Scalable Cryptographic Authentication for High Performance Computing

Andrew Prout, William Arcand, David Bestor, Chansup Byun, Bill Bergeron, Matthew Hubbell, Jeremy Kepner, Peter Michaleas, Julie Mullen, Albert Reuther, and Antonio Rosa
MIT Lincoln Laboratory, Lexington, MA 02420
{aprout, warcand, david.bestor, cbyun, bbergeron, mhubbell, kepner, pmichaleas, jsm, reuther, antonio.rosa}@ll.mit.edu

I. Introduction

High performance computing (HPC) uses supercomputers and computing clusters to solve large computational problems. Frequently HPC resources are shared systems and access to restricted data sets or resources must be authenticated. These authentication needs can take multiple forms, both internal and external to the HPC cluster. A computational stack that uses web services among nodes in the HPC may need to perform authentication between nodes of the same job or a job may need to reach out to data sources outside the HPC.

Traditional authentication mechanisms such as passwords or digital certificates encounter issues with the distributed and potentially disconnected nature of HPC systems. Distributing and storing plain-text passwords or cryptographic keys among nodes in a HPC system without special protection is a poor security practice. Systems that reach back to the user's terminal for access to the authenticator are possible, but only in fully-interactive supercomputing where connectivity to the user's terminal can be guaranteed.

Point solutions can be enabled for these use cases, such as software-based role or self-signed certificates, however they require significant expertise in digital certificates to configure. A more general solution is called for that is both secure and easy to use. This paper presents an overview of a solution implemented on the interactive, on-demand LLGrid computing system [1,2,3] at MIT Lincoln Laboratory and its use to solve one such authentication problem.

II. Lincoln Laboratory Grid (LLGrid)

One of the main design requirements of the LLGrid system was to make HPC usage accessible to the entire Lincoln technical staff by making HPC systems as easy to use as a personal computer. In striving to toward this goal, we developed a system that takes advantage of the desktop development environments (i.e., file system access and integrated development environments like Matlab) with which most engineers and scientists are already familiar. At Lincoln we currently have over 500 users. The vast majority of these users utilize LLGrid in an interactive supercomputing manner. As depicted in Figure 1, interactive supercomputing involves large jobs that require answers in minutes or hours and cannot wait in a queue.

This work is sponsored by the United States Air Force under Air Force contract FA8721-05-C-0002. Opinions, interpretations, conclusions and recommendations are those of the author and are not necessarily endorsed by the United States Government.

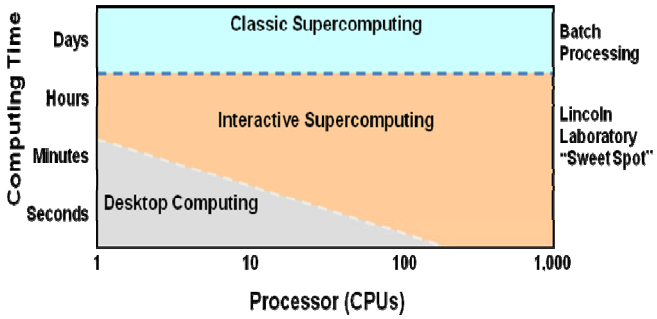

Figure 1: Interactive supercomputing vs. classic supercomputing and desktop computing. Interactive supercomputing jobs require answers in minutes or hours but cannot tolerate waiting in a queue. Classic supercomputing jobs take hours to days to execute and can tolerate waiting in a queue. Desktop computing jobs take minutes to run on a desktop (e.g. algorithm proof-of-concept).

III. Cryptographic Key Management on HPC Clusters

A challenge to any cryptographic key management system is to balance the need to make the cryptographic keys and their associated Public Key Infrastructure (PKI) certificates available for authorized use, but simultaneously ensure they are properly protected. The most widely used solution for this balance is a smart card that can be accessed using the PKCS#11 interface standard. While scaling hardware-based smart cards to a HPC environment is not feasible, a cryptographic library that acts similar to a smart card is practical. Several commercial and open source implementations exist, but these focus on different use cases that are tightly coupled with the software packages they ship with, and are not designed to protect cryptographic keys from export by their authorized user. [4,5]

We developed our own cryptographic library based on the PKCS#11 standard that connects to a daemon running as a separate user that stores and manages the cryptographic keys for all users on the system. This cryptographic key storage system is designed to allow the cryptographic keys to be manipulated only by system administrators. Unprivileged users can only access the cryptographic keys through the PKCS#11 interface and cannot export them to other locations. This provides security for the cryptographic keys while still allowing them to be used with common security protocols such as Transport Layer Security (TLS).

Access control to the cryptographic keys is based on the user context of the calling process. No additional authentication of the user is required allowing the cryptographic keys to be used in both attended and unattended processing. This solution does not provide protection against malicious activity by system administrators, compromise of cluster node, or malicious cryptographic key use by rogue processes running as the user. This solution does prevent malicious or unintended export of the cryptographic keys to other systems by unprivileged user processes.

We developed utilities to manage the cryptographic keys and PKI certificates for users across all nodes in the cluster simultaneously. Generation or renewal of cryptographic keys and PKI certificates across hundreds of nodes can be accomplished with a single command in parallel. This allows cryptographic keys and PKI certificates to be pre-generated by the system administrator for each user to ensure they are available on all nodes of the cluster.

Speed is also an important factor. Commercial smart cards are rated for approximately one operation per second. Reaching back to route cryptographic operations through the user's smart card would quickly bottleneck on this factor. By using a virtual smart card, and accepting the security limitations of that choice, we are only limited by the processing speed of the compute node, which is capable of over a hundred operations per second per CPU core, and scales linearly.

IV. Integration with Subversion

This system for cryptographic authentication to external systems was used to enable authenticated access to subversion repositories on the Lincoln Laboratory SourceForge (LLForge) subversion server without putting users' passwords at risk. The subversion client will normally either request to save the user's password in plain text within their home directory or prompt the user to enter their password for each subversion command. Recognizing that widespread storage of passwords in plain text is to be avoided and that frequent prompting of the user to type their password was disruptive, particularly for unattended processing, a better solution was sought.

The subversion server's TLS settings for HTTPS were reconfigured to request PKI certificate authentication and accept the LLGrid PKI certificate authority. The subversion client was configured to attempt authentication using certificates stored in the PKCS#11 cryptographic library interface discussed above, thus the LLGrid user identity running the svn client process could be proven to the LLForge subversion server without any user interaction or storage of plaintext passwords.

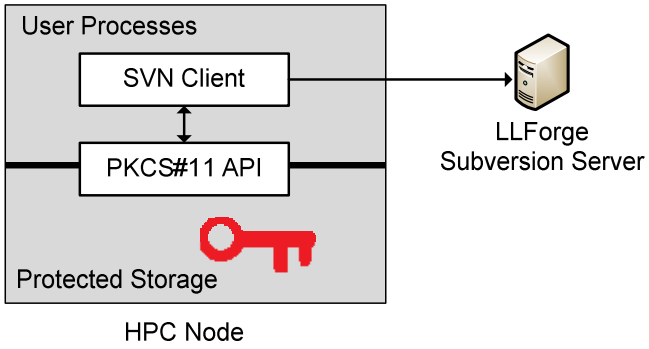

Figure 2: The subversion client running on a HPC node is able to make use of the cryptographic key through the PKCS#11 interface to prove the user identity to the subversion server, but unable to directly access the protected storage.

V. Results

We have built an infrastructure for management of cryptographic keys and PKI certificates that can be used for user authentication in HPC environments. This work is applicable to any situation where a scalable method of proving the identity of a HPC process owner to a remote system is needed and can be used to bootstrap trust for other HPC security implementations, such as the work by Foster et al, [6] or to enable the security features of web services as explored by van Engelen. [7] Additionally we have used this system to integrate the LLGrid HPC with the LLForge subversion server to balance ease of use with security.

Future work will investigate the applicability of this solution to other web services operating in HPC or connected to environments.

References

[1] N. Travinin Bliss, R. Bond, J. Kepner, H. Kim, and A. Reuther, "Interactive Grid Computing at Lincoln Laboratory," Lincoln Laboratory Journal, Vol. 16, Number 1, 2006.

[2] A. Reuther, B. Arcand, T. Currie, A. Funk, J. Kepner, M. Hubbell, A. McCabe, P. Michaleas, "TX-2500 – An Interactive, On-Demand Rapid-Prototyping HPC System," HPEC 2007, Lexington, MA, Sep. 2009.

[3] A. Reuther, J. Kepner, A. McCabe, J. Mullen, N.T. Bliss, and H. Kim, "Technical Challenges of Supporting Interactive HPC," In Proceedings of the High Performance Computing Modernization Office (HPCMO) Users Group Conference (UGC) 2007, Pittsburgh, PA, 18-22 June 2007.

[4] http://live.gnome.org/GnomeKeyring

[5] http://www.mozilla.org/projects/security/pki/nss/

[6] I. Foster, N. Karonis, C. Kesselman and S. Tuecke, "Managing security in high-performance distributed computations," Cluster Computing, Vol 1, Number 0, 1998.

[7] R. van Engelen, "Pushing the SOAP Envelope With Web Services for Scientific Computing." In proceedings of the International Conference on Web Services (ICWS), pages 346-352, Las Vegas, 2003.