Network Working Group
<draft-ietf-cat-kerb-key-derivation-00.txt>
Internet-Draft

M. Horowitz
Cygnus Solutions

November, 1996

Key Derivation for Kerberos V5

Status of this Memo

This document is an Internet-Draft. Internet-Drafts are working documents of the Internet Engineering Task Force (IETF), its areas, and its working groups. Note that other groups may also distribute working documents as Internet-Drafts.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as ``work in progress.''

To learn the current status of any Internet-Draft, please check the ``1id-abstracts.txt'' listing contained in the Internet-Drafts Shadow Directories on ds.internic.net (US East Coast), nic.nordu.net (Europe), ftp.isi.edu (US West Coast), or munnari.oz.au (Pacific Rim).

Distribution of this memo is unlimited. Please send comments to the <cat-ietf@mit.edu> mailing list.

Abstract

In the Kerberos protocol [RFC1510], cryptographic keys are used in a number of places. In order to minimize the effect of compromising a key, it is desirable to use a different key for each of these places. Key derivation [Horowitz96] can be used to construct different keys for each operation from the keys transported on the network. For this to be possible, a small change to the specification is necessary.

Overview

Under RFC1510 as stated, key derivation could be specified as a set of encryption types which share the same key type. The constant for each derivation would be a function of the encryption type. However, it is generally accepted that, for interoperability, key types and encryption types must map one-to-one onto each other. (RFC 1510 is being revised to address this issue.) Therefore, to use key derivcation with Kerberos V5 requires a small change to the specification.

For each place where a key is used in Kerberos, a ``key usage'' must be specified for that purpose. The key, key usage, and

encryption/checksum type together describe the transformation from plaintext to ciphertext, or plaintext to checksum. For backward

compatibility, old encryption types would be defined independently of the key usage.

Key Usage Values

This is a complete list of places keys are used in the kerberos protocol, with key usage values and RFC 1510 section numbers:

1. AS-REQ PA-ENC-TIMESTAMP padata timestamp, encrypted with the client key (section 5.4.1)
2. AS-REP Ticket and TGS-REP Ticket (includes tgs session key or application session key), encrypted with the service key (section 5.4.2)
3. AS-REP encrypted part (includes tgs session key or application session key), encrypted with the client key (section 5.4.2)

4. TGS-REQ KDC-REQ-BODY AuthorizationData, encrypted with the tgs session key (section 5.4.1)
5. TGS-REQ KDC-REQ-BODY AuthorizationData, encrypted with the tgs authenticator subkey (section 5.4.1)
6. TGS-REQ PA-TGS-REQ padata AP-REQ Authenticator cksum, keyed with the tgs session key (sections 5.3.2, 5.4.1)
7. TGS-REQ PA-TGS-REQ padata AP-REQ Authenticator (includes tgs authenticator subkey), encrypted with the tgs session key (section 5.3.2)
8. TGS-REP encrypted part (includes application session key), encrypted with the tgs session key (section 5.4.2)
9. TGS-REP encrypted part (includes application session key), encrypted with the tgs authenticator subkey (section 5.4.2)

10. AP-REQ Authenticator cksum, keyed with the application session key (section 5.3.2)
11. AP-REQ Authenticator (includes application authenticator subkey), encrypted with the application session key (section 5.3.2)
12. AP-REP encrypted part (includes application session subkey), encrypted with the application session key (section 5.5.2)

13. KRB-PRIV encrypted part, encrypted with a key chosen by the application (section 5.7.1)
14. KRB-CRED encrypted part, encrypted with a key chosen by the application (section 5.6.1)
15. KRB-SAVE cksum, keyed with a key chosen by the application (section 5.8.1)

16. Data which is defined in some specification outside of Kerberos to be encrypted using an RFC1510 encryption type.
17. Data which is defined in some specification outside of

Kerberos to be checksummed using an RFC1510 checksum type.

A few of these key usages need a little clarification. A service which receives an AP-REQ has no way to know if the enclosed Ticket was part of an AS-REP or TGS-REP. Therefore, key usage 2 must always

be used for generating a Ticket, whether it is in response to an AS-REQ or TGS-REQ.

There might exist other documents which define protocols in terms of the RFC1510 encryption types or checksum types. Such documents would not know about key usages. In order that these documents continue to be meaningful until they are updated, key usages 16 and 17 must be used to derive keys for encryption and checksums, respectively. New protocols defined in terms of the Kerberos encryption and checksum types should use their own key usages. Key usages may be registered with IANA to avoid conflicts. Key usages shall be unsigned 32 bit integers. Zero is not permitted.

Defining Cryptosystems Using Key Derivation

Kerberos requires that the ciphertext component of EncryptedData be tamper-resistant as well as confidential. This implies encryption and integrity functions, which must each use their own separate keys. So, for each key usage, two keys must be generated, one for encryption (Ke), and one for integrity (Ki):

```
Ke = DK(protocol key, key usage | 0xAA)
Ki = DK(protocol key, key usage | 0x55)
```

where the key usage is represented as a 32 bit integer in network byte order. The ciphertest must be generated from the plaintext as follows:

```
ciphertext = E(Ke, confounder | length | plaintext | padding) |
             H(Ki, confounder | length | plaintext | padding)
```

The confounder and padding are specific to the encryption algorithm E.

When generating a checksum only, there is no need for a confounder or padding. Again, a new key (Kc) must be used. Checksums must be generated from the plaintext as follows:

```
Kc = DK(protocol key, key usage | 0x99)

MAC = H(Kc, length | plaintext)
```

Note that each enctype is described by an encryption algorithm E and a keyed hash algorithm H, and each checksum type is described by a keyed hash algorithm H. HMAC, with an appropriate hash, is recommended for use as H.

Security Considerations

This entire document addresses shortcomings in the use of cryptographic keys in Kerberos V5.

Acknowledgements

I would like to thank Uri Blumenthal, Sam Hartman, and Bill Sommerfeld for their contributions to this document.

References

[Horowitz96] Horowitz, M., "Key Derivation for Authentication, Integrity, and Privacy", draft-horowitz-key-derivation-00.txt, November 1996. [RFC1510] Kohl, J. and Neuman, C., "The Kerberos Network Authentication Service (V5)", RFC 1510, September 1993.

Author's Address

Marc Horowitz
Cygnus Solutions
955 Massachusetts Avenue
Cambridge, MA 02139

Phone: +1 617 354 7688
Email: marc@cygnus.com