Efficiency of the GSI Secured Network Transmission*

Bartosz Baliś[1,2], Marian Bubak[1,2], Wojciech Rząsa[3], and Tomasz Szepieniec[2]

[1] Institute of Computer Science, AGH, al. Mickiewicza 30, 30-059 Kraków, Poland
{balis,bubak}@uci.agh.edu.pl
[2] Academic Computer Center – CYFRONET, Nawojki 11, 30-950 Kraków, Poland
t.szepieniec@cyf-kr.edu.pl,
[3] Rzeszów University of Technology, Wincentego Pola 2, 35-959 Rzeszów, Poland
wrzasa@prz-rzeszow.pl
phone: (+48 12) 617 39 64, *fax:* (+48 12) 633 80 54

Abstract. The security of information transmitted over the computer networks is a frequently raised problem. The more information is transmitted and the more important it is, the more important security issues become. Network protocols used in the Internet and local networks were not designed to meet high security demands of current applications. Though cryptography algorithms enabled security of data transfer over insecure network protocols, they introduce additional overhead. Cryptography became important part of the security policy of Grid systems. In this paper, we perform experiments to estimate the overhead introduced by this solution. The obtained results are of great importance for the OCM-G – Grid enabled monitoring system.

Keywords: Grid, security, GSI, application monitoring, OCM-G

1 Introduction

The Grid concept proposed by Foster, Kesselman and Tuecke [8] was designed to enable coordinated resource sharing between distributed entities. In a grid system, no existing relationships or trust between the entities are assumed. There are also no single control points, though the sharing should be strictly controlled. Available resources (computing capability, disk capacity, etc.) are shared by means of the existing network infrastructure provided by the Internet, thereby a global distributed heterogenous system is created and may be used by different people for various purposes. In addition, the shared resources can be more efficiently exploited.

Grid applications are complex, therefore tools facilitating their development process are required [4]. The OCM-G application monitoring system is designed as an agent between such tools and application processes running on nodes belonging to distributed Grid sites [1]. The OCM-G is designed as a distributed

* This work was partly funded by the European Commission, project IST-2001-32243, CrossGrid [6]

M. Bubak et al. (Eds.): ICCS 2004, LNCS 3036, pp. 107–115, 2004.

and decentralized system, thus scalability required in the Grid environment is achieved. Monitoring system consists of two parts – permanent, handling multiple applications of numerous users and transient, belonging to the monitored application owner. The OCM-G works in on-line monitoring mode to address the requirements concerning delivery time between the user and the application processes.

Security issues are essential for the OCM-G as it supports multiple users and may control applications processes. It is obvious that the monitoring system should not lower security of the site and, at the same time, the solutions introduced to achieve the security should not degrade the scalability of the monitoring system [2].

Application of security aspects to network communication results in noticeable overhead. However as we mentioned above security cannot be omitted while considering the system as OCM-G despite of the strict demands concerning transmission time in on-line monitoring system. Therefore we decided to perform tests in order to estimate unwanted effects connected with security policy. However the experiments were designed to fit OCM-G realities, we believe the results can be useful for developers of the other Grid systems. Before detailed description of our tests we briefly introduce GSI that is security solution designed for the Grid systems.

2 GSI – Security Solution for the Grid Environment

The Grid concept assumes the use of existing network infrastructure to perform communication between resources and users. However, this implies some problems, security being one of the most important ones.

The security requirements concerning network data transmission can be divided into the following aspects.

- Authentication: the peers of the connection should be identified upon connection establishment.
- Authenticity and integrity of transmitted information should be ensured in order to avoid accidental or deliberated alteration.
- Confidentiality of transmitted data prevents eavesdropping.

Vulnerabilities of protocols used in the Internet or local networks are well known. Primary threats related to network transmission are briefly described below.

Sniffing or *eavesdropping* is possible in some low level communication protocols. It is a significant threat to confidentiality of the transmission. *Spoofing* is possible for each protocol commonly used in the Internet. Depending on the protocol it allows to impersonate host or makes attacker capable of deceiving authentication methods based on the source address of the packet or even allows a third host to become an agent between two other hosts, and fully control connections. Different varieties of spoofing threaten all aspects of secure data transmission. *Session take over* (or session hijacking) allows an attacker to intercept an already established TCP/IP session. Since the authentication is usu-

ally performed only on initialization of a connection this is significant threat to authenticity of transmitted data [3,5].

One may conclude that security problems cannot be solved by the means provided by network protocols. Therefore, cryptography algorithms [11] are applied in order to ensure the desired level of security. Asymmetric cryptography is used while secure connection establishment in order to perform reliable authentication and exchange symmetric keys between the peers. Thereafter efficient symmetric algorithms are used to encrypt transmitted data. Authenticity and integrity are ensured by the use of digital signature that is computed for each fragment of transmitted data. TLS is widely used example of protocol that can be used to securely transfer data over insecure networks [7].

Security of the Grid applications is also achieved by the use of cryptography. However the issues that we encounter in Grid environments are significantly more complex and cannot be solved by any existing protocol. Therefore, *Global Grid Forum Grid* Security Working Group [10] was formed in order to define Grid security requirements and find appropriate solutions. Since there are various existing concepts and well tested security solutions addressing different security issues, the GSI WG decided to develop the security solution for the Grid systems on the basis of existing ones. Thus, cryptography is used to ensure security in the Grid environment.

3 Tests

Applying cryptography algorithms to data transmission satisfies security requirements of the distributed systems. However, though these algorithms are becoming more and more efficient, they still introduce a significant overhead. In this section, we present results of tests performed in order to estimate this overhead. We describe three tests to evaluate resource consumption and other undesirable effects of cryptographic algorithms in different stages of the connection.

The security requirements of different Grid applications may vary. Therefore we performed the tests for different security levels applied: **CLEAR** – no security mechanisms were enabled, **AUTH** – authentication and authorization were performed upon connection establishment, **PROTECT** – authenticity and integrity of the transmitted data via digital signatures were ensured, **ENCRYPT** – transmitted data were encrypted. Note that the subsequent security level include all aspects of the previous ones.

We found it convenient to use two different execution environments for two kinds of tests. Experiments concerning the transmission overhead were carried out on slower machines, therefore it was easier to observe the CPU time usage. DoS vulnerability tests required possibly large cluster of machines, while high computing capability did not present a problem.

All test programs were implemented using `Globus_IO` – a communication library being part of the *Globus Toolkit 2* [9]. `Globus_IO` implements the GSI security solutions and is designed specifically for the Grid systems.

3.1 Transmission Overhead Test

The first experiment we performed in order to estimate the transmission overhead related to secure data transfer. The aim of this test is to evaluate the overhead resulting from the security mechanisms applied, depending on the security level ensured while data transmission.

Implementation and Testbed. In order to carry out the test we have implemented two programs `sender` and `responder`. The `sender` was designed to initialize network connection, transmit data over the network and receive them back. The `responder` was to receive data and transmit them back to the `sender`.

The test was performed on a PC cluster. The sender process was executed on an Intel Celeron 300 MHz machine, the responder on an Intel PIII 600 MHz one. The hosts were connected with 100Mb switched LAN.

Experiment and Results. The measurement consists in transmitting data through the network between the `sender` and the `responder`. The `sender` measured CPU time of the two-way transmission. The test was carried out for different quantities of 100-bytes packets with different security levels applied. The results are presented on Fig. 1.

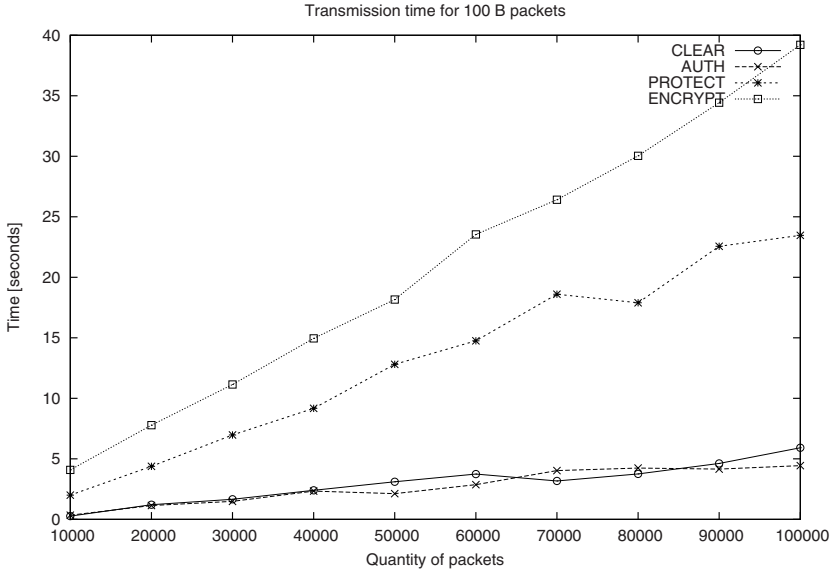

Fig. 1. Results of the security overhead test

We may notice a linear relationship between CPU time and quantity of packets for all security levels, however, for higher security levels the CPU time increases faster. In order to estimate the overhead resulting from the proposed

solution, we present an average transmission time for discussed security levels (see Fig. 1).

3.2 Transmission Time and Packet Size

The previous test was performed for a fixed size of the packet. Now we will study the dependency between transmission time and the size of transmitted packet. We should expect that it is more efficient to transmit data in large packets.

For this test we used the `sender` and the `responder` processes configured as in the previous one. We performed two-way transmission of 10,000 packets of different size, and measured CPU time consumption. In order to compare obtained results, we computed average CPU time required to transmit 100B in each size of the packet. The results of the experiment are presented on Fig. 2.

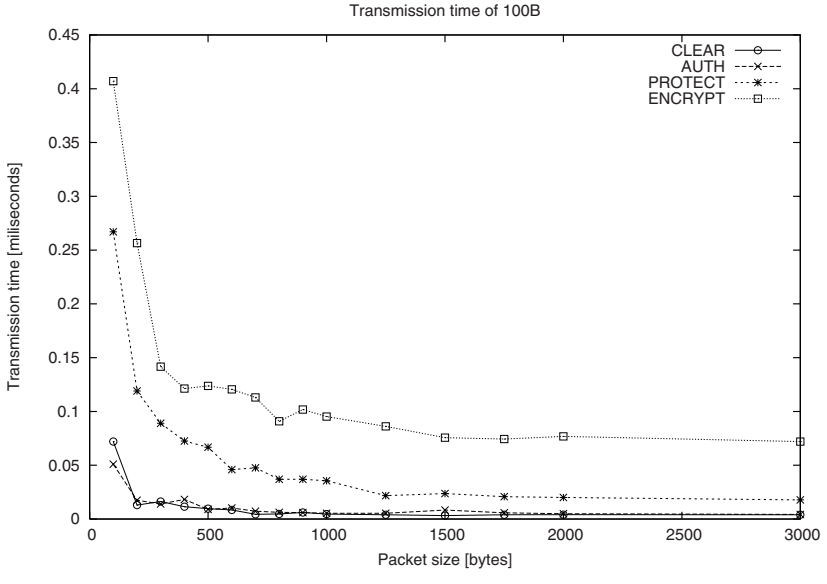

Fig. 2. Size of the packets and CPU time consumption for different security levels

3.3 Suspectibility to Denial of Service Attack

It is widely known that the establishment of a secure connection is resource consuming. After connection establishment transmitted data is encrypted with symmetric cryptography algorithms, however, before parameters of the connection are negotiated numerous asymmetric-cryptography operations should be performed. The asymmetric algorithms use longer keys and more complex mathematical operations are performed therefore they are less efficient than symmetric ones and consume significantly more resources.

From above considerations we may conclude that servers may not be able to simultaneously establish numerous secure connections. Moreover the port listening for secure connections may be easily used as a target for *Denial of Service* attack.

The aim of the test described below is to estimate vulnerability of the secure connections server as regards DoS attack. We try to find the practical limit of secure connections that can be handled by the server, verify the results of server overload and estimate number of connections that can be safely handled.

Implementation and Experiment Environment. In order to perform the experiment we implemented two short programs: `client` and `server`. The `server` was listening for connections, and after a connection arrived, it logged the time of the arrival. Then it tried to accept the connection and logged the time of establishment or failure. After the `client` started it waited until specified clock time and tried to establish connection with the specified host on specified port. The start time and connection establishment or failure time were logged. In this scenario, the `client` was the intended attacker while the server was the "victim" of this attack.

The experiment was performed on another PC cluster with machines equipped with two 2.4 GHz Intel Xeon processors and 1GB RAM each. Eighteen machines connected with 100 Mb switched LAN were involved in the test.

Description of Experiment and Results. The experiment consists in simulated Denial of Service attack that was performed by the `clients` to the `server`. The connections to the `server` running on one of the machines were requested from the other seventeen hosts. On the hosts we spawned numerous `client` processes. Each process tried to establish connection with the `server` located on specified node. The clocks of particular nodes were synchronized with the NTP protocol [12], thus we were able to achieve the required accuracy while triggering the `clients`.

During the experiment we increased the number of `client` processes on each node to reach maximum number of connections incoming in one second that can be handled by the `server`. The test was performed for the connections with different security levels applied. We present maximum number of connections that were requested in one second and properly established (see: Tab. 1). We have observed that in case of the connection failure `client` returned *Connection timed out* error.

We did not provide results of a DoS attack for *clear text* transmissions since a successful DoS attack using raw TCP protocol may strongly affect server's availability and we did not perform such a test. We only present results of a test which shows that *clear text* transmissions are significantly more efficient that secured (see: Tab. 2).

Table 1. Results of the simulated DoS attack

Security level	Connections				
	Requested		Established		Failed
	Overall	In 1 second	Overall	In 1 second	
AUTH	901	**899**	**881**	30	20
PROTECT	901	**894**	**871**	30	30
ENCRYPT	901	**896**	**897**	30	4

Table 2. TCP connections established in one second

Security level	Connections				
	Requested		Established		Failed
	Overall	In 1 second	Overall	In 1 second	
CLEAR	1700	**1692**	**1700**	1691	0

4 Conclusion

The first presented experiment has shown that the average CPU time required to perform data transmission with integrity checking is over four times greater, than the time required for the *clear text* transmission. Transferring data via encrypted connections requires even more resources: the average CPU time for this connection is over seven times greater than for insecure ones. However, we should realize that the CPU time required for the most resource consuming connection is in the order of 1/10 milliseconds even though we did not perform the experiment on an extremely fast hardware. Thus, even the connection that requires the most computing resources should not cause significant overhead, neither in the CPU consumption nor in the delay of message delivery.

However, while designing the security policy of a system we should consider which security level is really required for data transfer. From the results of the first test we can see that the differences in CPU time consumption between particular levels of security is significant. Therefore, it is desirable to restrict the security aspects only to those which are really required by the system.

Considering the results of the second test we may conclude that the size of the transmitted packet is significant for the transmission efficiency. The average CPU time required to transmit 100 bytes significantly decreases with packet size increasing to the size of 1.5 kilobytes. Thus, it appears to be more efficient to transmit small amount of large packets than huge amount of small packets. This is also true for the raw TCP connections, however, this factor seems to be more significant for the secured ones.

The establishment of a secure connection is an exceptionally resource consuming process. Therefore, secure connections servers can handle fewer connections than those which use raw TCP sockets. However, we can conclude from the results of the third test that the amount of connections which can be handled by such a server in one second is large enough to respond requirements of the OCM-G. Also, we have found important the fact that in our test, the client that

could not establish a connection with an overloaded server received the *connection timed out* network error. Thus it was possible to conclude the reason for which the connection could not be established.

5 Summary and Future Work

In this paper, we have shown the results of three experiments which tested the efficiency of different aspects of connection secured with cryptographical algorithms. We have evaluated the overhead resulting from secure data transfer as well as connection establishment.

It can be seen from the presented tests that applying security mechanisms to secure network connections results in significant overhead. The increase of CPU time required to transmit protected information in comparison to clear data transfer, as well as increased vulnerability of secured connections to the DoS attack cannot be passed over. Security offered by asymmetric cryptography algorithms results in significant resource consumption.

For the reasons described above, security mechanisms applied to data transmission in a system should always match the real security requirements. It seems to be useful to introduce optional lower security level for the less vulnerable network connections whenever it is permitted by the system security policy, especially when we desire a highly efficient data transmission.

On the basis of the results we have presented we can conclude, the overhead resulting from the security aspects applied to network connections are acceptable for the OCM-G as well as increased vulnerability to the DoS attack. However to make our results widely useful it seems to be necessary to work out an analytic model, that would make possible to estimate the overhead for the other, similar cases.

References

1. Baliś, B., Bubak, M., Funika, W., Szepieniec, T., and Wismuüller, R.: An Infrastructure for Grid Application Monitoring In: D. Kranzlmueller, P. Kacsuk, J. Dongarra, J. Volker (Eds.) Recent Advances in Parallel Virtual Machine and Message Passing Interface, Proc. 9th European PVM/MPI Users' Group Meeting, Linz, Austria, 2002, LNCS 2474, pp 41-49
2. Baliś B., Bubak M., Rząsa W., Szepieniec T., Wismüller R.: Security in the OCM-G Grid Application Monitoring System. In proc. of 5th International Conference on Parallel Processing and Applied Mathematics, September 2003.
3. Bellovin, S.: Security Problems in the TCP/IP Protocol Suite. Computer Communication Review **19** (2) 1989 pp 32-48
`http://www.research.att.com/~smb/papers/ipext.ps`
4. Bubak, M., Funika, W., and Wismüller, R.: The CrossGrid Performance Analysis Tool for Interactive Grid Applications. In: D. Kranzlmueller, P. Kacsuk, J. Dongarra, J. Volker (Eds.) Recent Advances in Parallel Virtual Machine and Message Passing Interface, Proc. 9th European PVM/MPI Users' Group Meeting, Linz, Austria, 2002, LNCS 2474, pp 50-60

5. Bellovin S.: Defending Against Sequence Number Attacks. RFC 1948.
6. The CrossGrid Project. http://www.eu-crossgrid.org
7. Dierks, T., Allen, C.: The TLS Protocol Version 1.0. RFC 2246.
8. Foster, I., Kesselman, C., Tuecke, S.: The Anatomy of the Grid. International Journal of Supercomputer Applications **15** (3) 2001
9. Globus Alliance homepage: `http://www.globus.org`
10. GGF GSI Working Group: `http://www.ggf.org/security/gsi/index.htm`
11. Menezes, A., van Oorschot, P., Vanstone, S.: Handbook of Applied Cryptography. CRC Press, 1996
    `http://www.cacr.math.uwaterloo.ca/hac/`
12. Network Time Protocol project homepage `http://www.ntp.org/`