Channel-Binding for HTTP Digest

Stefan Santesson
Microsoft
stefans@microsoft.com

Background

- Implemented by Microsoft
- Documented in draft-santesson-digestbind-01.txt
- http://tools.ietf.org/html/draft-santesson-digestbind-01
- Aimed to be published as Infromational

Design principles

- Tied to TLS end point binding (Hash of Server certificate)
- Protocol identification through nonce field: “+UpGrAdEd+v1” is added to server WWW-Authenticate Response header as well as the client Authorization Request Header.
- Adding directives to the HTTP Authorization Request Header from client to server using the auth-param extensibility.

New directives

```
hashed-directives   = "hashed-dirs" "=" <"> 1#token <">
service-name        = "service-name" "=" service-name-value
charset             = "charset" "=" "utf-8”
channel-binding     = <"> 32LHEX <">


service-name-value = serv-type "/" host [ "/" serv-name ]
serv-type          = 1*ALPHA
host               = 1*( ALPHA | DIGIT | "-" | "." )
serv-name          = host
```

- charset defined in RFC 2831 and Service name identical to digest-uri of RFC 2831.

hashed-dirs directive

- The names of the directives, which values are hashed and included in the cnonce (Sent by the client), provided as a quoted coma separated list.
- For version 1 (v1) of this specification, this directive MUST contain:

hashed-dirs = "service-name,channel-binding"

channel-binding directive

- Channel binding types definitions registered in IANA registry for RFC 5056

  http://www.iana.org/assignments/channel-binding-types/

- MUST contain a 32 byte (256 bit) end point channel binding value of type tls-server-end-point

tls-server-end-point

- **Description:** The hash of the TLS server's end entity certificate as it appears, octet for octet, in the server's Certificate message (note that the Certificate message contains a certificate_list, the first element of which is the server's end entity certificate.) The hash function to be selected is as follows: if the certificate's signature hash algorithm is either MD5 or SHA-1, then use SHA-256, otherwise use the certificate's signature hash algorithm. The reason for using a hash of the certificate is that some implementations need to track the channel binding of a TLS session in kernel-mode memory, which is often at a premium.