on providing secure network services to a large number of users. One such project, currently being implemented in Europe, aims to provide secure Intranet and Electronic Commerce services to several million users. As part of the planning process for this project, extensive benchmark tests were conducted. Tests which SmartGate passed with flying colours. This white paper summaries the key findings of those tests and demonstrates the scaleability of SmartGate.

2. Factors Affecting Scaleability

SmartGate is a client/server security system implementing three related services that combine to provide a complete security solution. The three services are:

- Strong Mutual Authentication
- End to end application level encryption
- Fine grained access control

The responsibility for these services is split between the SmartPass (a module running on the users workstation) and the SmartGate server which runs either as a standalone system or integrated with a Firewall.

The authentication provided by SmartGate uses Smartcards or in the most recent release of the product, Public Key Certificates. For either of these alternatives the SmartGate server creates a user database which holds details of SmartPass users. Those details are accessed when a SmartPass user requests a new secure connection through the SmartGate server to a protected application. This database lookup is handled by the *authentication server* component of SmartGate server. If the users' connection request passes the authentication check then an encrypted connection is created between the SmartPass and SmartGate server. The encryption service is handled by a number of different server modules (depending on the type of service requested), but they can collectively described as the *encryption server*. In the standard SmartGate configuration the authentication server and encryption server reside on the same machine, but this configuration can be changed easily. To provide increased throughput and additional resilience, multiple encryption servers can be run on different systems, sharing a common authentication server and database.

The scaleability of SmartGate is determined by two factors:

- The Total number of SmartGate users that can be defined in the authentication database.
- The number of simultaneous active connections that can be supported by the encryption servers sharing a single authentication database.

Both of these factors were tested as part of the benchmarking tests.

3. Benchmarking Size of User Population

The benchmark tests carried out to determine SmartGate's capabilities for supporting a large number of users were very simple. A SmartGate server was loaded with a user database of various sizes and the time taken for a number of basic transactions measured. The chosen transactions were:

- Add a single user to the SmartGate database
- Establish a single authenticated and encrypted client connection through a SmartGate server to an application server. (POP3 connection to a mailbox).

These transactions were chosen because they represent common operations in and because they both trigger database accesses. The SmartGate server platform chosen for these tests was a very low specification system, a 486 66Mhz system with 16 Mbytes of memory and a 2Gbyte disc drive running BSDI. This system was chosen deliberately, if it can be demonstrated that SmartGate can support a large user population of a system of this type, then there will clearly be no problem supporting a similarly large user base on the systems more likely to be deployed in large projects. The results of these tests are summarised in the following table:

SmartGate Database size	Time to add 1 User	Time to make client Connection	Time to download e-mail (125 K)
20 Users	< 1 sec	3 secs	2 secs
10,000 Users	< 1 sec	2 secs	2 secs
1,400,000 Users	< 1 sec	2 secs	1 secs

These tests conclusively prove that the time required to complete operations that generate database access is independent of the SmartGate user database size. SmartGate provides a highly scaleable solution that can support very large user populations.

4. Benchmarking number of active connections

Benchmarks on the number of simultaneous active connections that can be supported by a SmartGate server were carried out at Sun Microsystems Palo Alto Benchmarking Centre using Sparc Enterprise 3,000 and Enterprise 4,000 servers. These systems were chosen as they represent the mid-range servers in Suns Enterprise range.

The system used to host the SmartGate server was an Enterprise 4,000 server with 8 CPUs and 1 Gbyte memory. A second Sparc Server (an Enterprise 3,000) was used to emulate a varying number of SmartGate users. The tests were able to demonstrate that the E4,000 was able to operate with up to 4,000 simultaneous user connections and *still have spare capacity*. Tests at higher load levels could not be carried out because the system (E3,000) used to simulate the user connections could not reliably generate more than 4,000 connections. Figure 1 charts the spare CPU capacity on the E4,000 running a various numbers of simultaneous SmartGate connections.

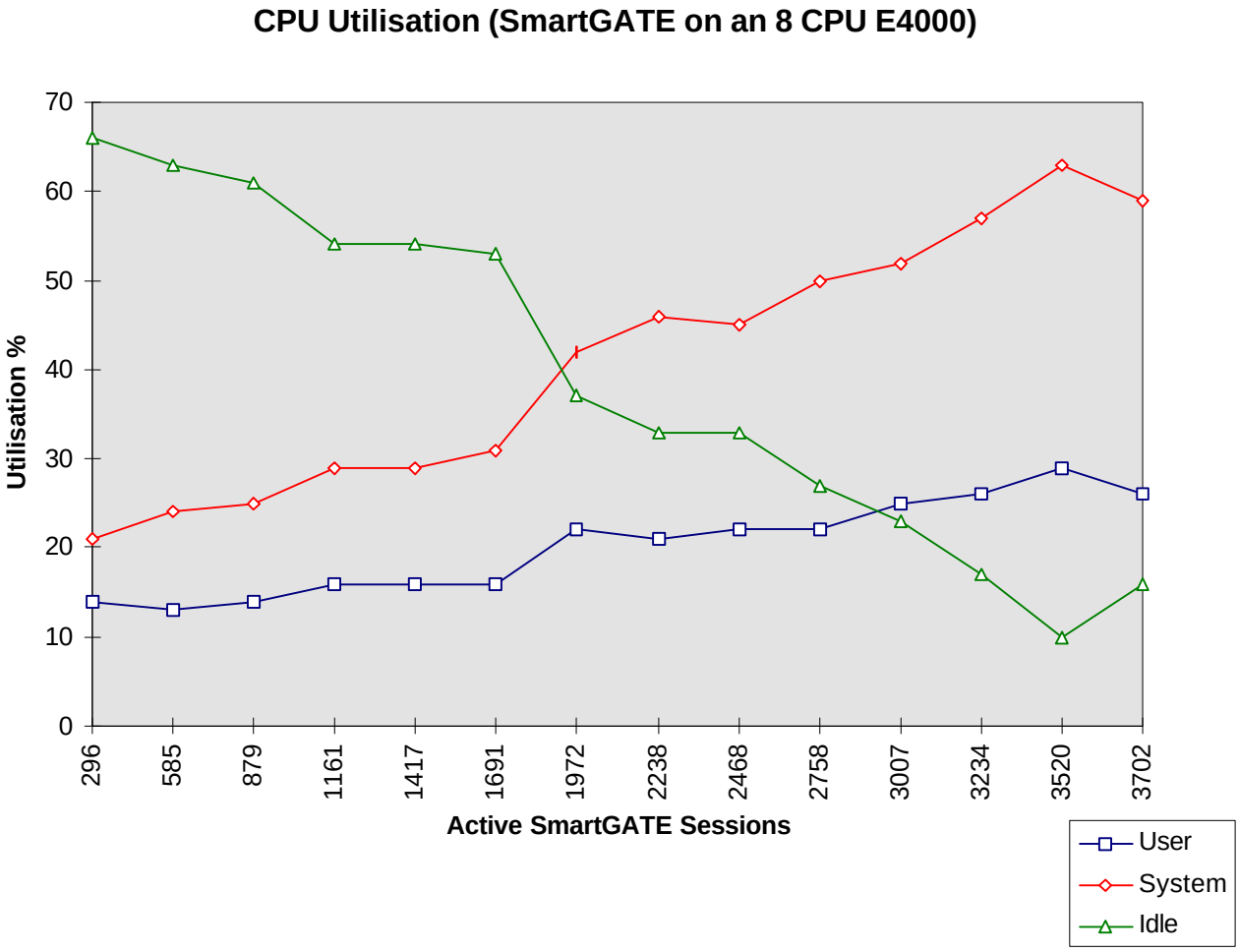

Figure 1, SmartGate CPU Usage on an Enterprise 4,000 Server

This series of tests demonstrates that a single SmartGate server is able to support at least 4,000 simultaneous active connections. It is likely that the same server could support significantly more connections in a live environment. This conclusion is based on observation of system performance during the tests and on the fact that the tests were limited to measuring CPU availability. CPU availability is not necessarily the primary factor determining the capacity of a system to support multiple SmartGate sessions, but is does provide a good indication of the capability of the system.

The figures used to produce this graph are taken from the peak CPU loading observed in each of a series of tests designed to exercise the SmartGate server under varying load conditions. Further observations showed that the peak CPU loading occurs at the start of each test run, when the test client system was trying to establish a large number of connections as quickly as possible. Once the desired number of connections had been made CPU usage drops to a lower level. Tests on the E4,000 showed that the CPU load dropped by almost half once the connections were made. Assuming that on an operational system the rate of arrival of new connections is more or less constant

then the upper limit for an E4,000 may be closer to 8,000 simultaneous connections. However this was not tested because of limitations in the test environment. A more detailed description of these tests is provided in appendix 1.

5. Appendix 1, Testing Methodology

In normal operation, SmartGate is implemented as a pair of proxy servers. The first proxy server is implemented within the SmartPass (client applications are configured to connect to that proxy) the second proxy server is implemented within the SmartGate server. Authentication and encryption operations take place between these two proxies. Once a correctly authenticated connection is received by the SmartGate server, the proxy component of the server relays the connection to an application server.

The goals of these benchmark tests were to determine the capacity and throughput of the SmartGate server. The SmartPass and the application server were not subject to test. To achieve these goals it was necessary to provide a mechanism to establish a large and variable number of connections from a series of SmartPasss. through a SmartGate server, to an application server. This could not be easily achieved with the standard SmartPass as it runs on Windows workstations. For this reason a modified test client was built for Unix.

The test client (*tcptest*) combines the functions of the application client and SmartPass in a single utility. The utility operates by making a variable number of connections (nConnections) through a SmartGate server and then sending a variable number of messages (nMessages) to a simple application server (*sgpload*) which simply echoed back the message. The connection to the SmartGate server includes the authentication and encryption functions of a normal SmartGate session. Early testing showed that the best performance from the test client and server was obtained if nConnections was fixed at 60 or below and nMessages was varied between 500 and 1500. Higher numbers of simultaneous connections to the SmartGate server were made by running multiple copies of the test utility on different local port numbers as shown in Figure 3. Each connection from multiple copies of the test client to the SmartGate server was made on port 2023, the standard port used for encrypting regular TCP connections (POP3, SMTP etc.), making the tests as close as possible to the eventual live configuration.

The majority of tests were carried out with multiple copies of the test client (tcptest) running on a Sparc Enterprise 3,000 and multiple copies of the test server running on a Sparc Enterprise 10,000. In most cases SmartGate server ran on an Enterprise 4,000 but tests were also carried out with SmartGate server on other systems.

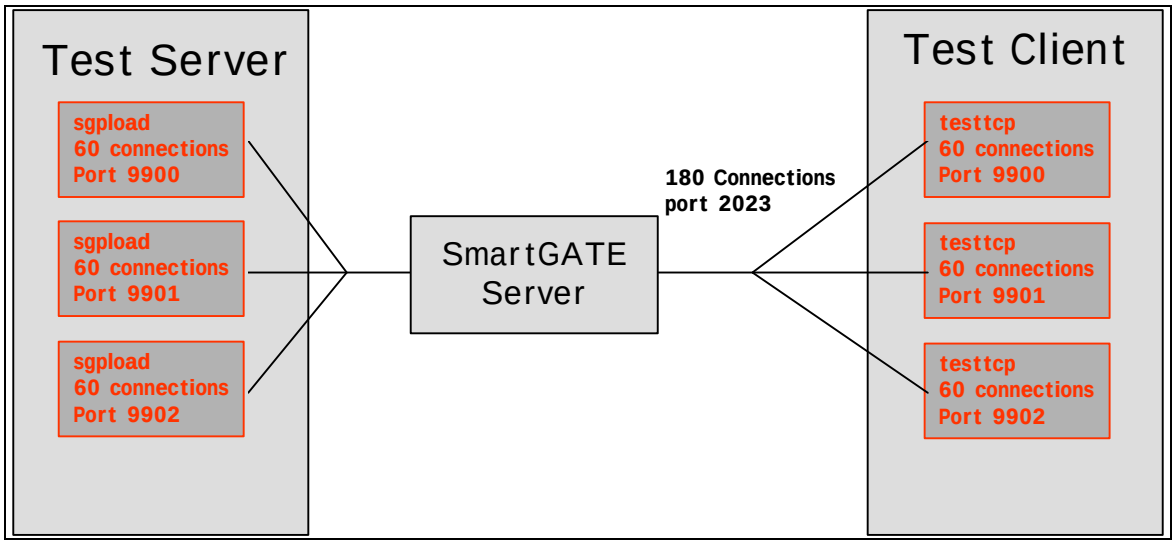

Figure 2, SmartGate Configuration for Benchmark Testing

For each test various system parameters (CPU load, free memory, network load etc.) were measured at 5-second intervals. Graphs were produced showing the peak (or lowest) value of each of these parameters plotted against the maximum number of open SmartGate connections. Many of these graphs are reproduced in this report. The maximum number of SmartGate connections was measured by successive use of the *ps* command and counting the number of active *sgate* processes. SmartGate server spawns an *sgate* process for each active connection.

Testing showed that peak load on the SmartGate server occurred when new connections were being established. When large numbers of connections were made the time taken to establish all the connections exceeded the time taken for the first copies of *tcptest* to run to completion. If this were allowed to happen then the maximum number of simultaneous connections would be reduced. This problem was solved by increasing the number of messages (nMessages) sent by each instance of *tcptest*. This increased the time required to complete the test ensuring that the number of connections processed by the SmartGate server reached the desired maximum. The disadvantage of this approach is that it makes comparisons of the time taken to complete a test invalid. However monitoring the CPU availability of the systems running the test client server showed that the total time to complete a test was dependent on CPU cycles in the test server platform and not on the SmartGate server.