INTERNET-DRAFT
draft-ietf-cat-kerberos-err-msg-00.txt
Updates: RFC 1510
expires September 30, 1997

Ari Medvinsky
Matt Hur
Dominique Brezinski
CyberSafe Corporation
Gene Tsudik
Brian Tung
ISI

Integrity Protection for the Kerberos Error Message

0. 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).

The distribution of this memo is unlimited. It is filed as draft-ietf-cat-kerberos-pk-init-03.txt, and expires June xx, 1997. Please send comments to the authors.

1. Abstract

The Kerberos error message, as defined in RFC 1510, is transmitted to the client without any integrity assurance. Therefore, the client has no means to distinguish between a valid error message sent from the KDC and one sent by an attacker. This draft describes a method for assuring the integrity of Kerberos error messages, and proposes a consistent format for the e-data field in the KRB_ERROR message. This e-data format enables the storage of cryptographic checksums by providing an extensible mechanism for specifying e-data types.

2. Motivation

In the Kerberos protocol [1], if an error occurs for AS_REQ, TGS_REQ, or AP_REQ, a clear text error message is returned to the client. An attacker may exploit this vulnerability by sending a false error message as a reply to any of the above requests. For

example, an attacker may send the KDC_ERR_KEY_EXPIRED error message in order to force a user to change their password in hope that the new key will not be as strong as the current key, and thus, easier to break.

Since false error messages may be utilized by an attacker, a Kerberos client should have a means for determining how much trust to place in a given error message. The rest of this draft describes a method for assuring the integrity of Kerberos error messages.

3. Approach

We propose taking a cryptographic checksum over the entire KRB-ERROR message. This checksum would be returned as part of the error message and would enable the client to verify the integrity of the error message. For interoperability reasons, no new fields are added to the KRB-ERROR message. Instead, the e-data field (see figure 1) is utilized to carry the cryptographic checksum.

3.1 Cryptographic checksums in error messages for AS_REQ, TGS_REQ & AP_REQ

If an error occurs for the AS request, the only key that is available to the KDC is the shared secret (the key derived from the clients password) registered in the KDCs database. The KDC will use this key to sign the error message, if and only if, the client already proved knowledge of the shared secret in the AS request (e.g. via PA-ENC-TIMESTAMP in preauth data). This policy is needed to prevent an attacker from getting the KDC to send a signed error message and then launching an off-line attack in order to obtain a key of a given principal.

If an error occurs for a TGS or an AP request, the server will use the session key sealed in the clients ticket granting ticket to compute the checksum over the error message. If the checksum could not be computed (e.g. error while decrypting the ticket) the error message is returned to the client without the checksum. The client then has the option to treat unprotected error messages differently.

```
KRB-ERROR ::=  [APPLICATION 30] SEQUENCE {
        pvno       [0]  integer,
        msg-type   [1]  integer,
        ctime      [2]  KerberosTime OPTIONAL,
        cusec      [3]  INTEGER OPTIONAL,
        stime      [4]  KerberosTime,
        susec      [5]  INTEGER,
        error-code [6]  INTEGER,
        crealm     [7]  Realm OPTIONAL,
```

```
        cname       [8]  PrincipalName OPTIONAL,
        realm       [9]  Realm,          --Correct realm
        sname       [10] PrincipalName, --Correct name
        e-text      [11] GeneralString OPTIONAL,
        e-data      [12] OCTET STRING OPTIONAL
}
```

Figure 1

3.2 Format of the e-data field

We propose to place the cryptographic checksum in the e-data field. First, we review the format of the e-data field, as specified in RFC 1510. The format of e-data is specified only in two cases [2]. "If the error code is KDC_ERR_PREAUTH_REQUIRED, then the e-data field will contain an encoding of a sequence of padata fields":

```
METHOD-DATA ::=  SEQUENCE of PA-DATA
PA-DATA ::= SEQUENCE {
            padata-type    [1] INTEGER,
            padata-value   [2] OCTET STRING
}
```

The second case deals with the KRB_AP_ERR_METHOD error code. The e-data field will contain an encoding of the following sequence:

```
METHOD-DATA ::=  SEQUENCE {
                 method-type    [0] INTEGER,
                 method-data    [1] OCTET STRING OPTIONAL
}
```

method-type indicates the required alternate authentication method.

It should be noted that, in the case of KRB_AP_ERR_METHOD, a signed checksum is not returned as part of the error message, since the error code indicates that the Kerberos credentials provided in the AP_REQ message are unacceptable.

We propose that the e-data field have the following format for all error-codes (except KRB_AP_ERR_METHOD):

```
E-DATA ::=  SEQUENCE {
            data-type    [1] INTEGER,
            data-value   [2] OCTET STRING,
}
```

The data-type field specifies the type of information that is carried in the data-value field. Thus, to send a cryptographic checksum back to the client, the data-type is set to CHECKSUM, the data-value is set to the ASN.1 encoding of the following sequence:

```
Checksum  ::=  SEQUENCE {
```

```
                cksumtype   [0] INTEGER,
                checksum    [1] OCTET STRING
    }
```

3.3 Computing the checksum

After the error message is filled out, the error structure is converted into ASN.1 representation. A cryptographic checksum is then taken over the encoded error message; the result is placed in the error message structure, as the last item in the e-data field. To send the error message, ASN.1 encoding is again performed over the error message, which now includes the cryptographic checksum.

3.4 Verifying the integrity of the error message

In addition to verifying the cryptographic checksum for the error message, the client must verify that the error message is bound to its request. This is done by comparing the ctime field in the error message to its counterpart in the request message.

4. E-DATA types

Since the e-data types must not conflict with preauthentication data types, we propose that the preauthentication data types in the range of 2048 and above be reserved for use as e-data types.

We define the following e-data type in support of integrity checking for the Kerberos error message:

CHECKSUM = 2048 -- the keyed checksum described above

5. Discussion

5.1 e-data types

The extension for Kerberos error messages, as outlined above, is extensible to allow for definition of other error data types. We propose that the following e-data types be reserved:

KDCTIME = 2049
The error data would consist of the KDCs time in KerberosTime. This data would be used by the client to adjust for clock skew.

REDIRECT = 2050
The error data would consist of a hostname. The hostname would indicate the authoritative KDC from which to obtain a TGT.

5.2 e-data types vs. error code specific data formats

Since RFC 1510 does not define an error data type, the data format must be explicitly specified for each error code. This draft has proposed an extension to RFC 1510 that would introduce the concept of error data types. This would allow for a manageable set of data types to be used for any error message. The authors assume that the introduction of this e-data structure will not break any existing Kerberos implementations.

6. Bibliography

[1] J. Kohl, C. Neuman. The Kerberos Network Authentication Service (V5). Request for Comments: 1510

[2] J. Kohl, C. Neuman. The Kerberos Network Authentication Service (V5). Request for Comments: 1510 p.67

7. Authors

Ari Medvinsky <ari.medvinsky@cybersafe.com>
Matthew Hur <matt.hur@cybersafe.com>
Dominique Brezinski <dominique.brezinski@cybersafe.com>

CyberSafe Corporation
1605 NW Sammamish Road
Suite 310
Issaquah, WA 98027-5378
Phone: (206) 391-6000
Fax: (206) 391-0508
http:/www.cybersafe.com

Brian Tung <brian@isi.edu>
Gene Tsudik <gts@isi.edu>

USC Information Sciences Institute
4676 Admiralty Way Suite 1001
Marina del Rey CA 90292-6695
Phone: (310) 822-1511