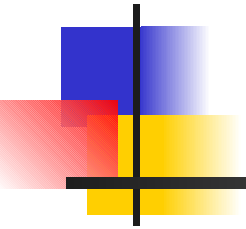
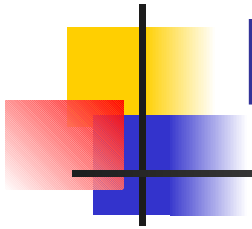


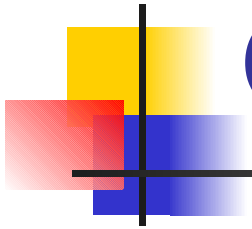
OTP Kerberos

Kerberos Working Group
IETF, Montreal
July 2006
WORKING DRAFT: 30 June 2006



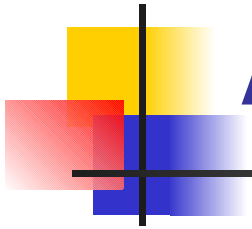
Background

- Proposal is to define extensions to Kerberos V5 (rfc4120) to support pre-authentication using an OTP
- Three main aims:
  - Allow an OTP to be used without a password in pre-authentication
  - Support both connected and disconnected tokens
  - Support multiple OTP algorithms
- Part of One-Time Password Specification (OTPS) series of documents on aspects of OTP usage and integration
- Inspired by expired I-D draft-ietf-cat-kerberos-passwords-04 and draft-ietf-krb-wg-kerberos-sam-03



Comparison with Expired Draft

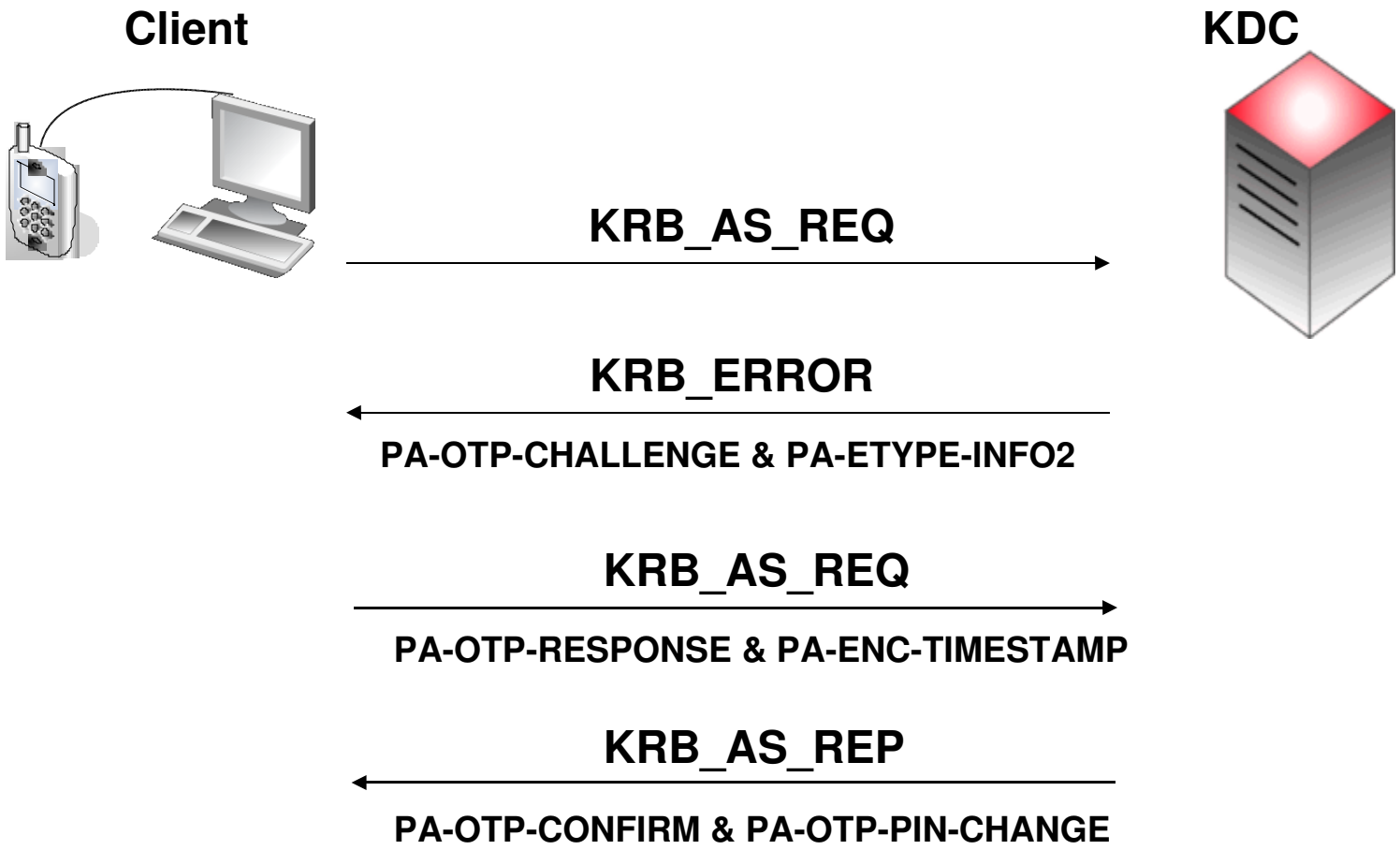
- Designed to support both connected and disconnected tokens
  - Previous proposal aimed at disconnected tokens.
  - Supported token type but not key identifier.
- Designed to support multiple OTP algorithms
  - Counter
  - Challenge-response
  - Time
  - Time + counter etc.
- Uses hardening value to harden OTP
  - Previous proposal recommends use of password with OTP to harden key.
  - New proposal is to combines OTP with hardening value using PBKDF2
- Current proposal only supports case where KDC can generate OTP
  - Cannot support basic S/Key
  - Use Ephemeral D-H to transport OTP value?
- Supports PIN change using password change extension (rfc3244)

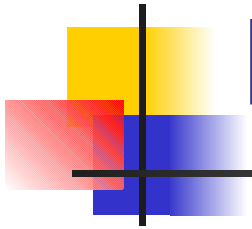


Approach

- Information on how to generate OTP sent in challenge from KDC
- Client uses OTP to generate encryption key
- Key used to encrypt timestamp as in standard password pre-auth
- OTP hardened using hardening value generated by KDC

Pre-authentication Exchange





Pre-authentication Exchange

- Client sends initial KRB_AS_REQ possibly containing password-based pre-auth.
- KDC responds with KRB_ERROR containing:
  - PA-OTP-CHALLENGE indicating how OTP is to be generated
  - PA-ETYPE-INFO2 indicating how key is to be generated
- Client generates OTP and uses it to generate encryption key
- Client sends second KRB_AS_REQ to KDC containing
  - PA-OTP-RESPONSE with information on how OTP was generated
  - PA-ENC-TIMESTAMP containing encrypted timestamp
- KDC validates pre-authentication data and returns KRB_AS_REP
  - PA-OTP-CONFIRM containing OTP hardening value
  - PA-ENC-PIN containing new PIN for the user

PA-OTP-CHALLENGE

```
PA-OTP-CHALLENGE ::= SEQUENCE {
                    flags
  ChallengeFlags
  otp-challenge[0] OCTET STRING
  OPTIONAL,                      otp-
  length    [1] INTEGER
  OPTIONAL,                      otp-
  service   [2] UTF8String
  OPTIONAL,                      otp-
  keyID     [3] OCTET STRING
  OPTIONAL,                      otp-
  algID     [4] INTEGER
  OPTIONAL                     }
```

PA-OTP-RESPONSE

```
PA-OTP-RESPONSE ::= SEQUENCE {
        iterationCount[0] INTEGER
  OPTIONAL,                    identifier
   [1] OCTET STRING OPTIONAL,
        otp-challenge [2] OCTET STRING
  OPTIONAL,                    otp-time
   [2] KerberosTime OPTIONAL,
        otp-counter   [3] OCTET STRING
  OPTIONAL,                    otp-format
   [4] OTPFormat    OPTIONAL,
        otp-keyID     [5] OCTET STRING
  OPTIONAL                     }
OTPFormat ::= INTEGER {
   decimal(0),
  hexadecimal(1),
  alphanumeric(2),
  binary(3)
}
```

Key Generation

- KDC MUST support at least one of the encryption types
  - aes128-cts-hmac-sha1-96
  - aes256-cts-hmac-sha1-96
- KDC's KRB_ERROR contains PA-ETYPE-INFO2 containing
  - etype,
  - salt
  - iteration count in the s2kparams
- Encryption key generated using as defined in RFC3962 using PBKDF2 but with addition of hardening value

```
tkey = random-to-key(PBKDF2(OTP, salt|hardening,
                    iteration_count, key_length))
key = DK(tkey, "kerberos")
```

Hardening Value

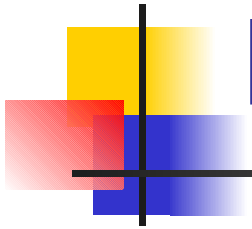
- Hardening value to be used by client sent by KDC in PA-OTP-CONFIRM in KRB_AS_REP
- Client stores hardening value associated with KDC
- If the client has a hardening value then an iteration count of 1 used
- Full iteration count used if no hardening value (e.g. first authentication)
- Identifier of value used included in PA-OTP-RESPONSE

PA-OTP-CONFRIM

```
PA-OTP-CONFIRM ::= SEQUENCE {

 identifier          OCTET STRING,
     encHardeningValue EncryptedData  --
 EncHardeningValue
 }

 EncHardeningValue ::= OCTET STRING SIZE
 (16..MAX)
```



PIN Change

- KDC can return PA-OTP-PIN-CHANGE in KRB_AS_REP
- Contents encrypted using current user key
- Can contain new PIN that user must use
- Can also instruct user that their PIN must be changed
- User PIN changed handled using ChangePasswdData from RFC3244

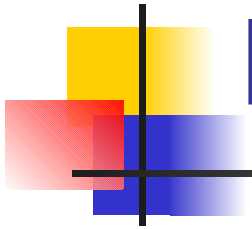
PA-OTP-PIN-CHANGE

```
PA-ENC-PIN      ::= EncryptedData --PA-
  ENC-PIN-ENC
PA-ENC-PIN-ENC ::= SEQUENCE {

   flags            PinFlags

   pin          [0] UTF8String OPTIONAL
                 minLength [1] INTEGER
   OPTIONAL                      maxLength [2]
   INTEGER     OPTIONAL                    }

PinFlags ::= KerberosFlags
-- systemSetPin (0)
```



Next Steps

- Solicit WG discussion on approach
- Resolve S/Key issue
  - Ephemeral D-H?
- Extend to support re-synchronizing of tokens