Short Paper: Thor – The Hybrid Online Repository*

Timothy W. van der Horst and Kent E. Seamons
Internet Security Research Lab
Brigham Young University
{timv, seamons}@cs.byu.edu

Abstract

Mobile environments create significant challenges for secure credential repositories. We examine these challenges with respect to existing repository practices and produce a set of requirements that a repository must meet in order to cope with the harshness of a mobile environment. We also present Thor (The hybrid online repository), a system that fulfills these requirements. Thor leverages preexisting local and remote repositories and enhances their usability and security through virtual organization, credential identifier obfuscation, and password management.

1 Introduction

Protocols that make use of *digital credentials*[1] employ safeguards that protect these credentials during the life of the transaction. For the most part, these protocols do not concern themselves with the protection of these credentials outside the context of a transaction. This responsibility is delegated to a *secure credential repository*.

A repository is a place where objects can be stored, protected, and maintained. A bank vault is a classic example of a secure repository, as is an armored car. Although these examples have several similarities, they are also specifically tailored to their operational environment. The bank vault is designed to occupy a static location that provides good physical security, while the armored car is designed to be mobile and to operate mainly in a *hostile* environment.

Many different types of secure credential repositories have been created for use in a static environment. A mobile environment, however, invalidates many of the fundamental assumptions upon which these repositories were built.

This research proposes a set of requirements for secure credential repositories in a mobile environment and presents Thor, a repository that satisfies all of these requirements.

*This research was funded by NSF grants no. CCR-0325951 and IIS-0331707, and The Regents of the University of California.

[1] A credential is defined as a digital certificate and a private key.

2 Requirements for Secure Repositories in a Mobile Environment

In order to identify credential repositories that will successfully operate and survive in a mobile environment, the differences between this environment and its static cousin must be identified and examined. These disparities fall into three categories: physical security, connectivity, and manageability.

Physical Security The portability of mobile devices aggravates their risk to physical security threats. Mobile devices are very susceptible to theft because of their small size and usage outside a trusted domain. These devices can also be easily broken. Whether they are stolen, dropped, thrown, or smashed, their loss affects the availability of the information contained therein. This illustrates the first requirement for a secure credential repository in a mobile environment:

R1 The loss of the mobile device must not equate to a loss of a user's credentials.

Connectivity Devices in a static environment often assume the existence of a reliable network connection in order to access a repository. Due to the transient nature of mobile devices, they operate in a variety of connected and disconnected topologies and thus mandate the second requirement:

R2 A user-defined subset of credentials in the repository must be accessible regardless of the current communications topology.

Manageability A user can possess a variety of digital credentials. The management, replication, and distribution of these credentials between all of the user's devices, mobile or otherwise, is a critical task. An essential component of this task is the content of each credential. If the credential describes or identifies a specific device, then it should only

be used from that device. On the other hand, if a credential represents a person, that credential should be accessible by that person from any of her devices. This necessitates the final requirement:

R3 The repository must have an interface through which users may manage and maintain all their credentials and choose the set of credentials that is most appropriate for each device; Changes to the repository can then be propagated to all of the users' participating devices.

3 Related Work

Existing repositories fall into three classifications. *Local repositories* exist entirely on-device and usually take the form of an encrypted database (e.g., the Java KeyStore) or a set of encrypted files (e.g., PKCS#12 or PKCS#15). A local repository, because it resides entirely on-device, cannot satisfy Requirement **R1** by itself.

Remote repositories reside on an off-device server. Sandhu et al. [6] defines two types of remote repositories: virtual soft tokens and virtual smart cards. Virtual soft tokens (e.g., SACRED [5]) are a network-based storage solution. This type of repository is never involved in the use of any credential's private key. In the virtual smart card paradigm, the repository is always involved in the use of the private key. This is accomplished by one of two methods.

In the first method, the repository has permission to access the private key and performs signatures on behalf of the client using that key, just like a physical smart card. In the second method, the repository and the client device use the 3-key RSA algorithm to create a joint signature (e.g., NSD Security's Practical PKI [1]). A remote repository, because of its connectivity requirements, cannot satisfy Requirement **R2** by itself.

Hybrid repositories, first described in [8], are a union of local and remote repositories. The hybrid repository can act as a strict remote repository, a local repository (a full copy of credentials still resides in the remote repository), or a mix of the two. This configurable capability gives the user the flexibility to control the availability of his credentials in a mobile environment necessary to fulfill Requirements **R1** and **R2**.

Some currently available repositories integrate some concepts of a hybrid repository. For example, the RSA Keon Web PassPort [4] stores credentials on a remote server and uses a browser applet to emulate a smart card in the device's browser. Although this emulated smart card is similar to a local repository, it is never written to persistent local storage. The ability to create and use persistent local storage is an important attribute of the hybrid repository because it allows accessibility in the disconnected topology (see Requirement **R2**).

4 Thor

Rather than creating a new repository from scratch, Thor (The hybrid online repository) leverages existing local and remote repositories to create a new, hybrid repository. The underlying repositories in Thor are ignorant to the role they play in the hybrid repository and require no modification before incorporation into Thor.

In order for Thor to interact with a specific repository, a wrapper interface must be created. This interface exploits the fact that every repository provides the same basic functionality: the ability to store and retrieve credentials. Although the actual names and parameters of these operations vary, all repositories expose an API for the following three operations: 1) Put a credential in the repository with a unique identifier; 2) Get a credential from the repository based on a unique identifier; 3) Delete a credential in the repository based on a unique identifier.

Several other operations are required in order to setup and clean-up Thor's interaction with the underlying repositories. These necessary operations are handled during the instantiation and finalization of the wrapper. This interface allows Thor to be oblivious to specific repository implementations, enabling a user to very easily change Thor's underlying repositories.

Thor forces its underlying repositories to treat credentials as opaque objects. These objects are encrypted and are analogous to the safe deposit boxes stored in a bank vault. The security of the credentials is increased because only the user can access to them in their decrypted form. This is ideal when the remote repository is hosted by a third party.

Thor organizes its repositories into a tree structure. The root of the tree is called the *root repository node*. Either type of remote repository can be assigned the role of a root repository node, provided entire credentials can be retrieved from it and stored in a local repository for operation in the disconnected topology.

The other nodes in the tree structure are called *leaf repository nodes*. Since a leaf node resides on the mobile device it should be a local repository. The addition of new credentials to the repository or changes made to existing credentials via a leaf node, must pass through the root node to be made available to other leaves. There is no inter-leaf communication.

The root repository node provides an off-device backup suitable for the satisfaction of Requirement **R1** as well as a centralized location to manage and maintain credentials (essential to Requirement **R3**). The leaf repository node provides a local copy of a user-selected subset of credentials that are accessible in the disconnected topology, thus satisfying Requirement **R2**.

The *Central Management Utility* is a local software agent that provides the user interface to Thor as well as all

the logic needed to create and manage the hybrid repository. It connects to any of the user's participating repositories using the repository interface described above. This utility also provides a single location for a user to manage all his credentials and propagate changes to the user's participating devices, fulfilling the conditions dictated by Requirement **R3**.

Thor enables the integration of several usability and security enhancements through the use of *meta-data* that is encrypted and stored as a "credential" in the repository. This meta-data contains additional information (e.g., identifiers, modification dates, issuer, subject, etc.) about the credentials in the repository. By retrieving just the meta-data, queries can be performed on the credentials without further communication with the repository.

The first enhancement enabled by Thor is a method for creating a *virtual organization* of the credentials. This organization is stored persistently within the meta-data.

Although a credential is encrypted, its identifier can leak information about its contents. For example, the label: "Visa card credential" leaves little doubt as to the contents of the encrypted credential. A nonsensical label such as "JMVRYIKG/GGFBN25" reveals nothing about its contents. This is called *credential identifier obfuscation*.

The use of non-descriptive identifiers prevents a malicious insider or other attacker from focusing an attack on an obviously valuable target. To be effective this obfuscation should be done on all credentials. A mapping of meaningful credential identifiers to the obfuscated ones is easily stored with the meta-data. Since the meta-data itself is an attractive target, its identifier is also obfuscated.

An effective way to increase the security of the credentials in Thor is to encrypt each credential with a strong, high-entropy password. This makes each credential as hard to break as any other. Unfortunately, it is very hard for the average user to remember such passwords. This problem is compounded when multiple such passwords must be memorized, and usually results in the writing down of passwords in a non-secure place (e.g., a Post-it).

The idea of protecting many passwords with a single password is not a new concept. Many systems, e.g., Password Safe [2], have been designed to accomplish this very idea. The meta-data is an ideal location to store this password management information. Both password management and identifier obfuscation are implemented transparently such that the user only has to enter one password to gain access to the entire repository and only sees the descriptive credential identifiers.

Implementation We have created a prototype of Thor that uses a SACRED Credential Server [3] as its root repository node and a Java KeyStore as the leaf repository node. For more specific design and implementation details refer to [7].

5 Conclusions and Future Work

A mobile environment contains many hazards to digital credentials. The requirements specified in this paper enable the identification of repositories that cope with the physical harshness and connectivity issues of this environment.

Thor combines existing local and remote repositories into a single virtual repository. This hybrid repository, in addition to satisfying the requirements of a mobile environment, empowers users by automating several transparent features that enhance the protection of their sensitive digital credentials. These additional benefits are achieved without any modification to the underlying repositories.

Many repositories have not yet been incorporated into our system. Specifically, we have implemented an interface that allows a traditional online email account to act as a remote credential repository. This type of repository is very advantageous because it does not require the creation and management of a specialized server. Integration of this repository, as well as other repositories, is a definite priority.

Currently, Thor is limited to credential retrieval and storage. The integration of a virtual smart card (the emulation of a smart card in software similar to RSA's Keon Web PassPort) so that other applications could access and use Thor like a physical smart card is very desirable. This also ensures that a user's private keys never leave the protection of the repository. Building on this methodology, several virtual smart cards can be created to limit the applications that have access to specific credentials.

Usability is an important aspect of any secure system. Further evaluation of Thor's user and repository interfaces is needed.

References

[1] NSD Security Solutions Whitepaper. *Available at http://www.nsdsecurity.com*.

[2] Password Safe. *Available at http://www.schneier.com*.

[3] A Reference Implementation of SACRED. *Available at http://sacred.sourceforge.net*.

[4] RSA Keon Web PassPort Technical Overview. *Available at http://www.rsasecurity.com*.

[5] S. Farrell. Securely Available Credentials Protocol. *IETF Standards Track RFC 3767*, June 2004.

[6] R. Sandhu, M. Bellare, and R. Ganesan. Password-Enabled PKI: Virtual Smart Cards versus Virtual Soft Tokens. *PKI Research Workshop*, April 2002.

[7] T. W. van der Horst. Thor: The hybrid online repository. Master's thesis, Computer Science Deptartment, Brigham Young University, February 2005.

[8] T. W. van der Horst, T. Sundelin, K. E. Seamons, and C. D. Knutson. Mobile trust negotiation: Authentication and authorization in dynamic mobile networks. In *Conference on Communications and Multimedia Security*, September 2004.