rfc4231

Opération Turn

Cette opération étendue inverse les rôles du client et du serveur pour des échanges de protocole dans la session, ou pour permettre à chaque parties d'agir en tant que client et serveur.

Requête Turn

L'opération Turn est définie par l'OID **1.3.6.1.1.19**. La requête Turn est une **ExtendedRequest** où **requestName** contient **1.3.6.1.1.19** et **requestValue** est une valeur **turnValue** encodée BER :

```
turnValue ::= SEQUENCE {
 mutual BOOLEAN DEFAULT FALSE,
 identifier LDAPString
}
```

Réponse Turn

La réponse est une **ExtendedResponse** où les champs **responseName** et **responseValue** sont absent. un **resultCode** à success indique qu'il est capable d'inverser la sessions.

Authentification

Le modèle d'authentification de cette extension assume une authentification séparée des parties dans chacun de leur rôle. Un échange Bind est attendue entre les parties et leur nouveau rôle pour établir les identités dans ces rôles.

Une fois le Turn complété, le partie répondant dans son nouveau rôle client a une association anonyme au partie initiant son nouveau rôle serveur. Si le turn est mutuel, l'association d'authentification du partie initiant est laissé intacte.

Le partie répondant peut établir son identité dans son rôle client en demandant et complétant une opération Bind.

La suite discute de divers scénarios d'authentification. **A** réfère au partie initiant (le client à l'origine) et **B** réfère au partie répondant (le serveur à l'origine)

Utilisation avec TLS et une authentification simple

```
A->B: StartTLS Request
B->A: StartTLS(success) Response
A->B: Bind(Simple(cn=B,dc=example,dc=net,B's secret)) Request
B->A: Bind(success) Response
```

```
A->B: Turn(TRUE,"XXYYZ") Request
B->A: Turn(success) Response
B->A: Bind(Simple(cn=A,dc=example,dc=net,A's secret)) Request
A->B: Bind(success) Response
```

Dans ce scénario, TLS est initié et A établis son identité avec B avant de Turn en utilisant un mécanisme DN/password. après le Turn, B établis son identité avec A.

Utilisation avec TLS et SASL EXTERNAL

```
A->B: StartTLS Request
B->A: StartTLS(success) Response
A->B: Bind(SASL(EXTERNAL)) Request
B->A: Bind(success) Response
A->B: Turn(TRUE,"XXYYZ") Request
B->A: Turn(success) Response
B->A: Bind(SASL(EXTERNAL)) Request
A->B: Bind(success) Response
```

Dans ce scénario, TLS est initié, et A établis son identité avec SASL avant de Turn. Après le Turn, B établis son identité avec A.

Utilisation de l'authentification mutuelle et SASL EXTERNAL

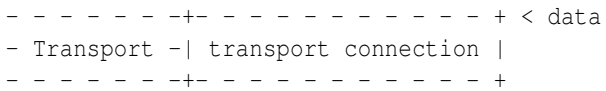
```
A->B: Bind(SASL(GSSAPI)) Request
<intermediate messages>
B->A: Bind(success) Response
A->B: Turn(TRUE,"XXYYZ") Request
B->A: Turn(success) Response
B->A: Bind(SASL(EXTERNAL)) Request
A->B: Bind(success) Response
```

Dans ce scénario, un échange d'authentification mutuel GSSAPI est complété entre A et B. après le Turn, B demande que A utilise une identité externe pour établir son identité d'authorisation LDAP.

Couches TLS et SASL

La table suivante décris la relation entre la couche du message LDAP, SASL, TLS et la connection de transport dans une session LDAP.

```
- - - - - - -+- - - - - - - - - - - - +
- - - - - - -| LDAP message layer _ |
- - - - - - -+- - - - - - - - - - - - + > LDAP PDUs
- - - - - - -+- - - - - - - - - - - - + < data
- - - - - - -| ____ SASL layer ____ |
- - - - - - -+- - - - - - - - - - - - + > SASL-protected data
- - - - - - -+- - - - - - - - - - - - + < data
- - - - - - -| _____ TLS layer ____ |
Application -+- - - - - - - - - - - - + > TLS-protected data
```



Cette extension n'altère pas la relation, ni ne supprime les restriction générales des couches TLS et SASL multiples. StartTLS est utilisé pour initier la négociation TLS. si TLS est déjà établis, StartTLS peut échouer.