Antigone: Policy-based Secure Group Communication System and AMirD: Antigone-based Secure File Mirroring System

Jim Irrer and Atul Prakash
Department of EECS
University of Michigan, Ann Arbor, MI 48109

Patrick McDaniel
AT&T Labs - Research
P.O. Box 971
Florham Park, NJ 07932-0971

Project web site: http://antigone.eecs.umich.edu

This demonstration gives examples of how the Antigone policy-based secure group communication system supports applications. The concept behind Antigone came from the applications whose security needs may vary, depending on operating environment, principals in a group, or the data being exchanged. For example, some applications may handle proprietary data that requires confidentiality, others may make press releases or other announcements that only require integrity and authentication. As another example, resource-plentiful infrastructures such as workstations connected via a high-speed local area network will be less concerned with the extra computational cycles and bandwidth required to ensure high-grade security, as compared to resource-constrained hand-held wireless devices where performance and battery-life concerns may limit the choice of security mechanisms.

Antigone supports a policy language, called Ismene. The Ismene policy language is capable of expressing highly flexible security policies and is extensible to support domain-specific requirements. Standard cryptographic algorithms such as DES, Blowfish, MD5, and SHA are supported. However, the Antigone and Ismene systems are designed in a modular fashion to allow other algorithms to be incorporated. Systems level features are also implemented modularly using software components that we term *mechanisms*. Mechanisms are instantiated from the policy specification to customize their use. For example, policy authors may currently choose either the NULL or SSL-based authentication mechanisms. While NULL requires no authentication, SSL does, and is customizable with a handful of parameters, such as number of retries and TCP port to use. Other mechanism types are Data Handler, Failure Detection and Recovery, Key Management, and Membership Monitoring. Again, the modularity of the Antigone

Proceedings of the DARPA Information Survivability Conference and Exposition (DISCEX'03)
0-7695-1897-4/03 $17.00 © 2003 IEEE

IEEE COMPUTER SOCIETY

Authorized licensed use limited to: Penn State University. Downloaded on April 28,2021 at 17:59:03 UTC from IEEE Xplore. Restrictions apply.

architecture allows more mechanisms to be added. Please refer to the project web site for more details.

AMirD: Antigone-based Directory Mirroring

AMirD provides an excellent example of capitalizing on Antigone's secure group capabilities. Functionally, the application allows secure directory mirroring, which can support needs such as mirroring web sites or software updates. Each site running AMirD has an authoritative list of files and directories for *export*, and a list of *import* files and directories it is interested in keeping up to date. To simplify this example, we will have a single exporter and multiple importers with a single import and export directory. The platforms we are using are a mixture of Linux and MS Windows machines.

AMirD performs three basic steps to complete a download. Initially, all interested parties join a control group. The exporter uses this group to announce the current state of its export directory.

Second, importers respond as to whether they are interested in files or not. Some importers may have been out of touch and missed an update, while other importers may have received all versions of file changes and are up to date. In the diagram below, importers 3 and 4 respond that they require the latest copy of file "foo".

In the third and final step, the exporter and interested importers form a download group. In this step the contents of the file is broadcast.

It should be noted that AMirD provides reliability for this data transfer, which under Antigone is the responsibility of the application. Antigone provides the specified level of authentication of sites in the group as well as data integrity/confidentiality.

Capitalizing on Antigone's flexible security support, the policy for AMirD can be configured to be dependent on the operating environment. Different policies can also be configured for control or download groups. The following tables enumerate the types of groups and their associated security guarantees. The type of group is dictated by an environment variable supplied to AMirD.

Policies vs. Scenarios

Control Group

	Local	**Mobile**	**Coalition**	**Site Mirroring**
Authentication	Null	OpenSSL	OpenSSL	OpenSSL
Key Mgmt.	KEK	AGKM - RC4	LKH	AGKM-blowfish
Failure Protection	None	None	Chained FP	None
Data Handler	Clear	sauth/integ/conf	sauth/integ/conf	sauth/integ

IEEE COMPUTER SOCIETY
Authorized licensed use limited to: Penn State University. Downloaded on April 28,2021 at 17:59:03 UTC from IEEE Xplore. Restrictions apply.

Download Group

	Local	Mobile	Coalition	Site Mirroring
Authentication	Null	OpenSSL	OpenSSL	OpenSSL
Key Mgmt.	KEK - Static	AGKM - RC4/Blowfish	KEK or AGKM - RC4/Blowfish	AGKM - Blowfish
Data Handling	conf	sauth/integ/ conf	sauth/integ/ conf	Sauth/integ

Secure Process Launch Daemon

The secure process launch daemon provides remote execution of processes and is used to start AMirD in this demonstration. Each of the members runs a small inetd-like daemon that uses Antigone. The daemon reads from standard input and sends it to the group, and reads from the group and sends it to standard output. This makes it ideal for piping between command line programs. To start AMirD, the exporter starts AMirD locally and sends a command to execute remote AMirD importers via this pipe. Each importer reads the command and executes it using a shell. As security is provided by Antigone, the system is immune to malicious or unauthorized commands. Many control and administration operations may be accomplished by updating files with systems such as AMirD. The Secure Process Launch Daemon allows additional range of commands to be remotely executed on multiple machines, such as remote software installation or re-booting a group of machines.

Acknowledgments

This work is supported in part by the Defense Advanced Projects Agency (DARPA) and Air Force Research Laboratory, Air Force Materiel Command, USAF, under agreement number F300602-00-2-0508. The U.S. Government is authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright annotation thereon. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of DARPA, the Air Force Research Laboratory, or the U.S. Government.

We are thankful for the contributions of several of our colleagues to the Antigone and AMirD system, including Alok Manchanda and Sharad Mittal.

IEEE COMPUTER SOCIETY

Authorized licensed use limited to: Penn State University. Downloaded on April 28,2021 at 17:59:03 UTC from IEEE Xplore. Restrictions apply.