Network Working Group
Request for Comments: 1805
Category: Informational

A. Rubin
Bellcore

June 1995

Location-Independent Data/Software Integrity Protocol

Status of this Memo

This memo provides information for the Internet community. This memo does not specify an Internet standard of any kind. Distribution of this memo is unlimited.

Abstract

This memo describes a protocol for adding integrity assurance to files that are distributed across the Internet. This protocol is intended for the distribution of software, data, documents, and any other file that is subject to malicious modification. The protocol described here is intended to provide assurances of integrity and time. A trusted third party is required.

Introduction

One problem with any system for verifying the integrity of a file is that the verifying program itself may be attacked. Thus, although users may be reassured by their software that a file has not changed, in reality, the file, and the verifier might have both changed. Because of this danger, a protocol that does not rely on the distribution of some special software, but rather, is based entirely on widely used standards, is very useful. It allows users to build their own software, or obtain trusted copies of software to do integrity checking independently. Therefore, the protocol described in this memo is composed of ASCII messages that may be sent using e-mail or any other means. There is an existing implementation, Betsi [1], that is designed this way. Betsi has been in existence since August, 1994, and is operational on the Internet. It can be accessed by sending e-mail to certify@bellcore.com with subject 'help', or via the world wide web at http://info.bellcore.com/BETSI/betsi.html.

The purpose of the proposed protocol is for authors to be able to distribute their files to users on the internet with guarantees of time and integrity, by use of a trusted third party. The protocol is divided into several phases:

I. Author registration
II. Author verification
III. File Certification
IV. File Distribution
V. File Integrity Verification

Phases I, III, IV, and V are defined in the protocol. Phase II is intentionally not defined. Author verification can be different for different applications, and the particular method chosen for phase II is identified in phases III and V. It is the hope that further Internet Drafts will describe the various possibilities for phase II. This memo describes the method for author verification in the Betsi system, and makes several recommendations.

Requirements

It is important that the integrity and time information be independent from the location of the file. Lowry [2] defines a syntax and protocols for location-independent objects. His system requires that end-users possess special software, and is still in the prototype stage. The protocol described in this memo has been implemented, and is already in wide-spread use across the Internet. It is simple, compact and easy to understand. The disadvantage of a very complex system is that users may not be inclined to trust the designers' claims if they cannot understand how it works.

Assumptions

The three entities in the protocol are Authors (A), Users (U), and a Trusted third party (T). The protocol described here is algorithm independent, and all of the messages are in ASCII. It is assumed that for each signature scheme used, there is a well-known verification key associated with T.

Any signature scheme may be used, as long as there is a standard ASCII representation of a digital signature. PGP [3] meets all of the above requirements, but it also requires encryption, and thus, export restrictions may deter some users. The DSS [4] is recommended, but some suspect that it contains a trapdoor [5] based on some results by Simmons [6]. It is also not clear that there is a standard for generating an ASCII signature using the DSS.

High level view

The protocol works as follows. In the first phase, authors request to register with the trusted third party, T. Any registered author can distribute files with integrity and time assurance. Time assurance means that there is a guarantee that a file existed at a given time. In the second phase, T somehow verifies the identity of an author who requests to register. Registration is not complete until this verification takes place.

To distribute a file, a registered author computes a cryptographic hash of the file, and sends it over an integrity protected channel to T. T then creates an object containing the hash, the current time, the name of the author, the name of the file, and some other information, seals the object, and returns it to the author. The author can then use the sealed object as a location-independent proof of the integrity and timeliness of the file.

Any user who obtains the file and the sealed object, can compute the cryptographic hash of the file, check the seal on the object, and verify that the object has not changed.

The trusted third party must maintain a widely available, dated, and signed, certificate revocation list (CRL). Users who access a file with a certificate must check that the CRL is current and complete, and that the certificate is not listed.

Author registration

In the first phase, authors request to register with the trusted third party, T. The author sends an ASCII message to T containing keywords followed by values. Some of the fields are optional, and are marked with a *. The values are represented with angle brackets < >.

```
  AUTHOR-NAME= <first m. last>
* AUTHOR-ORGANIZATION= <Company, school, etc.>
* AUTHOR-EMAIL= <e-mail address>
  AUTHOR-LOCATION= <city, state>
* AUTHOR-PHONE-1= <Home phone>
* AUTHOR-PHONE-2= <Work phone>
  SIGNATURE-SYSTEM= <name of signature system>
* MISC-FIELD-n= <Any number of additional fields can be defined here>
* AUTHOR-PUBLIC-KEY=
* <public key of author>
```

Each of the fields contains the keyword and the value on the same line, except for the public key. An ASCII version of the key is pasted on the line after the AUTHOR-PUBLIC-KEY keyword. The format

of this ASCII key will depend on the signature system used. The public key field is optional. The user may include his own, or one can be supplied by T during phase II. T responds with a message that the request was received, and that the user should wait for off-line verification. If a user receives this confirmation message, and he did not request to register, he knows that somebody may be attempting to register on his behalf.

Author verification

The trusted third party, T, must verify the identity of the author who sent the request message in phase I. The rest of the information in the request is also confirmed. This process takes place off-line. The method used is intentionally left open, but whatever technique is used must be identified in phases III and V.

In the Betsi implementation, T uses the phone company infrastructure. T calls directory assistance (1-xxx-555-1212) in the city of the author and asks for the author's number. Then, that number is called, and T asks the author to verify the information sent in the request. In particular, T insures that the author has registered his correct public key. Or, in some cases, T assigns a public key to the author. As Betsi is only operational in the United States, other mechanisms need to be in place for verifying identities of people internationally. Hopefully, standards for doing this will arise. The rest of the protocol is independent of whatever mechanism is used for off-line identity and public key verification.

File certification

Registered authors can obtain location-independent objects from the trusted third party, T, that vouch for the integrity and time of any file.

An author generates the following ASCII message and signs it with the signature key that corresponds to the public key that was registered.

```
  AUTHOR-NAME= <first m. last>
  HASH-FUNCTION= <md5,sha, etc.>
* FILE-LOCATION= <ftp site/directory>
  <list of hashes>
```

Each entry in the <list of hashes> consists of two mandatory fields and one optional one, as follows:

```
  <fixed-length hash of file> <name of file> <version number>
```

The <fixed-length hash of file> is a fixed-length hexadecimal value corresponding to the hash of the contents of the file. For MD5, the output is 32 hexadecimal digits. There is one space between the fields, and the name of the file contains no spaces. The <version number> is optional. The <list of hashes> contains at least one entry, and may contain as many as the author wants. The message is signed and sent to the trusted third party, T.

When T receives the request for file certification, he verifies the signature on the request and creates a location-independent certificate for the request. The certificate is signed by T, and contains the following information:

```
TRUSTED-PARTY= <identity of T>
AUTHOR-VERIFICATION-METHOD= <how authors are verified off-line>
AUTHOR-NAME= <first m. last>
AUTHOR-ORGANIZATION= <company, school, etc.>
HASH-FUNCTION= <md5,sha, etc.>
DATE= <date>
<list of hashes>
```

The <list of hashes> is the same as the one in the author's request. T signs the message and sends it to the author, who verifies the signature and the contents of the certificate. Note that the method for off-line author verification is included in the certificate.

File distribution

In the file distribution phase, the author distributes his file, along with the certificate from T. The file and certificate are location-independent. That is, the integrity and timeliness of the file can be verified independently from the location of the file and the certificate. This means that files can be distributed from insecure sites, and over insecure networks.

File integrity verification

The final phase is file integrity verification. A user obtains the public key of the trusted third party, T, from several independent sources, until he is convinced of its authenticity. The user then verifies the certificate for a file, and decides whether or not he trusts the method of off-line verification that was used by T. If so, then he extracts the name of the hash function in the certificate, and performs the hash function on the actual file. Finally, the user compares the hash of the file to the hash in the certificate. The user also checks the date in the certificate if he is concerned with this information. As a last step, the user checks the highly available certificate revocation list of T, to see if the current

certificate is listed. When all of this has concluded, if none of the assumptions of the system has been violated, then the user is assured of the integrity and timeliness of the file.

References

[1] Rubin, A., "Trusted Distribution of Software over the Internet", Internet Society Symposium on Network and Distributed System Security," pp. 47-53, 1995.

[2] Lowrey, J., "Location-Independent Information Object Security", Internet Society Symposium on Network and Distributed System Security," pp. 54-62, 1995.

[3] Zimmerman, P., "PGP User's Guide", 1992.

[4] National Institute for Standards and Technology, Digital Signature Standard (DSS), Federal Register 56(169), 1991.

[5] Schneier, B., "Applied Cryptography", ISBN 0-471-59756-2.

[6] Simmons, G., "The Subliminal Channels of the U.S. Digital Signature Algorithm (DSA)", Proceedings of the 3rd Symposium on: State and Progress of research in Cryptography, pp. 35-54, 1993.

Security Considerations

Security issues are discussed throughout this memo.

Author's Address

Aviel D. Rubin
Bellcore
Morristown, NJ 07960
USA

Phone: +1 201 829 5922
Fax: +1 201 829 2645
EMail: rubin@bellcore.com