1

Introduction

1.1 What is AAA?

AAA stands for *Authentication*, *Authorization*, and *Accounting*.

Authentication is the verification that a user who is requesting services is a valid user of the network services requested. The user must present an identity, like a user name or phone number, and credentials, like a password, a digital certificate, or one-time passphrase, to the verifier in order to be authenticated.

Authorization is the determination of whether requested services can be granted to a user who has presented an identity and credentials based on their authentication, service request, and system state. Authorization state may change over the course of a user's session due to consumption limits or time of day.

Accounting is the tracking of the user's consumption of resources for billing, auditing, and/or system planning. Typical accounting data collected includes the identity of the user, the service delivered, and when the service started and stopped.

Consider a voice-over IP (VoIP) service provider that offers telephony services to a large number of end users. End users can connect to the service with software for VoIP clients that runs on a smart phone, tablet or desktop PC, or they may use a purpose-built hardware phone.

When the user's device contacts the VoIP network, the VoIP service provider will *authenticate* the user accessing their network. That is, the provider wants to determine that the user, or her device, is who they say they are. The authentication mechanisms and credentials vary by deployment. For example, some deployments may use human-memorizable username and password combinations, while others may use a public key infrastructure with certificates stored on smart cards.

Once the VoIP service provider has successfully authenticated the user, the provider will then *authorize* them to use the services by verifying the conditions and privileges of the user's account and the status of the user's credits for the requested action, such as making a phone call.

If the user successfully passes the authorization procedure, the user's resource consumption will be *accounted*. Accounting resource consumption is useful for a number of reasons, including capacity planning, understanding user behavior to improve service experience, charging for service use, and measuring policy compliance. The kinds of data collected as part of the accounting process depend on the application

Diameter: New Generation AAA Protocol – Design, Practice, and Applications, First Edition.
Hannes Tschofenig, Sébastien Decugis, Jean Mahoney and Jouni Korhonen.

context and the needs of the service provider, and the data may need to be collected from various places in the network. For example, one VoIP service provider may collect data about transmitted voice packets. Another provider may be satisfied with collecting data about the call setup procedures only.

Typically VoIP deployments use Session Initiation Protocol (SIP) for call setup. In small VoIP deployments that use SIP, the AAA operations happen within the SIP proxy, which is a network element that helps to route SIP requests to their final destinations. As a SIP network grows larger, the VoIP service provider may deploy a dedicated and centralized AAA server to manage subscribers' information and their authorization properties on behalf of multiple proxies. When a service request arrives at a SIP proxy, the proxy will send AAA-related requests to the AAA server.

The SIP proxy in this distributed network is a kind of network access server. Network access server (NAS) is a generic term for the end user's entry point to a network. A NAS provides services on a per-user basis, based on authentication, and ensures the service provided is accounted for. A NAS contacts a separate AAA server to verify the user's credentials and then sends accounting data to the AAA server. A NAS, then, is an AAA client.

When the AAA functionality is outsourced from a NAS to the AAA server, there needs to be a protocol defined between the AAA client within the NAS and AAA server. Since the developers who created the NAS are likely different than the developers who created the AAA server, it is helpful to not only define a communication protocol, but also to agree on an open standard rather than to use a proprietary interface. In fact, various AAA protocol standards have been defined, with standards work starting with the early Internet dial-up services and progressing to cover connections to today's modern wireless networks.

1.2 Open Standards and the IETF

The standards organization that works to improve the interoperability of the Internet is the Internet Engineering Task Force (IETF), an international community of network designers, operators, vendors, and researchers that develop open, voluntary Internet standards. Examples of such standards include Internet transport (TCP/IP, UDP), email (SMTP), network management (SNMP), web (HTTP), voice over IP (SIP), and also AAA (RADIUS, Diameter). The IETF does not have formal membership requirements and is open to anyone interested in improving the Internet. The newcomer's guide to the IETF is known as *The Tao of the IETF* [1] and can be found online.

Standards work in the IETF is done in working groups, which discuss protocol solutions on mailing lists and in person at IETF meetings, and capture these solutions in documents known as Internet drafts. Working groups are self-organized by topic and are grouped into broad focus areas. Work on AAA protocols has taken place in multiple working groups.

The gauge of a protocol in the IETF is "rough consensus and running code". When the working group has arrived at rough consensus, the Internet draft enters a review period known as a Last Call, in which the larger IETF community can provide input. Internet drafts are then reviewed by the Internet Engineering Steering Group (IESG). When the IESG approves an Internet draft, the draft moves on to become a Request for

Comments (RFC), which, despite its categorization, is now at a level of stability that it can be implemented with confidence.

The details of IETF Internet protocols, such as port numbers, application identifiers, and header field names, are stored with the Internet Assigned Numbers Authority (IANA), which is responsible for the global coordination of Internet protocol resources.

1.3 What is Diameter?

Diameter is an open standard AAA protocol defined by the IETF. Diameter's features fulfill multiple requirements of network operators. The definition of the Diameter protocol is given in the Diameter base specification, RFC 6733 [2].

Various AAA protocols, such as the Common Open Policy Service Protocol (COPS) [3] and Remote Authentication Dial In User Service (RADIUS) [4], had been developed before work on the Diameter protocol started. Experience with these protocols provided the IETF community with requirements for a next-generation AAA protocol. These requirements are documented in RFC 2989, *Criteria for Evaluating AAA Protocols for Network Access* [5]. The design of Diameter incorporated the lessons learned from these various AAA protocols.[1]

As work continued on Diameter, the AAA working group of the IETF [6] evaluated the available AAA protocols against the requirements given in RFC 2989. Those requirements are:

Scalability: The AAA protocol has to be able to support millions of end users and tens of thousands of devices, AAA servers, Network Access Servers, and brokers.

Failover: Failover support aims to provide uninterrupted AAA service in the case of a failure. Failover requires the detection of a failed node and the re-routing of outstanding messages to an alternative node. Failover support may lead to the retransmission of messages, and those duplicate messages should be handled appropriately by the protocol.

Security: AAA protocols carry sensitive data, including long-term authentication credentials (such as passwords), session keys, service usage information as part of accounting records, and possibly the end user's location. From a data protection point of view, this personal data requires special care. Network operators also want to avoid malicious parties injecting false information into the system. For this purpose, providing a common, widely used security mechanism is desirable.

Reliable transport: When accounting records were transmitted over an unreliable transport protocol, as done in earlier protocols (such as RADIUS), packet loss translated to loss of money. Consequently, implementations added their own reliability mechanisms, leading to differences among vendors. Diameter incorporates the reliable transports the Transmission Control Protocol (TCP) and the Stream Control Transmission Protocol (SCTP) to ensure uniform behavior among implementations by different vendors.

Agent support: Earlier AAA protocols did not offer nodes that could route and redirect AAA messages, which can future-proof a AAA network deployment.

---

1 A joke among IETF participants is to point out that diameter is twice the radius. Hence, Diameter is meant to be the next generation AAA protocol after RADIUS.

Server-initiated messaging: In earlier AAA protocols, only the AAA client could initiate the message exchange. The AAA server's ability to initiate messages was added later as an optional feature, and therefore support for it could not be assumed. Server-initiated messaging is used, for example, when authorization characteristics change and re-authorization by the user or the end device is required. With long-running sessions, this initiation has to be triggered by the AAA server towards the AAA client.

Transition support: Since Diameter would be introduced into existing AAA deployments, it needed to provide a transition story to lower the deployment effort for network operators.

Ability to carry service-specific attributes: The AAA protocol needs to be extensible and provide the ability to define new attributes required by new services.

The AAA working group published their results in RFC 3127 [7], *Authentication, Authorization, and Accounting: Protocol Evaluation*, expressing a preference for Diameter since it met most of the requirements specified in RFC 2989 and needed only minor engineering to bring it into complete compliance. Since the Diameter specification was still under development, the working group could address the requirement gaps.

1.3.1 Diameter versus RADIUS

A book about Diameter cannot be silent about its predecessor, RADIUS. RADIUS was originally standardized in January 1997 by the IETF with RFC 2058 [8], which was replaced by RFC 2138 [9] a few months later, and was made obsolete in June 2000 by RFC 2865 [4].

Diameter was able to address deficiencies found in the RADIUS protocol, namely:

- No reliable message delivery. RADIUS used the User Datagram Protocol (UDP), an unreliable transport protocol, to communicate messages from a RADIUS client to a RADIUS server.
- RADIUS had a monolithic design whereby RADIUS attributes used by different applications were put into one bucket for transport with RADIUS. Unlike Diameter, RADIUS did not separate the message delivery from the application's semantics. Interoperability issues and lack of extensibility were common problems in deployments.
- The Datagram Transport Layer Security (DTLS) protocol did not exist at that time, therefore RADIUS had to rely on either IPsec [10] or no security protection at all.
- Only client-initiated messaging. RADIUS initially did not provide a mechanism for letting the server initiate messages.
- RADIUS only supported a basic set of data types, which made it difficult for application designers to define their own RADIUS attributes.

This was, however, not the end of the story since, paralleling the Diameter work within the IETF AAA working group and later continued in the RADIUS [11] and RADEXT [12] working groups, the RADIUS protocol experienced a number of improvements, many of which were inspired by work on the Diameter protocol:

- Reliable message delivery. RFC 6613 [13] added support for the TCP to RADIUS.

- Improved security. RFC 7360 [14] added the ability to use DTLS with UDP-based RADIUS messages. RFC 6614 [15] added support for TLS for TCP-based RADIUS messaging.
- Server-initiated messages. RFC 3576 [16] added support for server-initiated messages as part of the dynamic authorization extensions.
- Extended attribute type space. The demand for attributes had been close to exhausting RADIUS's 8-bit attribute type space. RFC 6929 [17] extended the type space and added a mechanism for complex attributes.
- Design guide. To combat interoperability problems caused by protocol design activities in various organizations, RADIUS design guidelines were published with RFC 6158 [18].

At the time of this writing, development of the RADIUS protocol is still ongoing in the IETF radext working group. However, not only does the IETF develop extensions for RADIUS, but other organizations do also. Hence, the best way to gain an overview of the available extensions is to look at the IANA registry for RADIUS [19].

Today, many of the features of Diameter are also available within RADIUS. It is therefore fair to ask which communities are driving the development of each protocol. It turns out that many small- and medium-size enterprises use RADIUS, including many WLAN hotspot deployments, universities, and digital subscriber line (DSL) and cable operators. On the other hand, large Internet service providers, and particularly mobile operators, use Diameter in their network architectures. The market is therefore is nicely divided, and does not lead to rivalry in the standardization environment.

1.3.2 Diameter Improvements

It is important to note that the Diameter base specification (RFC 6733 [2]) is a revision of the original Diameter protocol, specified in RFC 3588 [20], and is the output of the IETF DIME working group [21], which incorporated feedback of protocol implementers from interoperability testing events and discussions on working group mailing lists. RFC 6733 obsoletes RFC 3588.

The main differences between RFC 3588 and RFC 6733 are the following:

Security: RFC 6733 specifies Transport Layer Security (TLS) (when used with TCP) and DTLS (when used with SCTP) as the primary ways to secure Diameter messages. It also specifies the use of a well-known port, which is similar to how TLS is used with other application-layer protocols such as HTTPS. The end-to-end security framework described in RFC 3588 is deprecated since the actual technical solution has not yet been standardized. More discussion about Diameter security can be found in Chapter 5.

Diameter Node Discovery: A Diameter node discovers to which node it needs to talk via either manual configuration or a dynamic discovery procedure. RFC 6733 simplifies the dynamic discovery procedure since it was observed that many vendors had implemented only the DNS-based mechanism.

Extensibility: The story for extending Diameter presented in RFC 3588 was unclear and led to incompatible extensions. RFC 6733 clarified Diameter extensibility, and Chapter 7 is dedicated to the topic of Diameter extensibility to provide help to those who want to develop their own extensions.

Clarifications: RFC 6733 is full of helpful clarifications for readers and implementers. The clarifications are the results of many discussions within the working group to reconstruct the original intentions and to match them with implementations in the field.

More details about these differences can be found in Section 1.1.3 of RFC 6733.

Given these changes, we recommend that you look at RFC 6733 even though older implementations focus on RFC 3588. It is important to understand that many implementations will need time to meet the additional requirements outlined in RFC 6733. In particular, the security changes will lead to changes in implementation code. It is hoped that, by the time you read this book, many, if not most, vendors will have conducted interoperability tests and therefore have taken the various clarifications into account.

1.4 What is `freeDiameter`?

`freeDiameter` is an open source implementation of the Diameter protocol. Development on `freeDiameter` was started in 2008 as an academic project with the goals of evaluating and promoting the Diameter protocol as specified by RFC 3588. `freeDiameter` has evolved to follow the revisions of the Diameter protocol in RFC 6733, part of which were introduced as a result of the evaluation started with `freeDiameter`.

`freeDiameter` has been used in commercial Diameter deployments, and it can be used as a reference implementation that anyone developing a commercial Diameter stack can use for interoperability testing. It is also a platform made freely available to researchers and students for prototyping, and for evaluating their ideas for new services built upon Diameter. For these reasons, `freeDiameter` was written in the C language and has been engineered to be as flexible and extensible as possible, with a small system footprint and good performance.

We will use `freeDiameter` throughout this book to illustrate various concepts of the Diameter protocol. By following the hands-on examples in this book, `freeDiameter` will give you a better understanding of Diameter as you configure it to exchange Diameter messages between different nodes. Instructions on setting up `freeDiameter` can be found in Appendix A.

References

1 IETF. The Tao of IETF: A Novice's Guide to the Internet Engineering Task Force, Nov. 2012. http://ietf.org/tao.html.

2 V. Fajardo, J. Arkko, J. Loughney, and G. Zorn. Diameter Base Protocol. RFC 6733, Internet Engineering Task Force, Oct. 2012.

3 D. Durham, J. Boyle, R. Cohen, S. Herzog, R. Rajan, and A. Sastry. The COPS (Common Open Policy Service) Protocol. RFC 2748, Internet Engineering Task Force, Jan. 2000.

4 C. Rigney, S. Willens, A. Rubens, and W. Simpson. Remote Authentication Dial In User Service (RADIUS). RFC 2865, Internet Engineering Task Force, June 2000.

5 B. Aboba, P. Calhoun, S. Glass, T. Hiller, P. McCann, H. Shiino, P. Walsh, G. Zorn, G. Dommety, C. Perkins, B. Patil, D. Mitton, S. Manning, M. Beadles, X. Chen, S. Sivalingham, A. Hameed, M. Munson, S. Jacobs, B. Lim, B. Hirschman, R. Hsu, H. Koo, M. Lipford, E. Campbell, Y. Xu, S. Baba, and E. Jaques. Criteria for Evaluating AAA Protocols for Network Access. RFC 2989, Internet Engineering Task Force, Nov. 2000.

6 IETF. Authentication, Authorization and Accounting (AAA) (Concluded) Working Group, Mar. 2014. http://datatracker.ietf.org/wg/aaa/charter/.

7 D. Mitton, M. St.Johns, S. Barkley, D. Nelson, B. Patil, M. Stevens, and B. Wolff. Authentication, Authorization, and Accounting: Protocol Evaluation. RFC 3127, Internet Engineering Task Force, June 2001.

8 C. Rigney, A. Rubens, W. Simpson, and S. Willens. Remote Authentication Dial In User Service (RADIUS). RFC 2058, Internet Engineering Task Force, Jan. 1997.

9 C. Rigney, A. Rubens, W. Simpson, and S. Willens. Remote Authentication Dial In User Service (RADIUS). RFC 2138, Internet Engineering Task Force, Apr. 1997.

10 S. Kent and K. Seo. Security Architecture for the Internet Protocol. RFC 4301, Internet Engineering Task Force, Dec. 2005.

11 IETF. Remote Authentication Dial-In User Service (RADIUS) (Concluded) Working Group, Mar. 2000. http://datatracker.ietf.org/wg/radius/charter/.

12 IETF. RADIUS EXTensions (RADEXT) Working Group, Mar. 2014. http://datatracker.ietf.org/wg/radext/charter/.

13 A. DeKok. RADIUS over TCP. RFC 6613, Internet Engineering Task Force, May 2012.

14 A. DeKok. Datagram Transport Layer Security (DTLS) as a Transport Layer for RADIUS. RFC 7360, Internet Engineering Task Force, Sept. 2014.

15 S. Winter, M. McCauley, S. Venaas, and K. Wierenga. Transport Layer Security (TLS) Encryption for RADIUS. RFC 6614, Internet Engineering Task Force, May 2012.

16 M. Chiba, G. Dommety, M. Eklund, D. Mitton, and B. Aboba. Dynamic Authorization Extensions to Remote Authentication Dial In User Service (RADIUS). RFC 3576, Internet Engineering Task Force, July 2003.

17 A. DeKok and A. Lior. Remote Authentication Dial In User Service (RADIUS) Protocol Extensions. RFC 6929, Internet Engineering Task Force, Apr. 2013.

18 A. DeKok and G. Weber. RADIUS Design Guidelines. RFC 6158, Internet Engineering Task Force, Mar. 2011.

19 IANA. Radius Types, Jan. 2016. https://www.iana.org/assignments/radius-types/radius-types.xhtml.

20 P. Calhoun, J. Loughney, E. Guttman, G. Zorn, and J. Arkko. Diameter Base Protocol. RFC 3588, Internet Engineering Task Force, Sept. 2003.

21 IETF. Diameter Maintenance and Extensions (DIME), Mar. 2014. http://datatracker.ietf.org/wg/dime/charter/.