Network Working Group
Internet-Draft
Expires: January 14, 2010

N. Neumann
X. Fu
University of Goettingen
July 13, 2009

Diameter Application for Authentication and Authorization in Web Applications
draft-neumann-dime-webauth-01

Status of this Memo

Copyright Notice

Abstract

This document specifies the Diameter Application for Authentication and Authorization in Web Applications (Diameter WebAuth). This

Diameter application is intended to be used by Diameter clients to perform authentication and authorization operations with a Diameter server in web-based environments. It provides facilities to allow web sites to authenticate their web user clients using a number of (HTTP) authentication schemes. In addition, it supports user authorization using dedicated service identifiers. Diameter WebAuth may also be used by non web-based Diameter clients and servers that require a lightweight authentication and authorization Diameter application.

Table of Contents

1. Introduction

This document describes the Diameter Application for Authentication and Authorization in Web Applications (Diameter WebAuth). The intended area of application for Diameter WebAuth are web applications that want to utilize a Diameter server for authentication and authorization of their users. It enables a Diameter server to supply web sites that implement a Diameter WebAuth client with data to authenticate its user via common HTTP authentication methods. Furthermore, it allows the Diameter client to authorize the access to resources or services provided by the web site.

A relevant usage scenario of Diameter WebAuth is deployment in Identity Management Frameworks where there may be different trust relationships between the user, the web application server and the authentication server. This means

1. No re-usable authentication credentials are shared with the web application server,

2. The authentication server can hold back authentication or authorization information until they are actually needed by the web application server.

Diameter WebAuth specifically addresses the authentication and authorization requirements for the purpose of Identity Management.

Diameter WebAuth does not rely on other Diameter applications and is

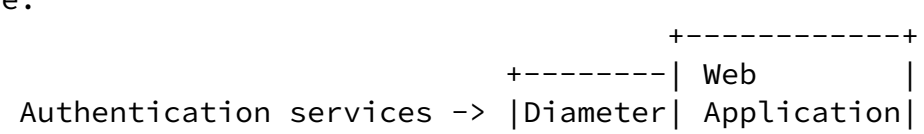
intended to be lightweight and straightforward. This makes it feasible in resource-constrained environments, such as authentication and authorization within embedded systems.

1.1. Motivation and Goals

Several Diameter applications have been defined for various services, like network access ([RFC4005]), Mobile IP ([RFC4004]) or the Session Initiation Protocol ([RFC4740]). The existing applications however are not particularly designed for the use in combination with web applications, many of which require authentication and authorization. Specifically, they do not offer methods suitable for authentication and authorization in a web-based environment, for example the HTTP Digest Authentication. Or they are intended for other applications and require extensive and complex implementation work which, however, is not needed for the intended use in web-based environments. Web applications (or web servers itself for that matter), therefore, implement proprietary authentication back-ends or use services that are not primarily designed for extensive authentication operations.

Such services are, for example, LDAP servers, database servers or IMAP servers. This is often the case, even though there is an AAA service like Diameter available within their administrative domain. The objective of this document is to specify a Diameter application that allows web servers and web applications to utilize a Diameter AAA infrastructure for authentication and authorization purposes.

Diameter WebAuth allows for a Diameter client and a Diameter server to be located either in the same or in different administrative domains. This allows for three party scenarios, for example, an end-user that has signed up with a dedicated identity management provider that operates a Diameter infrastructure providing authentication services for web application providers. As shown in Figure 1 in such three party scenarios, the end-user has a profound trust relationship with the identity management provider but not with the provider of the service he is accessing. Therefore special attention has to be paid to assuring the privacy of the end-user towards the application service provider while enabling the service provider to render its service.



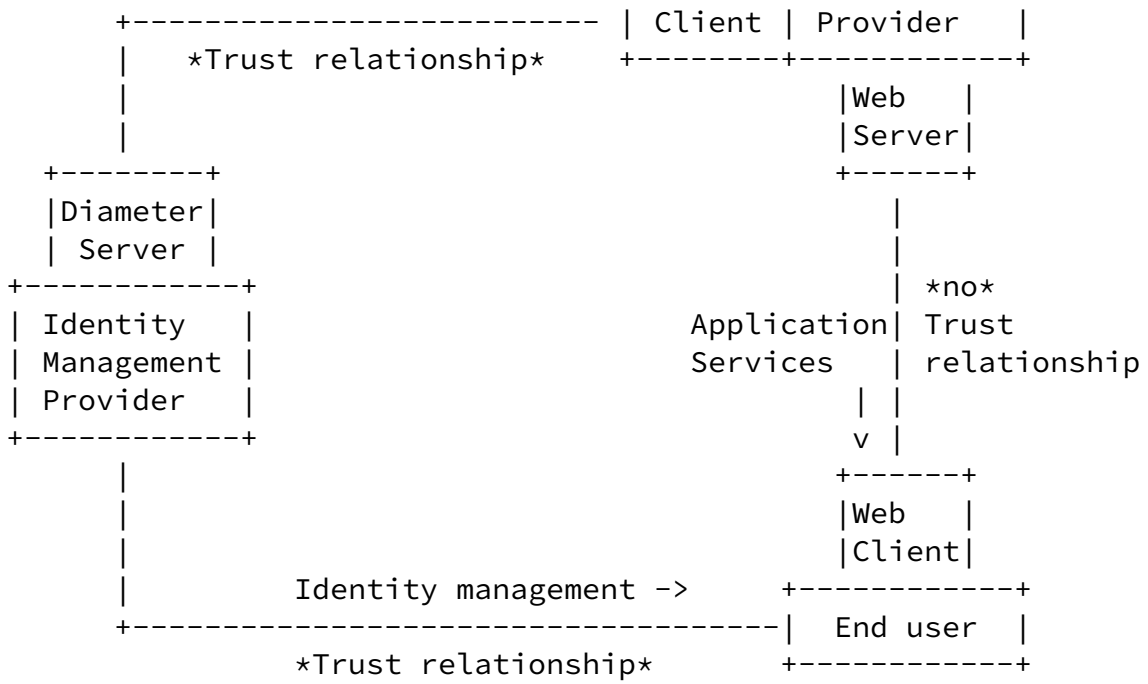


Figure 1: Trust relationships in a three party scenario

Overall, the goals for this Diameter application are as follows:

Lightweight and easy to implement in Diameter clients as well as Diameter servers. Examples for Diameter clients this application is intended for, are web servers and web applications. In general, every entity that provides services using a web-based user interface.

Secure and Privacy protection for scenarios where the Diameter server and the Diameter client are not part of the same administrative Domain. This is, for example, the case when the Diameter server is operated by a third party such as an identity management provider.

1.2. Use cases

This section describes a number of typical use cases that this

document is intended to cover. Those are authentication and authorization of a user by a web server or a web application.

1.2.1. A web site wants to authenticate a user

A user accessing a web site needs to be authenticated in order to link him to an identity. This can be necessary, for example, if a returning visitor ought to be matched to his profile or if access to the site is limited to registered users only. Authentication is also necessary to establish the identity of a user in order to perform service-specific authorization.

1.2.2. A web site wants to authorize a user for a specific service

A resource that is requested by a user requires special access privileges. The web site needs to authorize the user for this resource before allowing him access. It is possible for the web site to maintain different areas with different access requirements so that authorization needs to be repeated for different services and can yield different results.

1.3. Terminology

Basic Authentication: Verifying a users identity based on a plain text password and user name.

User Client: End user client which is used to access the resource on the protected server. Usually a web browser accessing a web server.

Web Application: A computer program that is accessed via a web interface. Usually served by an application server or a web server.

Web Site: A collection of web pages that are intended to be accessed by a web browser. They can be statically served by a web server or dynamically generated by a web application.

1.4. Requirements Language

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119 [RFC2119].

2. Overview

Diameter WebAuth provides a web server with the means to utilize an AAA infrastructure to authenticate and authorize its users for the access to its resources and services. The following sections detail these functions.

2.1. Authentication

Diameter WebAuth extends the facilities provided by the Diameter Base Protocol for a Diameter client to authenticate a user. Two requirements are to be kept regarding the authentication. First, Diameter WebAuth must use standard authentication methods that are supported by the user client. The reason for that is, that Diameter WebAuth only specifies the protocol between the Diameter client and the Diameter server. It cannot alter or adjust the service specific protocol between the user client and the Diameter client, HTTP in this case. The second requirement is that the authentication method needs to provide protection against unauthorized access to secret credentials. In case of username/password authentication this would be the password. Particularly this means that in scenarios where the Diameter client is outside the trust domain of the Diameter server, the secret credentials needs to be protected against the Diameter client itself.

The most common authentication method supported by web browsers is username/password authentication. RFC 2617 [RFC2617] specifies two HTTP authentication methods which are widely supported by web browsers: Basic Authentication and Digest Access Authentication. While the basic authentication exchanges the credentials including the password in cleartext, the digest access authentication uses a one-way hash function to prevent sending the password in cleartext. Although the digest authentication is not intended to be an

absolutely secure authentication scheme, it serves the purpose of

protecting the user password against snooping by any entity between the user client and the authenticator, which in this case would be the Diameter server. Besides HTTP digest access authentication, Diameter WebAuth will, nevertheless, support basic authentication as well. It can be used as a fall back in environments where digest authentication is not available or not necessary and to more generally support different authentication mechanisms, for example, HTML-form-based authentication. The authentication methods supported by this document are detailed in the following sections.

2.1.1. HTTP Basic Authentication

HTTP Basic Authentication as described in [RFC2617] is used when the WebAuth-Authentication-Type AVP is set to HTTP_BASIC in an AA-Request message. The HTTP basic authentication scheme uses a plain username/password combination for authentication. The client, therefore has to include a User-Name AVP and a User-Password AVP in the request. The Diameter server replies with an AA-Answer which MUST include a WebAuth-Authentication-Type AVP also set to HTTP_BASIC and the result of the request. The Diameter server replies with an AA-Answer message that includes a copy of the User-Name AVP and has its Result-Code AVP set to DIAMETER_SUCCESS if the username/password could be verified and DIAMETER_AUTHENTICATION_REJECTED otherwise.

2.1.2. HTTP Digest Authentication

HTTP Basic Authentication as described in [RFC2617] is used when the WebAuth-Authentication-Type AVP is set to HTTP_DIGEST in an AA-Request message. The HTTP digest authentication scheme uses a challenge/response mechanism, therefore, multiple protocol round-trips are needed. An example of an authentication session using the HTTP digest authentication scheme is shown in Figure 2. When a user client sends a request for a protected resource without including any credentials (1.), the Diameter client starts the authentication process. it sends an AA-Request to the Diameter server (2.) which includes a WebAuth-Authentication-Type AVP is set to HTTP_DIGEST. The Diameter server then generates a HTTP-Digest-Challenge AVP and sends it to the client in an AA-Answer with the Result-Code AVP set to DIAMETER_MULTI_ROUND_AUTH (3.). Using the credentials provided by the Diameter server, the Diameter client can construct an HTTP response with the appropriate WWW-Authenticate header and send it to the web user client (4.) to challenge an authentication. Next, the client assembles his authentication credentials and sends another request to the web server (5.) including the user's credentials. The Diameter client assembles an AA-Request to the Diameter server with the corresponding information from the clients request (6.). If the credentials match the records in the Diameter server, it returns an

AA-Answer with the Result-Code AVP set to DIAMETER_SUCCESS (7.). After receiving a positive authentication response, the web server can respond to the user clients request (8.) and grant access to the protected resource.

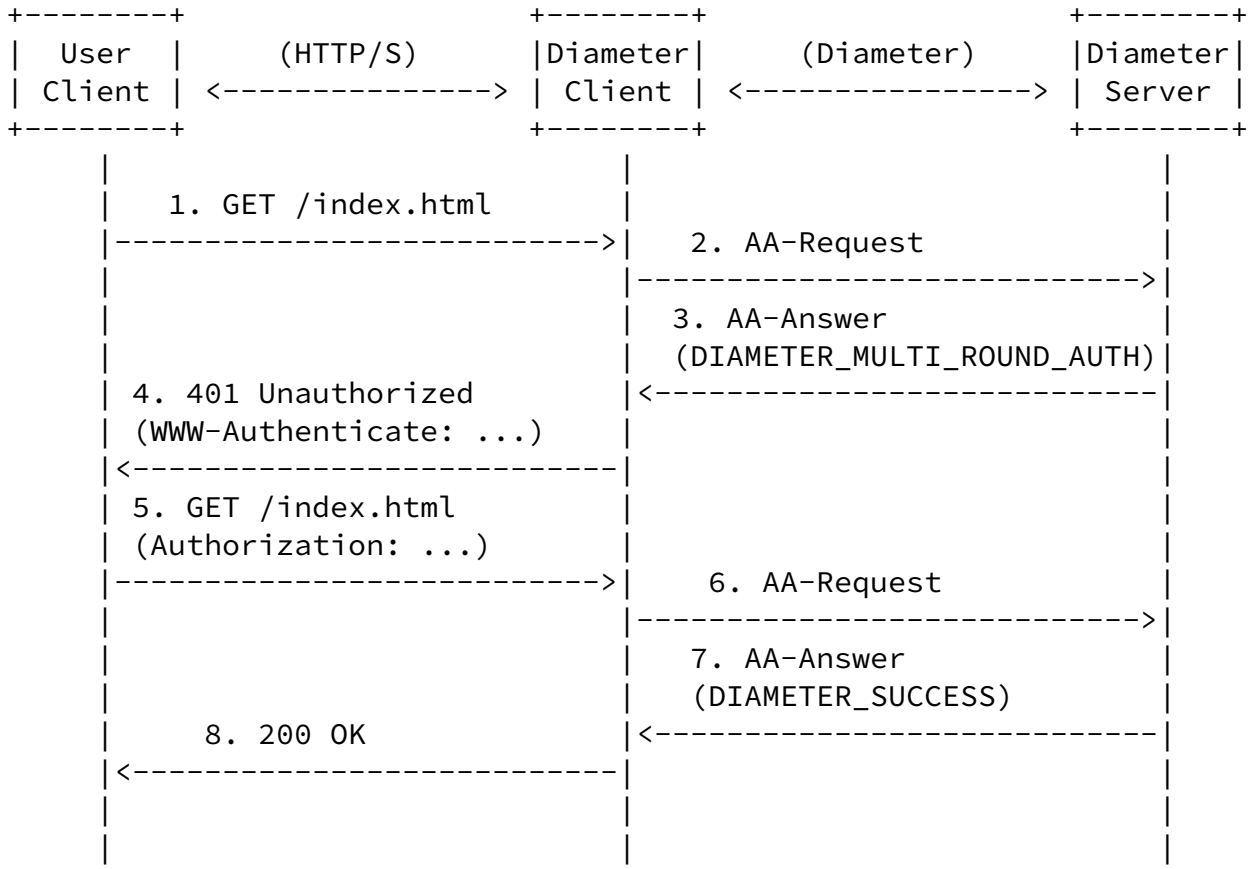


Figure 2: Diameter WebAuth using HTTP digest authentication

2.1.2.1. Quick Mode

Because the HTTP digest authentication scheme uses a challenge/response mechanism, usually at least two protocol round trips are necessary: one to exchange the challenge and one for the response. Besides the obvious additional costs in time, network utilization and processing power this also obligates the Diameter server to maintain a session with an associated state for the authentication procedure. Although each Diameter application implementing this document MUST support the HTTP digest authentication as described above, they CAN employ facilities to speed up the authentication and reduce the necessary protocol round trips to one. If Diameter server and Diameter client both implement and apply these facilities we call this the HTTP digest quick mode.

The HTTP digest quick mode aims at reducing the number of protocol

round trips by prematurely including information that is usually exchanged in a subsequent round trip. If Diameter server and

Diameter client employ these facilities there are a number of security relevant compromises implied that are discussed in Section Section 7.2.1.

In order for a Diameter client to offer a quick mode digest authentication to the Diameter server it will generate the digest nonce itself and do the HTTP authentication with the user client based on this nonce. Therefore it can start the Diameter WebAuth session with an AA-Request that includes a complete HTTP-Digest-Response AVP. The Diameter server can chose to continue the authentication process using this AVP as if the request was following an AA-Answer which included a server-generated HTTP-Digest-Challenge AVP. If the Diameter server does not want to agree on using the client side initiated quick mode, it MUST process the AA-Request as if it is an initial request and ignore the HTTP-Digest-Response AVP. Consequently, a Diameter client starting a digest quick mode authentication MUST anticipate the server not to agree on the quick mode and to reply with an AA-Answer containing a HTTP-Digest-Challenge AVP.

The server side quick mode is employed by a Diameter server by including a Digest-HA1 AVP in the HTTP-Digest-Challenge AVP in its AA-Answer. The Digest-HA1 AVP contains the H(A1) value as defined in [RFC2617] and allows the Diameter client to verify the user client's HTTP authentication response directly and without the need for another Diameter message exchange. The drawback of the server initiated quick mode is that the server will not get a message about the outcome of the authentication process. It therefore CANNOT assume the authentication to be successful. Also, similar to the client initiated quick mode, the Diameter server MUST anticipate the client not to agree on the quick mode and replying to the AA-Answer with another AA-Request that includes a HTTP-Digest-Response AVP with the user client's response to the authentication challenge.

2.2. Authorization

To facilitate different roles and access levels, this document adopts the specification of services made in RFC 4006 [RFC4006], namely the Service-Context-Id and Service-Identifier AVPs. The service

identifiers can be used by web applications to request user differentiated authorization. The Diameter credit-control application introduces service description based on a service context identifier combined with a service identifier. While the service context identifier is used to describe the service specific document that applies to the request, the service identifier designates the exact service within that document.

2.3. Advertising Application Support

Diameter nodes conforming to this document MUST advertise support by including the value of TBD in the Auth-Application-Id of the Capabilities-Exchange-Request and Capabilities-Exchange-Answer command defined in [RFC3588].

3. Command Codes

This section defines the Diameter message Command-Code values that MUST be supported by all Diameter implementations conforming to this document. The Command Codes are as follows:

Command-Name	Abbrev.	Code	Reference
AA-Request	AAR	265	Section 3.1
AA-Answer	AAA	265	Section 3.2

3.1. AA-Request (AAR) Command

The AA-Request (AAR) command is specified in RFC 4005, Section 3.1. [RFC4005] and It is used by the Diameter client to request authentication and/or authorization for its user.

If authentication is requested, depending on the authentication scheme and the sequence of requests different attributes MUST be present: User-Name and User-Password for basic authentication and a HTTP-Digest-Response if it is an AA-Request following an AA-Answer with its Result-Code set to DIAMETER_MULTI_ROUND_AUTH and including a

HTTP-Digest-Challenge.

If authorization is requested, the Service-Context-Id and Service-Identifier attributes are used to identify the service for which authorization is requested. If these attributes are missing in the request and the Auth-Request-Type attribute is set to AUTHORIZE_AUTHENTICATE, the Diameter server SHOULD handle the request as if authorization has not been requested.

The AA-Request command has the following ABNF grammar (AVPs not required by this document are omitted):


```
<AA-Request> ::= < Diameter Header: 265, REQ, PXY >
                 < Session-Id >
                 { Auth-Application-Id }
                 { Origin-Host }
                 { Origin-Realm }
                 { Destination-Realm }
                 { Auth-Request-Type }
                 { WebAuth-Authentication-Type }
                 [ User-Name ]
                 [ User-Password ]
                 [ HTTP-Digest-Response ]
                 [ Destination-Host ]
                 [ Service-Context-Id ]
                 [ Service-Identifier ]
               * [ Proxy-Info ]
               * [ Route-Record ]
               * [ AVP ]
```

3.2. AA-Answer (AAA) Command

The AA-Answer (AAA) command is specified in RFC 4005, Section 3.2. [RFC4005] and is sent by the Diameter server in response to an AA-Request.

If the AA-Answer is a response to a AA-Request initiating a digest

authentication, the Result-Code AVP MUST be set to DIAMETER_MULTI_ROUND_AUTH and a HTTP-Digest-Challenge AVP MUST be present. If the AA-Answer is a response to an authorization request, the Service-Context-Id and Service-Identifier attributes identifying the service for which authorization is granted or denied MUST be present.

The Web-Authentication-Request command has the following ABNF grammar (AVPs not required by this document are omitted):


```
<AA-Answer> ::= < Diameter Header: 265, PXY >
                < Session-Id >
                { Auth-Application-Id }
                { Auth-Request-Type }
                { Result-Code }
                { Origin-Host }
                { Origin-Realm }
                { WebAuth-Authentication-Type }
                [ User-Name ]
                [ HTTP-Digest-Challenge ]
                [ HTTP-Authentication-Info ]
                [ Service-Context-Id ]
                [ Service-Identifier ]
              * [ Proxy-Info ]
              * [ AVP ]
```

4. Diameter WebAuth AVPs

This section provides a listing of the AVPs used in Diameter WebAuth commands and their values.

4.1. Authentication AVPs

Diameter WebAuth defines a new authentication AVP, namely the WebAuth-Authentication-Type AVP, which is described below.

4.1.1. WebAuth-Authentication-Type AVP

The WebAuth-Authentication-Type AVP (AVP Code TBD) is of type Enumerated. In an AA-Request it indicates the type of authentication mechanism that is requested by the client while in an AA-Answer it indicates the authentication mechanism the Diameter server used to answer the initial request. The Diameter server MUST always use the authentication type requested by the client in the request. The AVP is mirrored in the answer to allow the client to be stateless regarding the authentication type.

A Diameter server is not required to support all of the authentication types. An unsupported authentication type can for example not be implemented in the server or be disabled by a configuration option due to security or policy constraints. If an unknown or unsupported WebAuth-Authentication-Type is received in an AA-Request, the Diameter server MUST reply with an AA-Answer with its Result-Code AVP set to DIAMETER_INVALID_AVP_VALUE including a copy of the WebAuth-Authentication-Type AVP.

The following values are defined for the WebAuth-Authentication-Type

AVP:

HTTP_BASIC (0)
HTTP basic authentication as described in [RFC2617].

HTTP_DIGEST (1)
HTTP digest authentication as described in [RFC2617].

4.2. Authorization AVPs

Diameter WebAuth defines two new AVP used for authorization purposes,

namely the WebAuth-Authorization-Request and WebAuth-Authorization-Response AVPs. The AVPs are described below.

4.2.1. WebAuth-Authorization-Request AVP

The WebAuth-Authorization-Request AVP is send by a client inside an AA-Request which has its Auth-Request-Type AVP set to either AUTHORIZE_ONLY or AUTHORIZE_AUTHENTICATE. If a WebAuth-Authorization-Request AVP is present in such a request, it indicates to the Diameter server, that the client wants to authorize its user based on the values included in the AVP. The Diameter server processes the request according to its configuration and includes an appropriate Result-Code AVP in a subsequent AA-Response.

The exact manner in which the Diameter server processes the authorization request is implementation and configuration dependent. For example, the Diameter server can take every data that is provided within the WebAuth-Authorization-Request AVP into account and only grant authorization when the user qualifies for each and every one. Opposed to that, the Diameter server can also authorize the user if only one of the conditions is met. The definite authorization procedure is expected to be arranged between the Diameter client provider and the Diameter server provider.

The WebAuth-Authorization-Request AVP is of type Grouped and has the following ABNF grammar:

```
WebAuth-Authorization-Request ::= < AVP Header: TBD >
                                  [ WebAuth-URI ]
                                  [ WebAuth-Service ]
                                  [ Remote-User ]
                                  [ User-Name ]
                                  [ Remote-Address ]
                                * [ AVP ]
```


4.2.2. WebAuth-URI AVP

The WebAuth-URI AVP (AVP Code TBD) is of type UTF8String and contains the URI the user tried to access when the authorization was triggered

by the Diameter client as described in RFC 1738.

4.2.3. WebAuth-Service AVP

The WebAuth-Service AVP (AVP Code TBD is of type UTF8String and contains a name or identifier for the service the Diameter client wants to authorize the user for. The valid service names or identifiers are prearranged between the Web application provider and the Diameter server provider.

4.2.4. Remote-User AVP

The Remote-User AVP (AVP Code TBD) is of type UTF8String and contains the identification of the remote user that is trying to access the service for which the Diameter client is requesting the authorization. Contrary to the User-Name AVP the value in this AVP can have been obtained by the Diameter client by Diameter-external means.

4.2.5. Remote-Address AVP

The Remote Address AVP (AVP code TBD) is of type Address and contains the network address (e.g. IPv4 or IPv6 address) of the remote user that is trying to access the service for which the Diameter client is requesting authorization.

4.3. Diameter Base Protocol AVPs

This Diameter application uses the following AVPs specified in RFC 3588 [RFC3588]:

Attribute Name	AVP Code	Value Type	Reference
Origin-Host	264	DiameterIdentity	RFC 3588, Section 6.3. [RFC3588]
Origin-Realm	296	DiameterIdentity	RFC 3588, Section 6.4. [RFC3588]
Destination-Host (??)	293	DiameterIdentity	RFC 3588, Section 6.5. [RFC3588]
Destination-Realm	283	DiameterIdentity	RFC 3588, Section 6.6. [RFC3588]
Auth-Application-Id	258	Unsigned32	RFC 3588, Section 6.8. [RFC3588]
Acct-Application-Id	259	Unsigned32	RFC 3588, Section 6.9. [RFC3588]
Result-Code	268	Unsigned32	RFC 3588, Section 7.1. [RFC3588]
Auth-Request-Type	274	Enumerated	RFC 3588, Section 8.7. [RFC3588]
Session-Id	263	UTF8String	RFC 3588, Section 8.8. [RFC3588]
Authorization-Lifetime	291	Unsigned32	RFC 3588, Section 8.9. [RFC3588]
Auth-Grace-Period	276	Unsigned32	RFC 3588, Section 8.10. [RFC3588]
Auth-Session-State	277	Enumerated	RFC 3588, Section 8.11. [RFC3588]
User-Name	1	UTF8String	RFC 3588, Section 8.14. [RFC3588]
Event-Timestamp	55	Time	RFC 3588, Section 8.21. [RFC3588]

4.4. Diameter Network Access Server Application AVPs

This Diameter application uses the following AVPs specified in RFC 4005 [RFC4005]:

Attribute Name	AVP Code	Value Type	Reference
User-Password	2	OctetString	RFC 4005, Section 5.1. [RFC4005]

4.5. HTTP-Digest Authentication AVPs

The following section describes the AVPs used for the HTTP-Digest Authentication in Web-Auth-Request and Web-Auth-Response commands.

4.5.1. HTTP-Digest-Challenge AVP

The HTTP-Digest-Challenge AVP is identical to the SIP-Authenticate AVP specified in RFC 4740, Section 9.5.3. [RFC4740] and is renamed here for descriptive reasons.

The HTTP-Digest-Challenge AVP has the following ABNF grammar:

```
HTTP-Digest-Challenge ::= < AVP Header: 379 >
                          { Digest-Realm }
                          { Digest-Nonce }
                          [ Digest-Domain ]
                          [ Digest-Opaque ]
                          [ Digest-Stale ]
                          [ Digest-Algorithm ]
                          [ Digest-QoP ]
                          [ Digest-HA1]
                        * [ Digest-Auth-Param ]
                        * [ AVP ]
```

4.5.2. HTTP-Digest-Response AVP

The HTTP-Digest-Response AVP is identical to the SIP-Authorization

AVP specified in RFC 4740, Section 9.5.4. [RFC4740] and is renamed here for descriptive reasons.

The HTTP-Digest-Response AVP has the following ABNF grammar:


```
HTTP-Digest-Response ::= < AVP Header: 380 >
                         { Digest-Username }
                         { Digest-Realm }
                         { Digest-Nonce }
                         { Digest-URI }
                         { Digest-Response }
                         [ Digest-Algorithm ]
                         [ Digest-CNonce ]
                         [ Digest-Opaque ]
                         [ Digest-QoP ]
                         [ Digest-Nonce-Count ]
                         [ Digest-Method]
                         [ Digest-Entity-Body-Hash ]
                       * [ Digest-Auth-Param ]
                       * [ AVP ]
```

4.5.3. HTTP-Authentication-Info AVP

The HTTP-Digest-Info AVP is identical to the SIP-Authentication-Info AVP specified in RFC 4740, Section 9.5.5. [RFC4740] and is renamed here for descriptive reasons.

The HTTP-Digest-Info AVP has the following ABNF grammar:

```
HTTP-Digest-Info ::= < AVP Header: 381 >
                     [ Digest-Nextnonce ]
                     [ Digest-QoP ]
                     [ Digest-Response-Auth ]
                     [ Digest-CNonce ]
                     [ Digest-Nonce-Count ]
                   * [ AVP ]
```

4.5.4. HTTP Digest AVPs

The following table lists all AVPS that are RADIUS attributes defined in RFC 5090 [RFC5090] and that are imported by RFC 4740 (see Section 9.5.6.) [RFC4740] to be used for the HTTP-Digest authentication.

Attribute Name	RADIUS Type
Digest-Response	103
Digest-Realm	104
Digest-Nonce	105
Digest-Response-Auth	106
Digest-Nextnonce	107
Digest-Method	108
Digest-URI	109
Digest-QoP	110
Digest-Algorithm	111
Digest-Entity-Body-Hash	112
Digest-CNonce	113
Digest-Nonce-Count	114
Digest-Username	115
Digest-Opaque	116
Digest-Auth-Param	117
Digest-AKA-Auts	118
Digest-Domain	119
Digest-Stale	120
Digest-HA1	121

4.6. Diameter Credit-Control Application AVPs

The following section describes the AVPs specified in RFC 4006 [RFC4006] and used by this application.

4.6.1. Other Credit-Control Application AVPs

The following AVPs are also used by this application:

Attribute Name	AVP Code	Value Type	Reference
Service-Identifier	439	Unsigned32	RFC 4006, Section 8.28. [RFC4006]
Service-Context-Id	461	UTF8String	RFC 4006, Section 8.42. [RFC4006]

5. IANA Considerations

This document serves as IANA registration request for a number of items that should be registered in the AAA parameters registry.

5.1. Application Identifier

This document assigns the value TBD, "Diameter Application for Authentication and Authorization in Web Applications", to the Application Identifier namespace defined in (RFC 3588 [RFC3588]. See Section Section 2.3 for more information.

5.2. AVP Codes

This document defines new standard AVPs, whose AVP Codes are to be allocated within the AVP Codes address space defined in (RFC 3588, Section 11.4. [RFC3588]. These AVP codes have been registered in the AVP Codes sub-registry of the AAA parameters registry. The sole new standard AVP that is specified for this Diameter application is the WebAuth-Authentication-Type AVP. See Section Section 4.1.1 for more information.

6. Privacy Considerations

The Diameter application aims at covering setups where Diameter clients and Diameter servers belong to more than one administrative domain. In those setups the end user often has a trust relationship with the provider of the Diameter server but not with the provider of the web applications that are the Diameter clients. In order to allow a smooth operation of the services the user requested, the Diameter server has to make certain personal information about the user available to the application provider. And although the user should be aware of that, it can be generally expected that access to such personal information is kept on a minimum need-to-know basis across different administrative domains. For example the application provider may need to know if the user has a certain membership which allows him to access the service he requested. The number and details about further memberships the user may or may not have however, is not relevant for the application provider at that moment. This section therefore addresses a number of privacy consideration that may arise in general or when dealing with a setup over multiple administrative domains. Since usually there are no private information that the client has but not the server, the privacy considerations will focus on the issue to protect information that are available in the Diameter server from access by the Diameter client.

Generally, in setups where user privacy is an aspect, Diameter servers SHOULD always require a user authentication before any kind of personal information is made accessible to the Diameter client. By requiring an authentication, user data probing by a rogue or compromised Diameter client is made more difficult since only data

from users that are currently logged onto the client or whose login credentials are known can be pried. If the authentication status for a session is not maintained on the server, every action specified in this chapter can be queried using an AA-Request command which then MUST also include proper authentication credentials. However since an authentication procedure possibly triggers some kind of user interaction in the web client, it is RECOMMENDED to keep such AA-Requests to a minimum. This can be achieved for example by querying the Diameter server for all the data that is likely to be needed for a session inside the first request. Although this may sound counterintuitive to the objective of keeping private information exposure on a minimum need-to-know basis, it doesn't make a

difference if data which a client is entitled to is transferred all at once in the beginning of a session or gradually throughout the session. Implementations of this document which want to allow privacy protection SHOULD offer a configuration option to enforce user authentication before any other operation is allowed.

A Diameter WebAuth implementation SHOULD protect personal data by keeping authorization data service specific and by limiting available authentication schemes to the ones which do not expose sensitive data. Keeping authorization data service specific means that the Diameter server SHOULD NOT authorize the user for services that the Diameter client doesn't actually offer. This means that an AA-Answer to an authorization request SHOULD NOT include Service-Identifiers for services that are unavailable at the client the request came from. Unfortunately, the Diameter server cannot directly influence the authentication scheme that the Diameter client uses with its web client (see also Section Section 7.3). However, limiting the available authentication schemes to more secure ones will hopefully encourage Diameter clients to be deployed using only the available authentication schemes to begin with. This should make eavesdropping on the Diameter client, web client connection more difficult and also will require more changes to a compromised Diameter client in order to gain access to plain text authentication credentials. The only authentication scheme which can be considered reasonable secure and is currently supported by this document is HTTP-Digest authentication.

7. Security Considerations

This document describes a Diameter application which enables web applications to access AAA services of a Diameter server. The Diameter Base Protocol (RFC 3588) is used for the communication between the Diameter client and the Diameter server. The security considerations for the Base Protocol, therefore, apply to this document as well (RFC 3588, Section 13) [RFC3588]. This document

assumes, that the message exchange between the Diameter client and the Diameter server can be reasonably secured by respecting the security considerations in RFC 3588.

For the communication between the end user and the Diameter client a

service specific protocol is used. When the Diameter client is employed in a web application, usually this will be the Hypertext Transfer Protocol (HTTP, RFC 2616). The security considerations for the service specific protocol SHOULD be considered when a Diameter WebAuth client is implemented. In case of a web application employing HTTP, the correspondent security considerations are made in RFC 2616, Section 15 [RFC2616]. In either case, since the service protocol is used to exchange authentication information with the end user, measures SHOULD be taken to secure the communication between the Diameter client and the end user client. To secure HTTP message exchanges, for example, HTTPS (HTTP over TLS, RFC 2818 [RFC2818]) SHOULD be used.

7.1. HTTP Basic Authentication

The basic authentication scheme uses a cleartext user name/password combination to authenticate a user. This makes the basic authentication absolutely insecure. First of all, the password is exposed to any third party which might be able to listen to the message exchange between the user client and the Diameter client. For example because the message exchange is not encrypted, the encryption was broken of for other reasons. And second of all, the password, inevitably, is exposed to the Diameter client. Especially in setups where the Diameter client and the Diameter server are not part of the same administrative domain this severely compromises the end user's identity. Even in setups where Diameter client and server are within the same administrative domain, user passwords should never be accessible in cleartext. Otherwise in case of a compromised Diameter client, all the user accounts are compromised too. Because basic authentication is that insecure, it SHOULD NOT be employed in a productive Diameter setup, unless absolutely no other option is viable. Furthermore basic authentication SHOULD only be used over encrypted and secure transport channels with some sort of server authentication before the credentials are sent. Besides these Diameter WebAuth oriented security considerations, those of the HTTP Authentication specification (RFC 2617, Section 4 [RFC2617]) also need to be considered. For example, they state explicitly that "the Basic authentication scheme is not a secure method of user authentication, nor does it in any way protect the entity, which is transmitted in cleartext across the physical network used as the carrier."

7.2. HTTP Digest Authentication

Like the basic authentication, the digest authentication uses a user name in combination with a password to authenticate the user. In contrast to the basic authentication, however, the digest authentication does not exchange the password in cleartext. It uses a one-way hash function to calculate a check value from the combination of the user password and a nonce that was exchanged with the authentication partner. Both authentication partners calculate this value on their own, and the client which is to be authenticated sends its value to the authenticator. If the values match, both used the same password and, therefore, the authentication is successful.

The digest authentication, if implemented and executed correctly, does provide a better authentication mechanism than basic authentication. Especially an eavesdropping third party cannot recover the cleartext password from an intercepted message exchange. Nor can he use it for replay attacks when the server does not reuse its nonce values. Nevertheless, the HTTP Authentication specification (RFC 2617, RFC 2617, Section 4 [RFC2617]) has a number of security considerations that must be considered. Especially is the digest authentication scheme susceptible to man in the middle attacks. It does provide some resilience against the attacker recovering the cleartext password in those cases though. Also the security considerations of the RADIUS Extension for Digest Authentication specifications (RFC 5090) which the Diameter digest authentication is derived from need to be considered RFC 5090, Section 8 [RFC5090]. As a result, the digest authentication scheme also SHOULD only be used over encrypted and secure communication channels. This includes the authentication of the Diameter client to the user client, for example HTTPS with public key certificates.

7.2.1. Digest Quick Mode

The HTTP digest quick mode (see Section Section 2.1.2.1) reduces the number of protocol round trips by prematurely including information that is otherwise exchanged in a subsequent round trip. The reduction in protocol round trips, however, needs to be balanced against the security compromises that come with it. The digest quick mode induces the release of control over the authentication process from the Diameter server to the Diameter client. Whether this is acceptable or not has to be carefully considered depending on the respective setup.

A client initiated quick mode means that the digest nonce which is used during the HTTP authentication with the user client is generated by the Diameter client instead of the Diameter server. This allows the Diameter client to carry out a number of attacks against the user

client and induces potential security risks for the secrecy of the user's password. A good nonce value is critical to the integrity of the HTTP digest authentication scheme. It is, therefore, imperative that the nonce generation on the Diameter client is as secure and reliable as the implementation on the Diameter server if no additional security risks are to be introduces. If a Diameter client is not implementing a secure and reliable nonce generation routine, from the security point of view it is better to use a server generated nonce and accept the additional protocol round trip.

Permitting client initiated quick mode also might allow an attacker to use a compromised Diameter client to carry out chosen plaintext attacks, precomputed dictionary attacks, online dictionary attacks or other form of attacks using cryptanalysis. A countermeasure against those form of attack is for user clients to use a client-specific nonce (cnonce) during the HTTP authentication. The behavior of the user clients, however, is out of control of the Diameter server. Moreover, in case the Diameter client is compromised by an attacker it is reasonably to assume that the attacker can carry out those forms of attacks regardless of the security parameters of the Diameter server.

The server initiated quick mode exposes the H(A1) value to the Diameter client. This allows for an attacker to carry out cryptanalysis attacks against this value instead of only the hash which is based on a one-time nonce. More importantly, the H(A1) value is the basis for the digest computation for a certain realm. This means that if an attacker gains access to this value the user password must be regarded as compromised for this realm. As a countermeasure, the names for realms that are secured by separate Diamter clients SHOULD be different. In general, realm names SHOULD always be unique within the domain of a Diameter server. In case a Diameter client was compromised, the realm name for the respective administrative domain needs to be changed, which in turn invalidates the compromised H(A1) value. The possibility to discover the original password from the H(A1) value, for example, by the means of a successful cryptanalysis attack, however, are not mitigated by this precaution.

7.3. Renegade or Compromised WebAuth Clients

Special considerations need to be made for the situation where a

Diameter WebAuth client is compromised or renegade. In both cases the WebAuth client will try to exploit its natural position as man in the middle between the user client and the Diameter server to compromise user accounts. A natural goal of an attacker in this position is to gain access to cleartext user credentials. Since the Diameter WebAuth server does not allow direct querying of user names

or passwords, the WebAuth client has two possibilities. It can probe for valid user name/password combination if the server accepts basic authentication AA-Requests or it can wait for user to authenticate themselves. Probing for valid combinations is not very promising and will not considered any further here. Having users to try and authenticate themselves to a WebAuth client that is trying to compromise their accounts, on the other hand, is a severe problem.

As discussed above, when using digest authentication even a man in the middle attack has only limited chances of recovering the cleartext password. A man in the middle attacker, however, can simply switch the authentication scheme used towards the user client to basic authentication. This would give him unrestricted access to the cleartext user name and password for every user that logs in through the Diameter client. This kind of attack is described in RFC 2617, Section 4 [RFC2617] as well. Coinciding with the RFC, the only viable options to counteract such attacks lie within the user agent. For example, only if the user agent warns the user when basic authentication is requested, or in general indicates to the user what kind of authentication is about to be used, this kind of attack can be prevented by the user. Another possibility is for the client to offer a configuration option which either disables basic authentication completely or just for different web sites. For the future of this document it also SHOULD be considered to implement other authentication methods. This will not prevent renegade or compromised WebAuth clients from being able to switch authentication schemes, but from a user's perspective it is much more obvious when, for example, instead of the usual certificate based authentication a web server suddenly ask for a password

8. References

8.1. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

[RFC2617] Franks, J., Hallam-Baker, P., Hostetler, J., Lawrence, S., Leach, P., Luotonen, A., and L. Stewart, "HTTP Authentication: Basic and Digest Access Authentication", RFC 2617, June 1999.

[RFC3588] Calhoun, P., Loughney, J., Guttman, E., Zorn, G., and J. Arkko, "Diameter Base Protocol", RFC 3588, September 2003.

[RFC4005] Calhoun, P., Zorn, G., Spence, D., and D. Mitton, "Diameter Network Access Server Application", RFC 4005,

August 2005.

[RFC4006] Hakala, H., Mattila, L., Koskinen, J-P., Stura, M., and J. Loughney, "Diameter Credit-Control Application", RFC 4006, August 2005.

[RFC4740] Garcia-Martin, M., Belinchon, M., Pallares-Lopez, M., Canales-Valenzuela, C., and K. Tammi, "Diameter Session Initiation Protocol (SIP) Application", RFC 4740, November 2006.

[RFC5090] Sterman, B., Sadolevsky, D., Schwartz, D., Williams, D., and W. Beck, "RADIUS Extension for Digest Authentication", RFC 5090, February 2008.

8.2. Informative References

[RFC2616] Fielding, R., Gettys, J., Mogul, J., Frystyk, H., Masinter, L., Leach, P., and T. Berners-Lee, "Hypertext Transfer Protocol -- HTTP/1.1", RFC 2616, June 1999.

[RFC2818] Rescorla, E., "HTTP Over TLS", RFC 2818, May 2000.

[RFC4004] Calhoun, P., Johansson, T., Perkins, C., Hiller, T., and P. McCann, "Diameter Mobile IPv4 Application", RFC 4004, August 2005.

Appendix A. Open Issues

- Add some attributes for WebAuth to better support web applications? Possible examples: Verification-Code (to use image verification) and attributes to convey pre- and/or post-authentication messages.

- Service identification by the combination of the Service-Context-Id and Service-Identifier (which is a numerical value) attributes seems very cumbersome for the web application environment. Maybe we should add a new AVP which identifies services by its name.

- Do we need an Interoperability section detailing the interoperability of WebAuth with other Diameter applications like Diameter EAP or Diameter CC?

- Support for further authentication schemes like client certificates (SSL)

Authors' Addresses

Niklas Neumann
University of Goettingen
Computer Networks Group
Goldschmidtstr. 7
Goettingen 37077
Germany

Email: niklas.neumann@cs.uni-goettingen.de

Xiaoming Fu
University of Goettingen
Computer Networks Group
Goldschmidtstr. 7
Goettingen 37077
Germany

Email: fu@cs.uni-goettingen.de