C-BAS: Certificate-based AAA for SDN Experimental Facilities

Umar Toseef, Adel Zaalouk, Tom Rothe, Matthew Broadbent, Kostas Pentikousis
EICT GmbH, Berlin, Germany
Email: [umar.toseef, adel.zalouk, tom.rothe, matthew.broadbent, k.pentikousis]@eict.de

***Abstract*—Efficient authentication, authorization, and accounting (AAA) management mechanisms will be key for the widespread adoption of SDN experimentation facilities beyond the confines of academic labs. In particular, we are interested in a robust AAA infrastructure to identify experimenters, police their actions based on the associated roles, facilitate secure resource sharing, and provide for detailed accountability. Currently, however, said facilities are forced to employ a patchy AAA infrastructure which lacks several of the aforementioned features. This paper proposes a certificate-based AAA architecture for SDN experimental facilities, which is by design both secure and flexible. As this work is implementation-driven and aims for a short deployment cycle in current facilities, we also outline a credible migration path which we are currently pursuing actively.**

I. Introduction

Innovation is the process of transforming an idea into tangible solutions. In network R&D, transforming a research idea into a deployed solution traditionally requires years. In response to this conundrum, several experimental facilities (EFs) were established aiming to foster large-scale experiments in a near real-world network environments. EFs employ virtualization and enable computational and network resource sharing over the same physical topology. Since resources are finite, they should only be granted to approve experimenters. Resource use should be policed according to authorization levels throughout the lifecycle of an experiment and detailed accounting records should be easy to maintain. This is realized through an authentication, authorization and accounting (AAA) architecture, which is present (at varying degrees of implementation sophistication) in all EFs we surveyed.

In particular for SDN EFs (SEFs) such as GENI and OFELIA [1], several control frameworks were introduced dealing with AAA as we discuss next. However, currently-deployed AAA mechanisms suffer from drawbacks such as, tight-coupling of AAA mechanisms with the implementation of the SEF architecture; little or no regard for reusability (i.e., one AAA architecture cannot be reused by a different SEF); and no support for a standard access interface between the experimental network and the AAA architecture. To address these drawbacks, we introduce certificate-based AAA in SEFs (C-BAS), which provides loose-coupling between the AAA architecture and the experimental network, it is reusable, and delivers AAA services through a well-defined interface, which ensures consistency and compliance between a SEF and the AAA architecture.

The rest of this paper is structured as follows. Sections II and III discuss the state of the art in this area. Section IV details C-BAS design, implementation, and migration path. Section V concludes the paper and lists future work items.

II. Related Work

AAA has been the focus of much research for decades. Several protocols and services have been developed to address one or more of the aforementioned functionalities. Kerberos [2], for example, was developed as a network authentication protocol and became a de facto method to authenticate client/server entities over an insecure network. Since its inception, Kerberos has been deployed widely. Over time, however, requirements shifted to supporting a more diverse set of resources than what Kerberos was originally intended for. This is particularly the case in SEFs, where resources are seamlessly combined to provide provisioning, access and administration for experimenters. Furthermore, the Lightweight Directory Access Protocol (LDAP) [3], and services supporting the protocol, are often used for authentication (AuthN). Together, they provide a method by which to retrieve information stored in a central location. Often used by email providers, LDAP can prove a powerful tool when used with access permissions. This allows an administration to restrict the information available in a response, depending on the requesting user. However, LDAP servers do not readily support user authorization (AuthZ) for actions on external resources. Although it is technically feasible to provide this functionality in a SEF, it leads to a cumbersome and time-consuming process to maintain all necessary information up-to-date. This is particularly pertinent when you consider that it is becoming the norm for facilities to grow and diversify over time; maintaining state and synchrony between the LDAP server and reality fast becomes a challenge.

Considering these two approaches, it emerges that a better method is necessary to meet the unique needs of a modern SEF, where resources are diverse in nature and no longer geographically collocated. Certificates, for instance, have the advantage that they readily enable SEF operators to make assertions as to which actions are permissible for a given user. This includes the ability to authoritatively verify these permissions on a remote resource. More importantly, a certificate-based approach does not mandate a single authoritative source of trust, such as a Kerberos or LDAP server: a certificate can be validated on any number of trust anchors. An assumption of trust, particularly from a single point of origin, is no longer satisfactory in SEFs which often span countries and continents [4], [5]. A certificate-based approach removes this dependency.

Since ease-of-use and availability are key features in drawing users to a facility, it is important for SEF operators to include resources in a speedy and trouble-free manner. Pushing existing technologies to their limit, novel techniques are required in order to integrate resources in a scalable manner. Partnered with the trend in providing a diverse set of resources and equipment, such as optical devices and non-

OpenFlow hardware [6], results in the need to extend the web of trust to devices that have never been previously collocated in the same SEF. Each resource has its own unique set of requirements, and a static approach is no longer feasible.

A number of SEFs were established in recent years, each with its own AAA processes. For example, Expedient [7] was originally developed as a centralized pluggable clearinghouse for GENI. Similarly, the OFELIA Control Framework (OCF), a derivative of Expedient, was developed for the OFELIA SEF. Both approaches gave no or little consideration for future evolution in AAA. For example, OCF uses password-based LDAP as the main mechanism for AuthN, making it hard to switch to another mechanism without requiring a complete upgrade for the whole AAA architecture. C-BAS, on the other hand, defines a pragmatic deployment strategy that can support multiple AuthN mechanisms.

III. Priliminaries

Authentication (AuthN) is the process of verifying whether an entity is in fact who or what it claims to be. A certificate is a digitally-signed document which authenticate the identity of an entity such as, for example, a user, a piece of software, or a server. A certificate asserts the ownership of a public key to the named subject (owner) of the certificate. Certificates are typically issued by a third party called the Certification Authority (CA), which is trusted by both the owner of the certificate and the entity relying upon the certificate. This implies that if an entity trusts the issuing CA, it will also rely upon the identity assertion made by the private key that corresponds to the public key of this certificate. This lays the foundation of certificate-based authentication.

X.509 [8] is the most widely accepted standard for digital certificates. Each X.509 certificate has several information fields, including the distinguished name (DN) of the issuing CA, validity period, subject DN and its public key and, most importantly, the digital signature of the issuing CA. A digital signature is basically a hash of the data fields of the certificate encrypted with the signer's private key. The CA digital signatures are included in a certificate to ensure the integrity of its contents which involves the process of decrypting the digital signature using the CA public key and matching the data with the self-computed hash of the certificate contents. X.509 allows a CA to revoke issued certificates as a solution to problems such as the loss of the private key corresponding to the certificate. For this purpose, a revocation list is maintained and published by the CA.

An important step in the process of certificate-based AuthN is to validate the presented certificate. For a certificate to be considered valid, (i) the issuing CA must be trustworthy for the relying party, (ii) the certificate must be valid at verification time, and (iii) the certificate ought not to be revoked by the issuing CA. If all three conditions are met, the entity provides a proof of being the legitimate owner of the presented certificate. In other words, the proof of possessing the private key corresponding to subject's public key in the certificate must be presented. This process is depicted in Fig. 1. The user provides his private key to the user-agent (Fig. 1:A), and the relying server provides some known data to the user-agent which is encrypted using the private key (Fig. 1:B) and is sent back to the relying party (Fig. 1:C). If this encrypted piece of data (named evidence) can be successfully decrypted using the subject's public key, it verifies the entity's possession of the

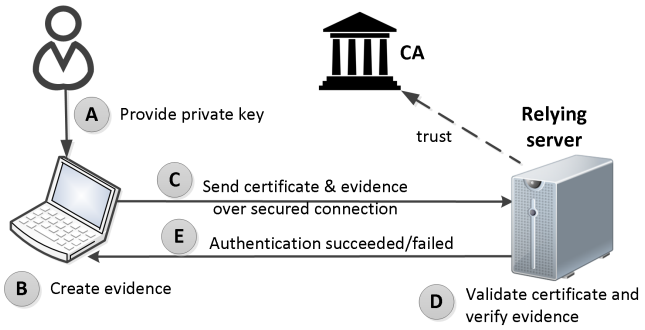

Fig. 1. Certificate-based identity verification/authentication

private key (Fig. 1:D) and hence the claimed identity of the entity as asserted by the certificate is also verified (Fig. 1:E).

AuthN alone is not sufficient to grant system access to a user and must be complemented by an authorization (AuthZ) process by which someone or something is allowed to perform certain operations or access particular information. AuthZ can be performed either by using a directory service like LDAP [3] where user privileges are listed against his identity or through digitally signed documents (credentials) that list user privileges on a target object. In principle, credentials are similar to certificates except that credentials assert privileges while certificates assert identities. A credential document contains the identity of the owner (the entity whose permissions are being specified) and the target (the entity on which the permissions are being specified), the validity period, list of privileges as well as the digital signatures of the issuing authority. If a certificate-based AuthN is used, it is advantageous to use an entity's certificate as part its identity in the credentials. This way, the identity of both owner and target can be reliably verified.

In the context of SEFs such as GENI and OFELIA, Slice Federation Architecture (SFA) credentials and Attribute-Based Access Control (ABAC) credentials have been considered. The SFA format [9] is rather simple and supports a table-driven mapping mechanism between attributes and allowable actions. ABAC is an extensible credential format which belongs to a scalable AuthN system based on formal logic [9]. ABAC credentials provide a mapping of principals to attributes for AuthN purposes and, further, support trust delegation statements and a reasoning engine to determine whether a given entity is trusted to take a particular action.

In SEF nomenclature, collections of managed resources are referred to as *aggregates* [10], while the experimenters using the facility are called *members* [11]. *Authorities* [11] are the services that manage assertions about members and their permissions with respect to aggregate resources. Members typically employ *user-agents*, i.e. software tools which interface with aggregates and authorities on behalf of members. For practical purposes it makes sense to group resources at aggregates as so-called *slices* [10]. A *sliver* [10] is a portion of a resource that has been granted to a member (in the context of a slice). Last but not least, *projects* [10] are groupings of slices and members for a particular administrative purpose.

As an example, consider Fig. 2 which illustrates member roles in a SEF. Member roles determine the access level (or privileges) of a person or entity. The member roles as per [9] are: (1) The *Project Lead* who is the owner and principal contact regarding all activities on a project or slice. There is

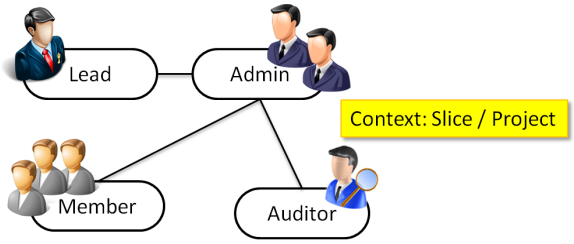

Fig. 2. Member role hierarchy

only one Lead per project/slice. (2) The *Project Admin* who is authorized to modify project/slice attributes as well as change the role of other members except the Lead. (3) A *Member* is a user that has read-write access on the entities managed by that group. (4) The *Project Auditor* is a member of the group with read-only privileges to monitor group activities.

Recently, efforts have been made to create and standardize a well-documented API for AAA operations that are common in all SEFs [11]. As such, modifications are necessary so that existing facilities, such as GENI and OFELIA. What is more important, however, is that the new API also provides a framework for new facilities, particularly those that are still developing their AuthN and AuthZ infrastructure, such as the ALIEN test-bed islands [6]. It is in these new facilities that a common AAA architecture, and in particular a shared implementation, would be the most beneficial.

Fig. 3 illustrates an example of a username-and-password based AuthN and AuthZ architecture [12]. A *Registration Portal* allows users to create accounts for accessing the SEF. A directory service, e.g. employing an LDAP server, stores user registration information including usernames and passwords. A *user-agent* enables a member to manage and create experiments, and *Aggregate Managers* manage the underlying compute and network resources. Since LDAP is used to store user credentials (i.e., username/password), the LDAP server becomes the primary point of contact for AuthN. For example, the VPN server and user-agent use an LDAP interface to verify the user login information. Similarly, virtual machines (or compute resources) instantiated by the users using a virtualization Aggregate Manager also perform AuthN via LDAP. Since this architecture lacks a proper ClearingHouse (cf. §IV-A), all slice information (e.g., members, expiration time, registered slivers etc.) is maintained in a database accessible to the user-agent.

An issue that has so far prevented SEFs from reusing existing AAA functionality is that it is often tightly coupled to the testbed itself. For example, OCF uses an LDAP server for credential management. However, the same credentials (and thus the use of the LDAP server) are also used in VPN AuthN necessary to connect to the SEF. The following section presents C-BAS, an AAA architecture capable of serving a multitude of different SEFs, free of implementation-specific details. As we explain later, C-BAS is suitable for migrating existing SEFs and for inclusion into ongoing efforts such as ALIEN islands.

IV. C-BAS: Certificate-based AAA for SDN

In contrast to the AAA architecture of Fig. 3, C-BAS (`www.eict.de/c-bas`) introduces a flexible system which can address most of the member needs without compromising its well-defined structure for AAA. For example, sometimes a member may want to temporarily delegate his privileges to another trusted member, entity or tool. Such requirements cannot be fulfilled without dealing with a long list of complexities in a username-and-password AuthN scheme. We seek a solution for such problems at three levels of granularity as described below (see Fig. 4).

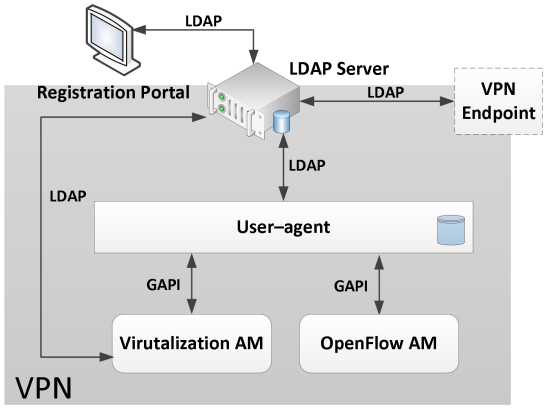

Fig. 3. An example of username-and-password based authentication and authorization architecture. The user agent communicates with the virtualization and OpenFlow aggregate managers (AMs) using GAPI [13].

Delegated credential: A user (delegator) can bestow all or parts of his privileges to another user (delegatee) by providing him with the delegate credentials. The delegated credentials are then used by the delegatee along with his identity to get himself authorized. In this case, only the delegatee is held accountable for his actions. The delegator is not accountable for delegatee actions. Delegated credentials must be digitally signed by the delegator.

Speaks-for credential: This is primarily intended for supporting tools that "speak-for" their users. Speaks-for credentials allow a tool to get authorized using the user's privileges. Though the user is accountable for each action taken, the information about the tool used is also logged in the system. This way, speaks-for credentials allow for tracing which tool was used to perform any operation on the experimental facility.

Speaks-as user: If speaks-for credentials are not supported by a tool or user-agent, then the speaks-as concept can be employed. For this purpose, the user will have to provide his certificate and the private key to the user-agent. The private key is required by the user-agent to digitally sign the user's request so that they can appear as if they are directly coming from the user. In this case, the user is accountable for each action taken and is also logged as the action-taker. Moreover, the information about the user-agent is not logged.

In Fig. 4:(a) and :(b), the delegatee user and the speaks-for capable user-agent provide their own identities along with the credentials to act on the user behalf. This means they are authenticated based on their supplied identities while authorization takes place based on delegated / speaks-for credentials. However, in Fig. 4:(c) the user-agent has no identity of its own and therefore supplies the user identity to authenticate and authorize itself as the operating user.

We consider two essential requirements in the design of the C-BAS architecture. First, C-BAS must offer a robust, secure and extensible certificate-based AAA mechanism. Second, we need to define a smooth migration path for the username-and-password based AAA mechanisms deployed in existing SDN experimental facilities.

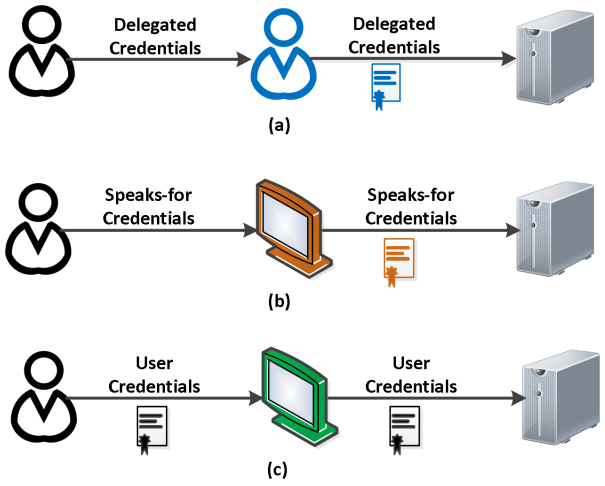

Fig. 4. Delegation of privileges. (a) delegated credentials, (b) speaks-for credentials, (c) speaks-as user

A. ClearingHouse Services

The C-BAS ClearingHouse (CH) comprises a set of related services supporting AAA operations in a SEF. CH is also a central location to lookup information about members, slices and other available services in the testbed. CH services can be categorized into three groups: (1) Registration and management services which provide a lookup for available services in the facility and facilitate the registration of new members, projects and slice objects. (2) AuthN and AuthZ services that manage the credentials of all SEF entities and enforce predefined policies. (3) Accountability services which keep track of all transactions. CH services are offered with the help of the following functions and authorities.

The **Member Authority** (MA) is responsible for managing and asserting user attributes. It generates member certificates and credentials, which specify the attributes and roles associated with a member, while the certificate identifies the member within the credentials. The MA maintains a database of registered members and their associated information including, but not limited to, certificates and credentials, SSH (Secure Shell) and SSL (Secure Sockets Layer) keys as well as the human readable identity information like real name, institute, contact details and so on. MA is a central location to lookup and modify member information as well as register new members. In addition, MA should also maintain a Certificate Revocation List (CRL) which can be accessed by other entities for member certificate verification as explained in Section III.

The **Slice Authority** (SA) creates and manages slice objects and the associated member credentials (called slice credentials). Slice credentials map member roles and privileges on a slice object, i.e., slice credentials authorize user actions at aggregates within a slice context. SA enables looking up slice credentials and supporting slice object operations such as “modify”, “renew”, “delete” and so on. The **Project Service** (PS) maintains a list of existing SEF projects and asserts the member roles. The PS service can be SA-hosted and provides access for creating, looking up, updating, and deleting projects.

The **Service Registry** (SR) serves as the primary network contact point as it keeps a record of all available registered services such as SA and MA and offers their URIs.

The **Logging Service** (LS) provides for accountability by storing the transaction details between user-agents and aggregate managers. LS provides traceability between the slice and slivers and can be complemented with information such as the slice-associated project and Lead for full accountability.

The aforementioned CH services should be accessed via the Common Federation API (FAPI) [11]. The user-agents and AMs are to communicate with the CH through XML-RPC calls over a secured connection (SSL). By supporting FAPI, the CH makes itself compatible with command line interface user-agents like Omni which are also capable of communicating with the aggregate managers using the GENI Aggregate Manager API [13].

B. Migration to C-BAS

We take CH as the main entity to provide all kinds of credentials and related information (e.g., slice, project, member info). Since the CH has an access interface based on FAPI, and no native support for LDAP, we face a migration problem for those entities in a SEF which can only communicate using LDAP. We see two options in order to offer a viable migration path for such entities. First, one could implement LDAP interface support in the CH so that legacy entities can interact with the CH. Although this promises a smooth migration path, implementing the LDAP interface at the CH is disproportionately complex and time-consuming. Alternatively, the CH can be introduced to the existing architecture (Fig. 3) as an add-on. This implies that until migration is completed, the two AA mechanisms will co-exist. The member registration application will be updated to send user registration data to both LDAP and CH. This allows legacy SEF entities to use LDAP for AuthN during the migration phase. The latter option is more appealing implementation-wise and allows existing username-and-password AuthN to work without interruptions until all entities (i.e., AMs, user-agents) implement the required interfaces to the C-BAS CH. In the remainder we consider this later option as a reference for discussion.

C. AuthN/AuthZ via User-agents

C-BAS uses certificates for AuthN/AuthZ that must be issued by a trusted CA which can be a service hosted by the CH. CH owns a self-issued certificate, termed as root certificate, which is trusted by other network entities. Any SEF entity that wants to verify the certificates and credentials issued by the CH (or its authorities) must have the root certificate of the CH. For this purpose, the CH root certificate(s) is pre-installed within the AMs and some user-agents to enable them to verify the credential issued by their trusted CH. Moreover, if the user-agent is partly or fully located in the experiment facility, it may also acquire its own identity (a certificate) from the CH in order to act as a speaks-for agent. We identify four ways (see Fig. 5) to enable AuthN and AuthZ via user-agents.

First, if a user-agent can support only the LDAP interface (cf. IV-B), then a user has to provide its username-and-password for the logging in (Fig. 5:1). These credentials are checked by the user-agent with the help of the LDAP server. Once logged-in successfully, the user-agent interacts with the AMs without passing them the user credentials. This is because a trust relationship is assumed between the user-agent and the AMs. Moreover, the user-agent cannot interact with the SA/MA in the CH and therefore accesses a locally maintained database of slice information. Such user-agents must be completely hosted by the SEF and expose only a web-based interface to the user.

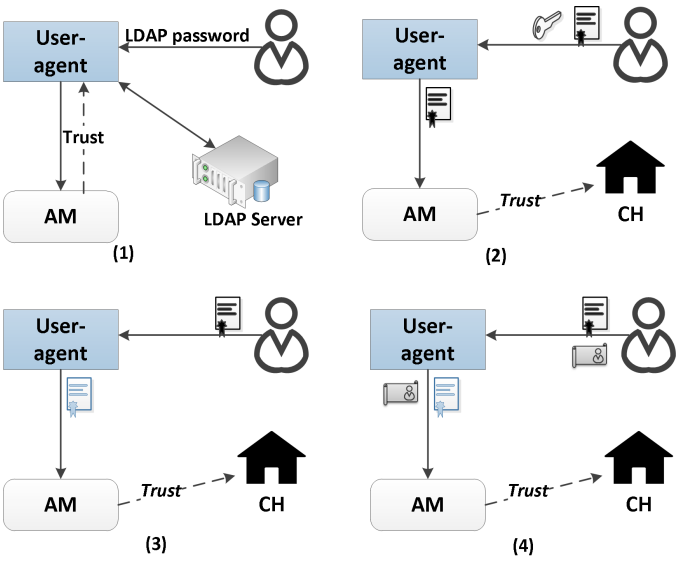

Fig. 5. Illustration of different user login options and user-agent communication with the AMs

Second, the user-agent accepts the user certificate and his private key during the configuration/initialization process and acts as a speaks-as tool (Fig. 5:2). The user certificate is used as the user's identity while his private key is used to digitally sign all operation requests before sending them to AMs. This approach is currently employed by a popular user-agent [14].

Third, the user agent expects the user certificate as his identity and verifies the ownership of that certificate through a challenge-and-response process (Fig. 5:3). This is essentially similar to Fig. 1. Once the user is authenticated, the user-agent acts similar to a speaks-for tool although speaks-for credentials are not required from the user. This is a better option as the user does not provide private information during login.

Fourth, the user-agent authenticates the user following the mechanism described in the third option but additionally it requires speaks-for credentials to act properly on the user's behalf (Fig. 5:4). This is a secure and elegant solution which facilitates full accountability, and is employed in C-BAS by default. In addition, for backward compatibility with legacy user-agents (cf. IV-B), C-BAS also supports the first option.

D. Slice Database Synchronization

A C-BAS compatible user-agent communicates slice-related information to the CH where the SA maintains a database for such purposes. However, as mentioned above, the CH will support legacy user-agents which can only interact using LDAP. This implies that for such user-agents a separate database of slice information will have to be maintained. This leads to a situation where two non-synchronized databases may co-exist to store slice information: (1) the database maintained by the SA inside the CH and (2) the database maintained for the legacy user-agent. This can lead to race conditions such as using the legacy user-agent to create a slice which has already been created using a C-BAS compatible user-agent. In order to provide a remedy for such problems, it is recommended that a legacy user-agent should notify the CH about the creation/update operation of any slice. This would be an attempt to achieve a minimum level of synchronization between the two databases and should be used during the migration phase only. This is illustrated in Fig. 6 in which the legacy user-agent sends a notification about the creation/update of a slice.

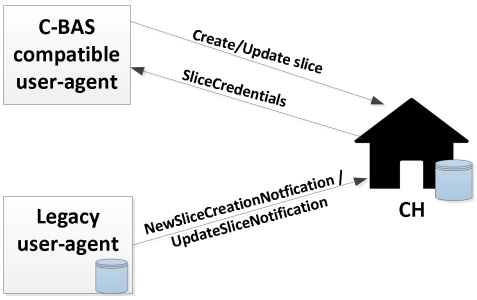

Fig. 6. Illustration of operations on slices by C-BAS and legacy user-agents

E. Certificate Chains and SSH Keys

MA and SA are expected to create and digitally sign the member and slice certificates and credentials. For this purpose the MA and the SA must be provided with the certificates which can be used to sign these digital documents. Such a certificate can be provided in the following ways: (i) there exists one root certificate for the CH which is also installed at the MA / SA as a root certificate. (ii) the CH root certificate is used to create authority certificates for the MA and the SA. This way, the MA and the SA use their own certificates to create member/slice certificates and credentials. Option (i) is simpler as it requires the installation of only one root certificate at the AMs and the user-agents for the verification of member and slice credentials. On the other hand, option (ii), which is adopted in C-BAS, is more elegant as each authority has its own identification in terms of certificate authority, and allows tracebacks for determining which particular MA / SA issued a certain certificate or credentials.

C-BAS needs to offer the user's public SSH key to the AMs in order to allow user login at the SEF and virtual machines. The user SSH keys can either be generated during member registration or at the first time resources are reserved for a member. A simple way would be to create SSH keys at registration time and store them both at the CH and on the LDAP server. In addition, the user may also be allowed to provide his public SSH key during the registration process. In C-BAS we have opted to store the user public SSH key during the registration process either by accepting it from the user or by generating the key pair for the user.

F. C-BAS in a Nutshell

Fig. 7 illustrates the proposed C-BAS architecture. Compared to Fig. 3, C-BAS introduces two new functional entities, namely the CH for managing user and slice certificates and credentials and a new registration portal which communicates user registration data both to the CH and the (legacy) LDAP server. It can be noticed that legacy user-agent relies on LDAP interface for user authentication while C-BAS compatible user-agent performs such operations using FAPI interface of the CH. As mentioned earlier, GENI supports two credential formats, i.e., SFA and ABAC. C-BAS supports the SFA credential format as mandatory but also supports the alternative ABAC format. As GENI moves towards depreciating SFA in favor of ABAC as it adopts a simpler credentials format and at the same time provides support for complex delegations [15], it is

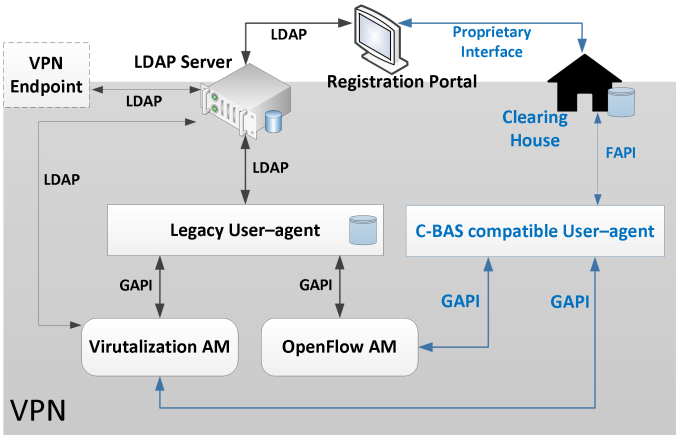

Fig. 7. C-BAS slice operations and migration path for legacy user-agents

envisaged that C-BAS will follow the same migration path. In the future, this will result in a common credential format and support federation across worldwide SEFs.

In C-BAS, all entities must establish an SSL connection to interact with the CH. For this purpose, these network entities are provided with the certificates issued by the CH during the bootstraping process. The process of issuing a certificate to the network entities and user-agents is not dynamic and must be performed by the CH administrator. User registration is performed by an application in the registration portal. As part of the registration process, the user may provide his existing public SSH key or let the registration application generate a SSH key pair for him. The user public SSH key along with his registration information is sent to the MA in first step. The MA creates the member certificate and credentials, stores them in its database, and sends the created user certificate back to the registration application. In the second step, the registration application communicates the registration information along with the public SSH key to the LDAP server that creates user registration record and stores the associated received information. After successful user registration, the registration application makes the user certificate available for user download. This certificate might be required by the user-agent for user authentication as explained in §IV-C.

The SEF member now has the option to use either the legacy user-agent which authenticates his with the help of the LDAP server or the C-BAS compatible user-agent which requires the authentication information provided to the user during the registration process as explained in §IV-C. If a legacy user-agent is used, the information about the slices is stored in a database maintained for legacy user-agents. Moreover, the virtual machines will also authenticate such user using the LDAP server. If a C-BAS compatible user-agent is preferred by the user, it fetches the user credentials from the MA and passes them to the SA when requesting a slice creation operation. The SA authorizes the request based on the provided member credentials, creates the slice and also the associated slice certificate and credentials. This information is stored in the SA database and the slice credentials are provided back to the user-agent. These slice credentials are then provided at the aggregate managers to authorize the user operation on this slice, for example, adding slivers to the slice. A slice has an expiration time before which it must be renewed or it will be deleted by the SA. In addition to creating or updating a slice, the SA also provides the lookup function. For this purpose, the user-agent must provide the user credentials to lookup slices associated with that user. Similarly, when an aggregate manager demands the public SSH key of the user to setup his login, for example, in his instantiated virtual machine, the user-agent requests its user's public SSH key from the MA by providing the user credentials.

V. Conclusion and Future Work

We introduced C-BAS, a certificate-based AAA architecture for SDN experimental facilities. After surveying the current state of the art in deployed AAA solutions for SEFs, we identified specific drawbacks as well as the emerging requirements for AAA architecture in such experimental facilities and explained how legacy solutions cannot be relied upon for further development of SEFs. Based on this discussion, the foundations of C-BAS were introduced. C-BAS has several distinct features which include its well-structured privilege system with the ability to cover a wide variety of use cases, its general design that makes it reusable for other similar experimental facilities, as well as its flexibility for extensions to support future evolution and customizations to address new needs. A functional description of the proposed architecture entities and essential workflow were also provided. Finally, this paper also serves as a guide for a seamless migration path for existing SDN experimental facilities which employ username-and-password authentication mechanisms.

Acknowledgment

This work was conducted within the framework of the FP7 ALIEN and FELIX projects, which are partially funded by the Commission of the European Union.

References

[1] M. Sune, et al., "Design and implementation of the OFELIA FP7 facility: The European OpenFlow testbed," *Computer Networks*, 2014.

[2] B.C. Neuman, et al., "Kerberos: an authentication service for computer networks," *IEEE Communications Magazine*, Sept. 1994.

[3] "Lightweight Directory Access Protocol (LDAP): The Protocol," IETF RFC 4511, Jun. 2006.

[4] G. Carrozzo, et al., "Large-scale SDN experiments in federated environments," in *Proc. SACONET WOSDN*, March 2014.

[5] R. Krzywania, et al., "Experiment Use Cases and Requirements," http://www.ict-felix.eu, FELIX Deliverable D2.1, Sept. 2013.

[6] D. Parniewicz, et al., "Design and Implementation of an OpenFlow Hardware Abstraction Layer," in *Proc. SIGCOMM DCC*, Aug. 2014.

[7] J. Naous, et al., "Expedient: A centralized pluggable clearinghouse to manage geni experiments," Jan. 2010.

[8] ITU-T Recommendation X.509, "The Directory: Public-key and attribute certificate frameworks," Version 3, Oct. 2012.

[9] "GENI: Global Environment for Network Innovations," http://www.geni.net.

[10] "GENI key Concepts," http://groups.geni.net/geni/wiki/GENIConcepts.

[11] "Common Federation API," http://groups.geni.net/geni/wiki/ CommonFederationAPIv2, Nov. 2013.

[12] "OFELIA Control Framework (OCF)," http://fp7-ofelia.github.io/ocf/.

[13] "The GENI Aggregate Manager API," http://groups.geni.net/geni/wiki/GeniApi, 2013.

[14] "The Omni client," http://trac.gpolab.bbn.com/gcf/wiki/Omni, 2014.

[15] N. Li, et al., "Design of a role-based trust-management framework," in *IEEE Symposium on Security and Privacy*, 2002.