https://doi.org/10.52326/ic-ecco.2021/NWC.03

AAA protocol in IoT

Marin Cazac
Technical University of Moldova, 168, Stefan cel Mare Blvd., MD - 2004, Chisinau, Republic of Moldova
Marin Cazac, marin.cazac@ffi.utm.md

***Abstract*—This document is representing not only the description of AAA protocol. It is representing real production problems which I encountered in many years of work with it.**

Each from this problem was taken separately. It was analyzed and the best solution was applied to it.

Article is containing not only analyzing of the problem. In this document the solution described was developed and tested in time on different networks starting with few hundred devices and finishing with tens of thousands of them. On production, solution applied worked for years. Based on this point, I can conclude that problems which were on the production, where solved by applying of those solutions.

In order to understand those problems, is needed to understand how this protocol is working and where are the bottleneck points of it.

Keywords—AAA; Authorization; Authentication; Accounting

I. INTRODUCTION TO AAA

AAA protocol stands for *Authorization*, *Authentication*, and *Accounting*. The description of those three words are:

Authorization: is the process of verifying the privileges or permissions of the authenticated IoT device to access system resources. Some companies, when producing the device, include a security certificate for authentication, and others authenticate them based on the MAC addresses of the device. In this order the IoT device can get authenticated into the system.

Authentication: is the process of verifying the details of an IoT device to identify it and grant access to the system. An IoT device must be network authenticated so that applications can control that device. Authentications are of several kinds.

Accounting: is the process in which the list of options supported by the device is checked. Including transmission and operating speeds according to the licenses set.

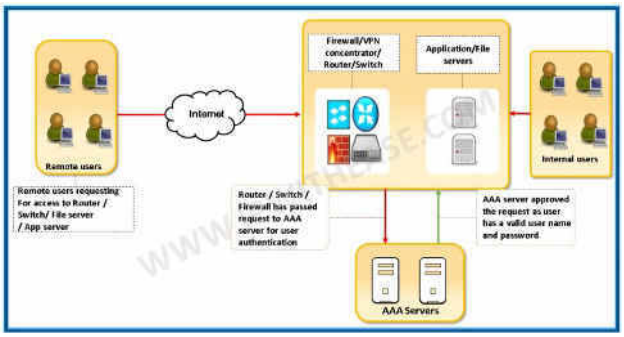

Figure 1. Graphical schema/description of AAA request-reply message

II. WHAT ARE THE MOST IMPORTANT TYPES OF AUTHENTICATION AND AUTHORIZATION THROUGH THE AAA MESSAGE?

A. Authorization of good format of the AAA message

Requires client authentication. Initial step in the authorization process. It means the users or device controlled by user which is sending good structured AAA request to an authorization server. [1]

B. Authorization with specific permissions

Grants the customer access request and can provide specific configuration information needed to start delivering the specific service to the end user. AAA server is interacting with specific module of an application. In a service provider, this module would manage resources and access, and configure the service equipment to provide inly the part of authorized service granted to user. It can also take the decisions itself at the authorization level because it has the application specific knowledge needed for that.

C. Authorizationl log files

For controlling and cheching purposes, the AAA server should have some database where can be stored events which will occur in the AAA server. They will have the time of the event, action proposed, and some additional data. Like MAC address or IP of source which

https://doi.org/10.52326/ic-ecco.2021/NWC.03

sent the message itself. The main point here can be the time of the connection and the traffic or other important data which can be accounted. With help of certificates, this module could support non-repudiation.

D. Database of rules and services

A database which contains all available services in the system and all its resources based on which authorization decisions can be taken and the policy rules to take those decision is also needed. Naming convenstion for services and resources is important as it will be asked from other AAA servers in order to be able to generate complex authorization requests.

E. Forwarding of the AAA messages

As it can be a root domain with multiple subdomains in it, which can create the AAA problem parsing issues, a mechanism to forward AAA messages between authentification servers is required. Finally, communication will be done by perr-to-peer protocol and by having all intermediate resources (AAA subdomains servers) transparently.[2]

III. MULTIPLE DOMAINS IN MULTIPLE TYPE OF SERVICE ARCHITECTURE

Can be distinguished six different models of communication:

A. AAA server can work in 3 different authorization models.

Same AAA server can support all 3 authorization models. They are: agent (or client/subscriber), pull (asking for some specific value on time interval), and push. Idea with this model is, that it can be an intermediate server, that can transfer AAA message from origin to next authentication hop, or to final authenticated server. Due to this, is crucial to have the server working in all 3 authorization models.

B. Main rule of AAA server

The main rule of AAA server knows exactly how to evaluate the structure of AAA message and how to parse the AAA requests.

C. No sophisticated logic is present in its implementation

AAA server has no knowledge what exactly data are containing the AAA message. Due to that, it is just parsing and sending or forwarding data to final device.

D. Forwarding of an AAA messages

There are 3 type of domains. *A*, *B*, and *C*. Each of them has their own naming space problems. In order to solve them is needed to use the forwarding of an AAA messages by locating the right server where message should be forwarded based on rules predefined in the system configuration.

E. Peer-2-peer protocol

Communication of type *A* should be a standart peer-2-peer protocol without distinguish between client and server like a separate roles. It means that all routing till the moment when message will be delivered, will be transparent like a peer-2-peer communication.

F. Services, applications and domains mixed together

AAA protocol can allow information equipment interconnected for accessing multiple services that's belong to multiple applications that can be located in multiple administrative domains. All of those can be combined in a single AAA message. Next picture will put a cleaner view on how it is combined and working:

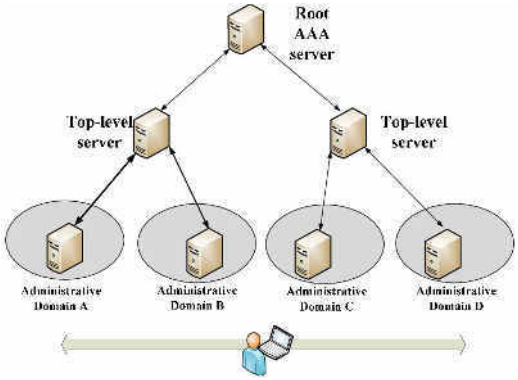

Figure 2. Example of a Multiple Domains and Multiple Type of Requests. (figure caption)

Top-level servers in Figure 2 are representing model *A* of communication. If the AAA request belong directly to Root AAA server, the communication for AAA message will be transparent and communication model used will correspond to point *E*.

IV. WHAT ARE THE MOST COMMON SECURITY RISKS OF THIS PROTOCOL?

Can be distinguished 3 types of most common security risks of this protocol [3]-[7]. They will be described below with 2 sub items for each security risk.

That's assume that there is a number of devices that are interconnected to our application. For some reason, another device is being purchased and trying to authenticate to the network. Given the fact that the number of devices is higher than the number allowed by the application, the last one will not be authenticated in the network. The most common problems are:

A. AAA flood request

The device will send Access-Request (AAA-Request) messages at a configured interval of X seconds, or calculated according to a certain algorithm, which does not exceed 2 minutes. This means for the system a fairly

https://doi.org/10.52326/ic-ecco.2021/NWC.03

large flood of data. Because the application party is required to reply to or ignore this message, this means that this message will be read, processed, analyzed and processed anyway.

B. *Wrong capacity processing by server*

If there are 2 times more devices than the number supported by the initial application, then they can be a real challenge for the server application that needs to authenticate the devices on the network. This large number of messages, the authentication application, may stop responding and shut down shortly. Usually, it can happen at the moment when initial AAA message is containing a certain amount of fields, and, in the time, number of fields are dramatically increased. At this moment of time, new calculations were not performed as all preview calculations were based on old counting, writing rules and limitation done at design processing of the application.

Depending on the construction of the authentication level of the application, and of the final devices, the decommissioning of the authentication and authorization module may lead to the decommissioning of all IoT devices connected to the network. This moment is important because it must be taken into account when creating and designing the network. In general, there are 2 types of problem solving.

C. *IoT devices with RAM memory only*

The device when connected to the network, receives authorization and authentication. This information is stored in its RAM. It will work connected to the network, even if on a regular request it will not receive a positive authentication response, until the device restarts. Then the RAM will be erased, and authentication will be required again.

D. *IoT devices with RAM and ROM memory*

The device when connected to the network, receives authorization and authentication. This information is stored in its ROM memory. It will work connected to the network, even if on a regular request it will not receive a positive authentication response, including after restarting the device. Then the RAM will be erased, but the ROM will keep the data of the last successful authentication. In this case, the removal of the device from the network can only be by sending a direct message to the device, deleting the last authentication data, and blocking it to the subsequent massive transmission of data on the network.

Massive request for authentication of all network elements simultaneously. It is assumed that we have a network that operates according to principle 2 point a) listed above. At some point, there is a power outage, and all devices are disconnected. When connecting power, all devices will require authentication simultaneously from the central device. It may shut down at a high demand for messages. The solutions to avoid the last problem are:

E. *Random generation time*

To configure a random time on the IoT device. The authentication request must be from 5 seconds plus a certain factor that can be calculated according to a certain formula, but not more than 2 or 5 minutes. This factor can still be specified later, if the device can keep this information in the ROM memory.

F. *AAA time to live configured on the queue*

Limiting the request for queue authentication messages. By using a stack, or a queue, and by limiting the number of messages on that queue, as well as their lifetime. For example, the incoming message may have a lifespan of 5 minutes. If the message was not processed, it was automatically removed from the queue and the next authentication request can be processed. If is not exceeded 5 minutes waiting time for the message in the queue, or the message limit per queue is not exceeded.

V. AUTHORIZATION INFORMATION

Authorization decisions needed and services involved in authorization process will be described in this section. [3]-[7]

1. Authorization message at parsing time should be able to take the next information: information about the requestor which can be MAC or IP based, the service/ method asked request, and the operating environment (sometime it can be VxWorks or some configured Ubuntu/Debian distribution, etc). AAA message will be transported via TCP/UDP protocols and specific listener based on destination port will receive the message and will parse it.
2. Final result will be the representation of AAA message information as a sets of attributes. This information can also be grouped in objects.
3. Due to needs of keeping all the data into a single message, appeared the need of packaging authorization information. In such manner sending it to different services or applications which can process is done more easily as it will be a single AAA message which will be sent.
4. Attributes are divided in 2 sides. From 1 side, they should be the standard one. (e.g. identity information, group information, sender information, destination information, ...) and for another side, they can be a specific application or service protocol implementation. In this way, this attribute can be read only by application of a single company or set of products.
5. Authorization decisions can be divided in 2 sides. From one hand it can be based on identity information,

https://doi.org/10.52326/ic-ecco.2021/NWC.03

and for another hand the decision can be based on role access control authorization.

6. Authorization information should specify the number of attributes and the additional AAA message size which will come with the message.

7. Other, non AAA protocols, should be able to add their own attributes and encapsulate it to AAA message. In this way, other protocol information can also reach needed server where their attributes will be parsed. Response of those attributes will come with AAA reply.

8. In some organization system administrators should be able to define their own attribute types. Later on, those attributes will be encapsulated into AAA message. For organizations those attributes can be very important in order to make right authorization decisions.

9. New attribute in the AAA message should be possible to be defined without including a central administration authority. Like a subdomain in entire domain may use its proper AAA plus new attributes.

10. For newest created attributes in the AAA message should be possible to define their own types. Based on this, and instance of an attribute can lead to different values. Like an enumerated value.

11. It can be possible to get different instances on the same AAA attribute based on its enumerated type or value. Each domain, sub-domain or authorization authority can have a different instance of same attribute.

12. Authorization decisions are taken based on some rules. AAA protocols should come together with a mechanism for updating the rules in order to be able to control authorization decisions.

13. In authorization decision can be a chain of AAA entities. It can happen when a subdomain architecture is build. In this case, first subdomain server can forward this request to root domain.

14. Intermediate AAA entities should be able to add their own local authorization attributes information to a AAA request or response. In this way will be possible to track the source of AAA message and to know where the response should be sent back.

15. Sometime can be required to have a separate deploy of AAA message entity. Or the deploy can be integrated with application entities. AAA endpoints can build their own AAA message. It can happen in MESH network configuration. In this case, 2 endpoint which exchange information directly.

16. Creation and encoding of rules that should be active and processed by the AAA server accordingly to the attributes given by another AAA server should be supported in AAA protocol. Depending on level of authorization of AAA attributes, the requesting AAA Server should be able to view those attributes or not. It can be done in the process of Marchalising the AAA message in different was. Since the data can be sent in a raw format, it is very important that it can be read and well parsed in other language. Due to that, a specific parser can be used in correspondent way by taking into account version number or flag indicated in AAA message.

17. Can be distinguished 2 type attributes: critical and non-critical. IE: There are critical information without it the AAA message cannot be recognized, and additional information which can come as part of additional functional, feature implementation, etc.

18. Authorization rules can be interpreted based of combinations of other authorization rules which were already evaluated. Important thing is member of one AAA group cannot access the resources of another member.

19Authorization decisions can be based on the geographic location of a requestor or AAA entity. In this way a new development logic is required to be implemented. From one hand is needed to add a specific geo location for this device, and from other hand, is needed to add an additional field for AAA message for subdomain forwarding.

20. Authorization decisions should be based on the identity (which can include a TLS/SSL certificate) or the equipment itself which is used by the requestor (can be identified based on MAC address), service(authentication one which can work in mode described in point *III. A*) or an AAA entity.

21. In case of having multiple instances of a specific attribute, there should be an unambiguous mechanism where a receiving peer can determine the value of specified instance. In this manner, marchalising parsing and versioning (sometime including gateway information) should be implemented. It has a correspondent order per version, so in that case mapping can be done accordingly.

VI. SECURITY OF AUTHORIZATION INFORMATION

Authorization information should go in secure way to its destination. It is very important aspect as it should assure that no repudiation attack or spoofing attack can take place. [3]-[7]

1. Authorization information should be securely communicated in AAA and application protocols. Algorithm which is responding for authenticity, integrity and privacy for this information should be specified.

2. Authorization information can be sent using AAA protocols in different security levels. From less secure mechanisms for data integrity/confidentiality to highly secure.

3. The security requirements can differ between different parts of a package of authorization information. This point can be interpreted as getting access to the network or to some resources. In case of network access, data can be sent in open format without any encryption. However, using access to some specific resources, is requiring to send the data only in encrypted format. As open format is not recognized by the resource which should grant access to some specific services or resources.

https://doi.org/10.52326/ic-ecco.2021/NWC.03

4. In AAA protocols should be implemented mechanisms and algorithms that will prevent intermediate administrators breaching security. Man-in-the-middle attacks, where an intermediate administrator can changes AAA messages on the fly should be prevented.

5. AAA protocols should have implemented on receiving and answering to a host for not more that few retries per minute. This parameter can be configurable on algorithm based. It has the meaning of not allowing flooding attacks where the attacker can send a lot of AAA messages till the moment when it will be detected and blocked by the system. This kind of attack is causing waste of CPU and RAM memory resources.

6. AAA protocols should be capable of communicating secure less in the network where the security was already implemented. Like VPN tunneling or IPSec. In that case, securing again over secured connection can be resource waiting.

7. Trusting peer-2-peer confidentiality, integrity, authentication, and non-repudiation can be required for packages of authorization information. It can be used by sending AAA information for root domain, and the subdomain at processing time, can remove from this message some data. At this point root domain will get an "easy" message which can be parsed quickly. Such type of messages is using less CPU or memory resources.

8. AAA protocols can be used in networks where no peer entity authentication is required (e.g. a network address which is inside a secure LAN can be enough). It will ask anyway for secure AAA message. This point is the opposite of point 6. However, it is saying that we are already in a secure environment, so no reason to secure again those data.

9. All AAA messages should be encrypted by default for all protocol options. Implementations of AAA entities (agent, pull and push) should use the secure defaults unless otherwise configured/administrated by the administrator of the resource. This idea can change in different organizations, but the principles will remain the same. Interpretation of "secure" can change based on organization rules and privacy levels.

REFERENCES

[1] S. Farrell, J. Vollbrecht, Generic AAA Architecture "*Introduction to AAA – Authorization, Authentication and Accounting*". Image based on RFC 2903. August 2000, pp.10-22

[2] P. Calhoun, L. Gommans, G. Gross, "*AAA Authorization Requirements*" RFC 2906, 2000, pp.2–22.

[3] Farrell, S., Vollbrecht, J., Calhoun, P., Gommans, L., Gross, G., de Bruijn, B., de Laat, C., Holdrege, M. and D. Spence, "*AAA Authorization Requirements*", RFC 2906, August 2000.

[4] Vollbrecht, J., Calhoun, P., Farrell, S., Gommans, L., Gross, G., de Bruijn, B., de Laat, D., Holdrege, M. and D. Spence, "*AAA Authorization Framework*", RFC 2904, August 2000.

[5] Vollbrecht, J., Calhoun, P., Farrell, S., Gommans, L., Gross, G., de Bruijn, B., de Laat, C., Holdrege, M. and D. Spence, "*AAA Authorization Application Examples*", RFC 2905, August 2000.

[6] Bradner, S., "*The Internet Standards Process -- Revision 3*", BCP 9, RFC 2026, October 1996.

[7] Blaze, M., Feigenbaum, J., Ioannidis, J. and A. Keromytis, "*The KeyNote Trust-Management System Version 2*", RFC 2704, September 1999.