Exploiting Proxy-Based Federated Identity Management in Wireless Roaming Access

Diana Berbecaru, Antonio Lioy, and Marco Domenico Aime

Politecnico di Torino, Dip. di Automatica e Informatica
Corso Duca degli Abruzzi 24, 10129, Torino, Italy

Abstract. Federated Identity Management technologies are exploited for user authentication in a number of network services but their usage may conflict with security restrictions imposed in a specific domain. We considered a specific case (roaming wireless access for guests) and extended the Stork SAML-based identity federation to cope with this problem by adding dynamic data, called meta-attributes, to be used for authorization even before the user authentication is completed. This concept may be easily extended to other data needed for trust verification and complex authorization decisions in a federated environment.

1 Introduction

Federated identity management (FIM) is a set of technologies and processes that let computer systems dynamically distribute identity information and delegate identity tasks across security domains [1]. By using FIM, web applications can offer cross-domain single sign-on (SSO), so that a user can authenticate once at the so-called Identity Provider (IDP) and then gain access to protected resources and services at a Service Provider (SP), provided the two organizations established a so-called "Circle of Trust". In SSO, data about identification, authentication and user attributes flow from the IDP to the SP, where it can be used in complex authorization decisions.

In a proxy-based FIM infrastructure [2], a network of *federation bridges* or *proxies* is built, where the authentication and attribute flow from the IDP to the SP through the proxies in the corresponding domains. This improves scalability because to add a new security domain only the proxies must be reconfigured with the location of the new foreign proxy, and to add a new IDP only the proxy in its domain must be reconfigured.

Problem Statement. In a proxy-based FIM scenario, we consider the case of a user and SP both placed inside a company network protected by a border firewall, whereas the user's IDP is in a foreign network domain. To let this user reach his IDP, the firewall must be configured with appropriate permissions: if the IDP is known "a priori" (or is not subject to frequent change) then it can be statically configured at the firewall, but if the IDP is unknown, or changes frequently then we have a configuration problem at the firewall.

Our Contribution. We propose a method for dynamic firewall reconfiguration based on data (meta-attributes and user attributes) retrieved through a proxy-based FIM infrastructure. In particular, we used the Stork infrastructure created in the frame of the homonym European project. Stork uses national proxies (one per country) that act as a bridge for user authentication and also retrieve, filter, and map user attributes, which are *certified* by the national electronic identity authorities.

We generalise this firewall configuration problem to investigating how a proxy-based FIM infrastructure can support a wireless roaming access (WIRA) service and, in general, an attribute-based access control (ABAC) process. In the ABAC model [3,4], attributes are used for classifying every party, and access control policies specify the attributes that requesters need to have in order to gain access to data and services. Normally, authorization decisions are taken *after* the user has been authenticated and user attributes have been retrieved. However, when a proxy-based FIM is in place, some authorization decisions must be taken on the SP side *before* the authentication of the users has completed. We call this “authorization-for-authentication”, i.e. the problem of validating access to entities and operations of the FIM infrastructure itself. We solve this problem by allowing proxies to dynamically retrieve, validate and distribute a set of “meta-attributes”: data concerning the FIM entities and operations rather than the users.

Organisation. Section 2 briefly describes the Stork architecture, along with its main components involved in the authentication process and attribute transfer. In Section 3, we propose meta-attributes as a Stork-based solution to the “authorization-for-authentication” problem, while Section 4 presents the design and implementation of a Stork-based WIRA service integrated with Shibboleth [5]. Conclusions are in Section 6.

2 The Stork Infrastructure

Stork (Secure Identity Across Borders Linked) is a EU-funded project [6] that has defined a pan-european proxy-based FIM infrastructure integrating the various national e-identity approaches. Stork offers a set of identity-related services, like authentication request/response handling and user attribute retrieval and transfer. These services are implemented either by a national proxy server, named PEPS (Pan-European Proxy Service), or by a dedicated software, named *middleware* [7]. Each EU Member State (MS) can freely decide which approach to support. A country adopting the proxy approach runs a PEPS, which is part of a distributed EU infrastructure connecting the national identity infrastructures (Fig. 1). Each PEPS has two interfaces: the one used for communicating with SPs and IDPs is MS-specific, whereas the one communicating with the other proxies uses SAML 2.0 [8] and is specified in the project [7,9]. The SP contacts its national proxy (called S-PEPS) to ask for user authentication, while an IDP responds to user authentication requests originating from its own proxy (called C-PEPS).

To access a web-based service, the user is first asked to authenticate by the SP (step 1) that, when recognizes a Stork user, creates an authentication request

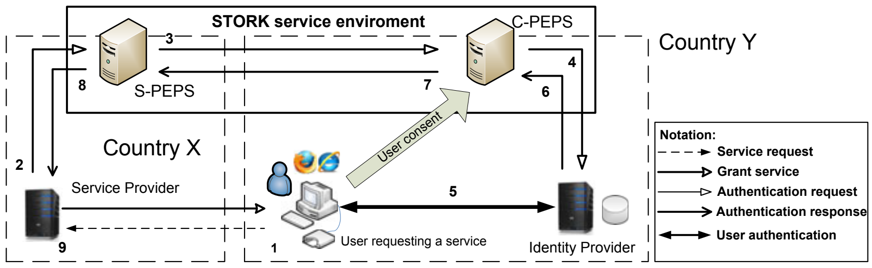

Fig. 1. The Stork PEPS-based approach

and sends it to the S-PEPS through the UA (step 2). Next, the S-PEPS asks the user where is he from, or more precisely in which country his authentication credentials were issued. Upon country selection, the S-PEPS constructs a signed SAML request, which is sent (through the user's browser) to the C-PEPS (step 3). This request is based on the one received from the SP and lists the required user attributes. The C-PEPS selects the appropriate national IDP for user authentication and creates an authentication request which is sent to the IDP (step 4). In step 5, the IDP performs the authentication exchange with the user and returns the appropriate response to the C-PEPS (step 6). After validating the response, the C-PEPS maps the received attributes to the Stork format, and (if necessary) derives additional attributes (e.g. "ageOver" derived from birthdate) for privacy protection purposes. In addition, the C-PEPS requires also the user's consent to forward his attributes to the S-PEPS. Finally, the C-PEPS creates a signed SAML response (containing the user attributes values) and sends it response to the S-PEPS, through the UA with the HTTP POST method (step 7). The S-PEPS performs similar operations as the C-PEPS, and sends subsequently the newly created SAML response to the SP (step 8), where the certified attributes are extracted and verified to eventually grant access to the requested service (step 9).

3 Meta-attributes Support in Stork

The proxy-based FIM in Fig. 1 assumes the following pre-defined trust relationships: all proxies trust each other, SPs and IDPs trust their national proxies, each user trusts its IDP and its national proxy. Establishing trust relationships consists in exchanging authentication credentials and associated metadata. Proxies, SPs and IDPs often exchange credentials in the form of X.509 certificates, while IDPs may decide to use different types of credentials for their users. The prime goal of FIM operations is establishing dynamic trust relationships between users and SPs by controlled retrieval and disclosure of user's identity attributes.

As introduced in Section 1, we extend the basic operations of the proxy-based FIM infrastructure to exchange a set of *meta-attributes*, i.e. data concerning some FIM entity or operation rather than the user. Meta-attributes help in taking access control decisions at the SP, and in mitigating various security threats, like man-in-the-middle attacks or attacks against platform integrity. The purpose of meta-attributes is easily explained in relation to the SAML metadata [10]. In fact, they both describe SAML endpoints and their supported operations. In the basic FIM model, SP and IDP typically exchange their SAML metadata out-of-band, when establishing a mutual trust relationship. SAML metadata may be published also at well-known URLs or in the DNS, as this information is assumed to be rather static. In proxy-based FIM, an alternative way to distribute information on FIM entities and operations is through the proxy infrastructure itself. Meta-attributes allow taking full advantage of proxies' scalability and their pre-defined trust relationships to disseminate and authenticate even highly dynamic metadata.

Based on the needs of the case "authorization-for-authentication", we have extended the Stork's PEPS component to resolve and transfer several meta-attributes: "Address" transfers PEPS and IDP location information to be used in IP-level access control decisions, "Credential" distributes FIM entities' credentials (like certificates), and "IntegrityMeasures" carry data useful to validate the PEPS operational integrity.

The Address Meta-attribute. In a WIRA service, the SP needs to know the IP address and TCP port of the IDP to configure dynamic network access control rules that permit the UA to connect to its IDP, but it may also obtain the C-PEPS location as well (in our experiments in Section 4, the C-PEPS location is known to the SP). In this scenario, the SP initially authorises the UA to connect only to the S-PEPS. The S-PEPS asserts the appropriate C-PEPS address as Address meta-attribute in its first response to the SP, so that the SP can grant the UA access to its C-PEPS as described in Section 2. In turn, the C-PEPS asserts the IDP address in its response to the S-PEPS, that forwards it in its second response to the SP. The SP now grants access to the IDP to allow user authentication and the Stork workflow to conclude. At this point, the SP may decide to grant "unrestricted" network access to the UA.

The Credential Meta-attributes. In Fig. 1, the UA sets up TLS channels with the other FIM entities it is contacting. Assuming that the user directly trusts the C-PEPS but not the S-PEPS, the UA needs to acquire the S-PEPS's certificate in a secure manner (e.g. out-of-band), otherwise either the user should decide directly to trust the S-PEPS certificate or the S-PEPS should have a certificate issued by one of the trusted CAs embedded in the browser. Similar considerations hold for the SP's certificate. All the options above are open to threats against the UA's trusted certificate store, including compromising the store itself, compromising anyone of the trusted (child) CAs, or misleading the user to add untrustworthy certificates to the store.

The Credential meta-attributes allow asserting information about PEPS and SP certificates. The SP and S-PEPS insert their own credentials inside their SAML requests, while the C-PEPS asserts in its response the certificates of the S-PEPS (known by a pre-defined trust relationship) and the SP (received in the request from the S-PEPS). If the UA is a dedicated SAML-enabled component, it can extract these certificates and verify “a posteriori” that the servers to which it connected were authentic (or abort the session otherwise). Implementing the same capability with web browsers is trickier since browsers do not expose TLS sessions and stored certificates through the APIs accessible to client-side web applications. In this case, the C-PEPS, upon extraction and validation, shows the SP and S-PEPS certificates to the user for manual inspection and confirmation. However, the user must compare the certificates’ fingerprints with those in the connections with the SP and S-PEPS. The usability of this solution is clearly unsatisfactory and its improvement requires an extension in the browser. For example, VeriKey [11] performs certificate verification with an automated process by means of dedicated verification servers and an extension or a plug-in for common web browsers (on the user side), while no modification is required on the web server to which the user connects to.

Credential meta-attributes are also useful at the SP. In the WIRA service, in addition to the IP address, the SP could retrieve the C-PEPS and IDP certificates and compare them against the ones presented by the servers connecting with the UA. The SP could even establish VPNs with the C-PEPS and IDP to enforce network access control beyond the IP-level.

Credential meta-attributes can also identify UA-owned credentials to help ensuring that the same UA participates in each step of the FIM workflow. In fact, SAML assertions are known to be vulnerable to attacks in browser-based implementations [12]: since SAML assertions are not strictly bound to the browser that originally acquired them from the IDP, attackers could impersonate the user by stealing valid SAML assertions from the browser. SAML 2.0 defines the ”Holder-of-Key” (HoK) element [13] as a confirmation method to prevent attackers using stolen SAML assertions. It allows specifying one or more credentials an entity should use to prove its entitlement to spend a given SAML assertion. These confirmation credentials generally differ from the long-term ones shared between the user and the IDP: in [14], they are certificates used by browser-based UAs to connect to IDPs and SPs via TLS with client-authentication. The Credential meta-attributes extend the HoK capability in the proxy-based FIM: the PEPS requires TLS client-authentication and compares the certificate presented by the UA with the Credential meta-attribute in the SAML assertions received from another PEPS or IDP; the PEPS reasserts UA’s confirmation credential in its SAML assertions. To support this capability, browser-based UAs should be able to generate volatile certificates to use with all the servers in the same FIM workflow session. Using a static self-signed certificate [15] requires no modification to common browsers, but the user must generate a new certificate for each session to achieve pseudonimity.

The IntegrityMeasures Meta-attributes. These attributes allow asserting data useful to validate the integrity of the PEPS. They are a natural extension of the other meta-attributes as they provide further information for authenticating the PEPS as trustworthy. For example, if TCG techniques [16] are used at PEPS, this information can include reference integrity measures of the PEPS platform.

The UA may then compare this information with fresh integrity measures on the S-PEPS, collected via TCG's remote attestation services. Similarly, the SP may check integrity measures when connecting to the C-PEPS.

Embedding Meta-attributes in SAML Assertions. A straightforward way to embed meta-attributes in SAML requests and assertions is defining new SAML attributes. This is the solution we adopted in our experiments for the Address meta-attribute. An alternative is overloading standard features of SAML 2.0 to provide equivalent semantics: for example, the Subject field allows identifying entities by URIs, which could include IP and port, while the HoK element could transport entity credentials. In this case, the PEPS would not add additional attributes to its assertions about the user, but it would return additional assertions on the entities involved in the FIM workflow (IDP, other PEPS, SP). The various assertions can be linked together just by asserting the role of the subject of each assertion. This approach would improve the flexibility of the solution based on specific SAML attributes.

4 WIRA Service Based on Stork Architecture

Well-known technologies, like Radius, IEEE 802.1X, or VPN, enable *known users* to dock to wireless and wired networks [17]. However, these techniques do not scale easily to the case of *visiting users*. For example, the Eduroam project [18] set up a network of Radius servers using EAP-TTLS to allow visiting users to authenticate with their home credentials. However, changing the authentication method requires upgrading the client application, and Radius/EAP do not offer the rich handling of certified user attributes supported by FIM technology. The ABFAB IETF WG is also working at the application of federated authentication for non-web services, essentially using EAP and Radius in place of SAML. A Shibboleth-based proposal for wireless roaming access in FIM scenario is described in [19], but it assumes that the IDP's location (DNS name and IP address) is known and pre-configured at the SP. We propose a proxy-based FIM solution to gain service scalability and flexibility via the retrieval of IDP attributes and the mapping of user attributes.

Based on Stork, we designed a flexible and scalable SAML-based service for wireless roaming access. The user is connected to a VPN and each connecting user is assigned a private IP address. The network access in the SP domain is protected by a captive portal, hosting a SAML-enabled module, the Access Control Decision Point (ACDP) and the Access Control Enforcement Point (ACEP) modules, as shown in Fig. 2.

The SP machine hosts also the gateway used for connecting the roaming user to the Internet. The packet filtering engine (resident at the ACEP) is driven by the ACDP module, which in turn is triggered by the SAML-enabled SP module, which constructs the SAML authentication requests, processes the responses received from the S-PEPS, and exports the attributes to the ACDP module. The ACDP retrieves the necessary meta-attributes and user attributes and triggers the enforcing firewall scripts of the ACEP module.

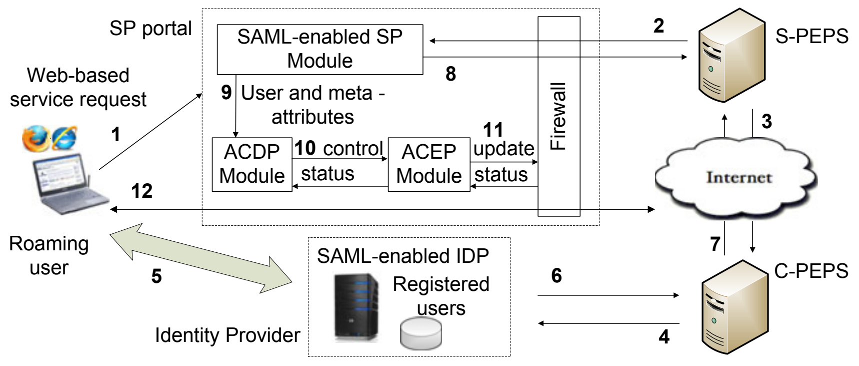

Fig. 2. WIRA service architecture exploiting Stork and Shibboleth 2

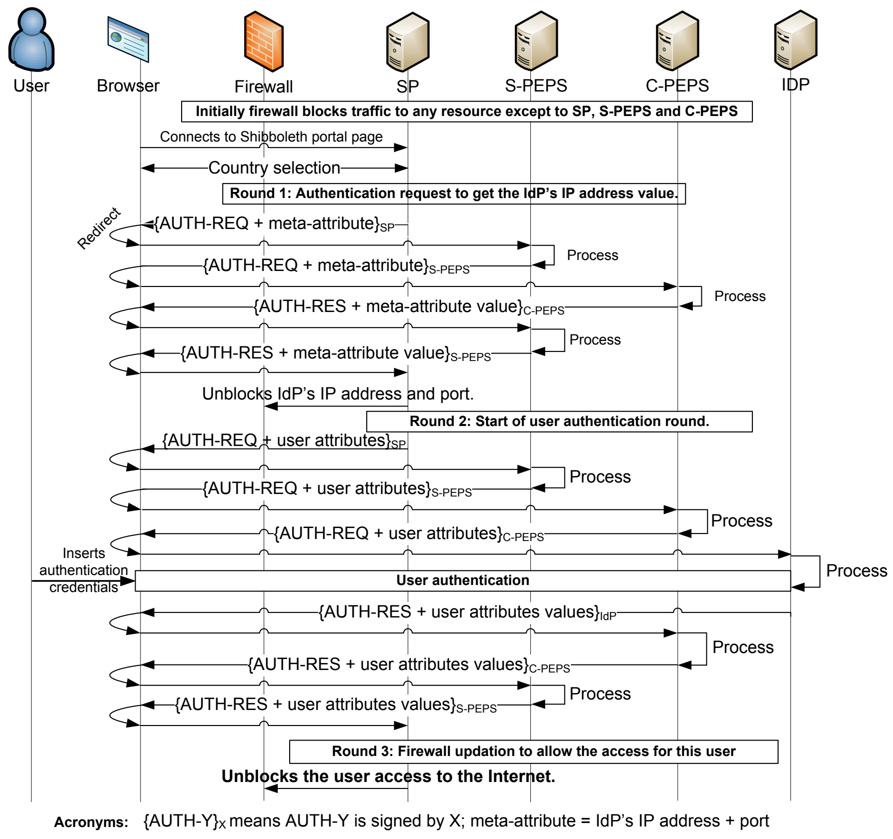
User
Browser
Firewall
SP
S-PEPS
C-PEPS
IDP
Initially firewall blocks traffic to any resource except to SP, S-PEPS and C-PEPS
Connects to Shibboleth portal page
Country selection
Round 1: Authentication request to get the IdP's IP address value.
Redirect
{AUTH-REQ + meta-attribute}SP
Process
{AUTH-REQ + meta-attribute}S-PEPS
Process
{AUTH-RES + meta-attribute value}C-PEPS
Process
{AUTH-RES + meta-attribute value}S-PEPS
Unblocks IdP's IP address and port.
Round 2: Start of user authentication round.
{AUTH-REQ + user attributes}SP
Process
{AUTH-REQ + user attributes}S-PEPS
Process
{AUTH-REQ + user attributes}C-PEPS
Process
Inserts authentication credentials
User authentication
{AUTH-RES + user attributes values}IdP
Process
{AUTH-RES + user attributes values}C-PEPS
Process
{AUTH-RES + user attributes values}S-PEPS
Round 3: Firewall updation to allow the access for this user
Unblocks the user access to the Internet.
Acronyms: {AUTH-Y}X means AUTH-Y is signed by X; meta-attribute = IdP's IP address + port

Fig. 3. WIRA Service functionality and messages exchanged

IDP Location Issue. In our service, the IDP location is not known in advance to the SP, but it is maintained at the C-PEPS. Thus, the IDP can freely move its location without notifying all potential SPs. Since the roaming user must connect to his IDP, the SP must selectively grant user access towards the IDP even though the user has not been authenticated yet. The SP enforces this access decision based on the SAML assertion contained in the response from the S-PEPS, which contains the IDP's location: basically, if the response received is valid then the IDP's location is transmitted to the ACDP module, which triggers the firewall scripts (residing at the ACEP module) to permit user network access towards the IDP (step 5 in Fig. 2) and later towards the Internet (step 12).

The WIRA service is composed of three rounds (Fig. 3). In the first one the IDP's Address meta-attribute is resolved at the C-PEPS, transmitted to the S-PEPS, and then forwarded to the SP as part of the authentication response. In the second round, the SP firewall allows the user to authenticate with his home IDP. On successful user authentication, the IDP creates a response with the requested user attributes values and sends it back to the SP, passing through the PEPS proxies. In the third round, the SP takes its authorization decision by allowing user partial or complete network access based on the user attributes received and its internal authorization policy.

5 Experimental Setup for the WIRA Service

The experimental setup consists of a wireless access point, the SP, IDP, PEPS and user's machines as shown in Fig. 4. The SP machine has two network interfaces, one with a public IP address and one with a private IP address in the same subnet as the roaming users. Iptables is used at the SP to permit or deny the IP traffic from roaming users towards the external world. The PEPS and the IDP hosts have public IP addresses. The SP machine runs Ubuntu (v9.04), Shibboleth SP (v2.0), PHP (v5.2.6) and Iptables (v1.4.1.1). The PEPS machine runs the PEPS package developed in the Stork project, which incorporates a SAML engine based on OpenSAML [20], whereas the SP and IDP run the Shibboleth software (v2.0). We chose Shibboleth because it incorporates a SAML 2.0 engine and allows to manage easily IDP and SP functionality through dedicated configuration files. The PEPS functionality is implemented by four Java servlets, two handling the SAML requests from the SPs and other PEPS respectively, and two handling the SAML responses from the IDPs and other PEPS. To create the Address meta-attribute on the C-PEPS, we implemented the `IPResolver` servlet. The S-PEPS, C-PEPS and Shibboleth IDP servlets run on an Apache Tomcat (v6.0.26) application server. To support user authentication with X.509v3 certificates stored in the browser or on user's smart-cards, we extended the Shibboleth IDP as by default it supports only password-based authentication.

By accessing the SP captive portal, the user is redirected to a PHP script page displaying the countries supporting Stork (step 1 in Fig. 4). Upon country selection, the script redirects the UA to the `WIRAstart` URL, whose handler is configured in the Shibboleth SP's `shibboleth2.xml` file indicating to the

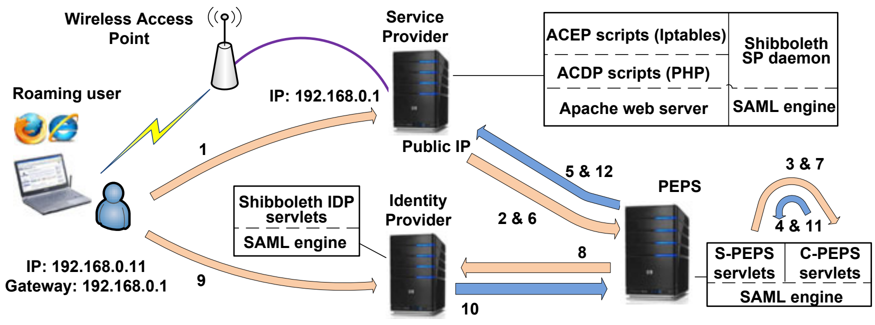

Fig. 4. Experimental setup configuration

Shibboleth SP daemon to generate a SAML request containing the `Address` meta-attribute. The request is sent through the user browser to the S-PEPS proxy servlet (step 2). To support the `Address` meta-attribute, we extended the SAML request with an additional Stork attribute (besides the ones described in [9]) and modified accordingly the Shibboleth SP's configuration and the original PEPS code. The S-PEPS validates the request, re-signs it, and forwards it to the `IPResolver` servlet on C-PEPS, which resolves the IDP's address and TCP port (step 3). Next, a SAML response containing the `Address` meta-attributed is constructed and is sent to the S-PEPS servlet (step 4), which verifies it, re-signs it, and sends it back to the `AssertionConsumerServiceURL` indicated in the `shibboleth2.xml` file (step 5).

The Shibboleth SP daemon validates the response and exports the `Address` value in the environment variable `idpaddr`. This mapping is configured in the Shibboleth SP file `attribute-map.xml`. Next, the UA is redirected to the SP portal's page `index.php`, configured as `homeId` in the `shibboleth2.xml` file. This page extracts the `idpaddr` value and calls a script containing Iptables rules to allow traffic from the UA (whose IP address is read from Apache server's environment variables) to `idpaddr`.

Subsequently, the UA is redirected to another URL configured in the Shibboleth SP's configuration file, so that the daemon generates another SAML request containing the needed user attributes. This request is sent to the S-PEPS proxy servlet (step 6) and then to C-PEPS proxy servlet (step 7), and finally to the Shibboleth IDP (step 8). On the Shibboleth IDP, the user can authenticate either with username and password, the X.509 certificate stored in the browser or a RSA-based smart card. Upon authentication, the Shibboleth IDP generates a SAML authentication response containing the user attribute values extracted from the database of registered users. In case the user authenticates with a smart-card, the national identification number in the certificate on the smart card is used as search key in the registered users database. The authentication response is sent back to the corresponding C-PEPS servlet, which will also filter the attributes that have not been requested and maps the requested ones to the Stork format. After being re-signed, the SAML response is sent to the S-PEPS servlet, which performs similar operations as the C-PEPS and sends finally the response back to the SP handler URL, i.e. the `AssertionConsumerServiceURL`.

If the response `StatusCode` is success, the SP daemon exports the requested attribute values in the environment variables. This ends the second round of user authentication. In the third round, the user is again redirected to the portal page `index.php`, the environment variables are read, and the script containing the Iptables rules for this round is called, granting access to the network based on the user attributes received.

6 Conclusions and Future Work

We have shown that dynamic firewall reconfiguration is needed for guest users to be authenticated via a proxy-based FIM infrastructure (as in the case of wireless roaming access) and proposed a practical solution based on addtional SAML meta-attributes and the Stork infrastructure. By generalising this approach, we could also distribute additional data for dynamic certificate validation and server integrity measurement. Future work will focus on extending our WIRA service to implement additional controls based on this dynamic data, and on best embedding this data in standard SAML syntax and operations.

Acknowledgements. This work was developed in the framework of the EU co-funded project STORK (INFSO-ICT-PSP-224993, `http://www.eid-stork.eu`). We also thank Hafeez HafeezurRehman for his help in implementing some features of the WIRA service.

References

1. Maler, E., Reed, D.: The Venn of Identity: Options and Issues in Federated Identity Management. IEEE Security & Privacy, 16–23 (March/April 2008)
2. Makoto, H.: Federation proxy for cross domain identity federation. In: Proc. of ACM DIM 2009, pp. 53–62 (2009)
3. Bonatti, P., Samarati, P.: Regulating service access and information release on the web. In: ACM CCS 2000, pp. 130–145 (November 2000)
4. Yuan, E., Tong, J.: Attribute Based Access Control (ABAC) for Web Services. In: Proc. of ICWS 2005, pp. 561–569 (July 2005)
5. Cantor, S. (ed.): Shibboleth architecture - Protocols and Profiles (September 2005), `http://shibboleth.internet2.edu`
6. Secure Identity Across Borders Linked (STORK) project - Towards pan-European recognition of electronic IDs (eIDs) (2008-2011), `http://www.eid-stork.eu`
7. Berbecaru, D., Jorquera, E., Alcalde-Moraño, J., Portela, R., Bauer, W., Zwattendorfer, B., Eichholz, J., Schneider, T.: Software architecture design. STORK Deliverable D5.8.2a (October 2010), `https://www.eid-stork.eu/`
8. OASIS: Assertions and Protocols for the OASIS Security Assertion Markup Language (SAML) V2.0. OASIS Standard (March 2005)
9. Alcalde-Moraño, J., Hernández-Ardieta, J.L., Johnston, A., Martinez, D., Zwattendorfer, B., Stern, M., Heppe, J.: Interface specification. STORK Deliverable D5.8.2b (October 2010), `https://www.eid-stork.eu/`

10. OASIS: Metadata for the OASIS Security Assertion Markup Language (SAML) V2.0. OASIS Standard (March 2005)
11. Stone-Gross, B., Sigal, D., Cohn, R., Morse, J., Almeroth, K., Kruegel, C.: VeriKey: A dynamic certificate verification system for public key exchanges. In: Zamboni, D. (ed.) DIMVA 2008. LNCS, vol. 5137, pp. 44–63. Springer, Heidelberg (2008)
12. OASIS: Security and Privacy Considerations for the OASIS Security Assertion Markup Language (SAML) V2.0. OASIS Standard (March 2005)
13. OASIS: Profiles for the OASIS Security Assertion Markup Language (SAML) V2.0. OASIS Standard (March 2005)
14. OASIS: SAML V2.0 Holder-of-Key Web Browser SSO Profile Version 1.0. OASIS Committee Specification (August 2010)
15. Gajek, S., Liao, L., Schwenk, J.: Stronger TLS bindings for SAML assertions and SAML artifacts. In: Proc. of ACM SWS 2008, pp. 11–19 (October 2008)
16. Trusted Computing Group, `https://www.trustedcomputinggroup.org`
17. Manulis, M., Leroy, D., Koeune, F., Bonaventure, O., Quisquater, J.-J.: Authenticated Wireless Roaming via Tunnels: Making Mobile Guests Feel at Home. In: Proc. of ASIACCS 2009, pp. 92–103 (2009)
18. Eduroam: `http://www.eduroam.org`
19. Linden, M., Viitanen, V.: Roaming Network Access Using Shibboleth. In: TERENA Networking Conference 2004, pp. 1–1 (2004)
20. OpenSAML libraries, `https://spaces.internet2.edu/display/OpenSAML/Home`