dCache, towards Federated Identities & Anonymized Delegation

dCache, towards Federated Identities & Anonymized Delegation

A Ashish[1], AP Millar[1], T Mkrtchyan[1], P Fuhrmann[1], G Behrmann[2], M Sahakyan[1], O S Adeyemi[1], J Starek[1], D Litvintsev[3], A Rossi[3]

[1] IT Dept., DESY, Notkestrasse 85, Hamburg, Germany
[2] NORDUnet, Copenhagen, Denmark
[3] Fermi National Accelerator Laboratory, PO Box 500, 60510, Batavia, IL, USA

E-mail: `Anupam.Ashish@desy.de`

Abstract. For over a decade, dCache has relied on the authentication and authorization infrastructure (AAI) offered by VOMS, Kerberos, Xrootd etc. Although the established infrastructure has worked well and provided sufficient security, the implementation of procedures and the underlying software is often seen as a burden, especially by smaller communities trying to adopt existing HEP software stacks [1]. Moreover, scientists are increasingly dependent on service portals for data access [2]. In this paper, we describe how federated identity management systems can facilitate the transition from traditional AAI infrastructure to novel solutions like OpenID Connect. We investigate the advantages offered by OpenID Connect in regards to 'delegation of authentication' and 'credential delegation for offline access'. Additionally, we demonstrate how macaroons can provide a more fine-granular authorization mechanism that supports anonymized delegation.

1. Introduction

dCache [3] is a storage system that is actively used in the High Energy Physics (HEP) community[4–8]. The HEP community has adopted and developed various Identity Management Systems to store, manage and verify digital identities, and restrict access to a system based on these determined identities [9–11]. dCache relies on the end user to provide the necessary credentials which are then used to validate and establish user's identity, i.e. *authenticate*, before initiating an operation. Subsequently, this identity information is used to authorize the user to perform the respective operation. In most cases, dCache performs the authorization phase internally. There are exceptions like *ALICE* - LHC experiment group, which relies on proprietary authorization scheme using cryptographically signed authorization tokens on top of Xrootd data access protocol [12].

Whereas in Federated Identity, the user is provided with a *transferable digital authenticated assertion*, which can be used to obtain access to a secured resource on all organizations belonging to the same federation [13, 14]. The process of approval of the access to the system is based on the attributes present in the authenticated assertion and the restrictions imposed are based on the consent given by the user during the process of obtaining this assertion.

dCache supports multiple identity and authentication mechanisms in order to authenticate users and authorize them for data access. VO Management System (VOMS) X.509 proxy certificates [9, 11] and Kerberos credentials [15] have been ubiquitously adopted by HEP sites as a proven Authentication and Authorization Infrastructure (AAI) stack [1] and are used extensively

for data access in dCache. A valid VOMS proxy certificate guarantees the membership to a trusted Virtual Organizations (VOs) on which subsequent authorization is based upon. In comparison, Kerberos access tokens require an interactive exchange between the client, service provider (e.g. dCache) and the KDC (Key Distribution Center) in order to obtain an authenticated assertion for subsequent data access. Scientists performing experiments at multiple HEP sites often rely on Kerberos and VOMS for the storage and access of experiment data.

The upcoming projects like European XFEL are showing an increasing dependence on unified service frameworks like Karabo to store, manage, access and process data [2]. These consolidated services provide portals for data access, processing and analysis to the scientists, which in turn depend on delegation of user credentials to themselves to perform these tasks independently [16]. Moreover, the service portals depend greatly on *controlled sharing* [17], with scientists visiting from partner universities and collaborating on multiple national and international projects.

The prevalent authentication and authorization mechanisms are insufficient for these modern use cases. While Kerberos access tokens cannot be used to inter-operate with similar services at other HEP sites [1], VOMS causes an externalization of data access control and is often seen as a burden for light-weight collaborations by smaller communities trying to adopt existing HEP software stacks [18].

dCache in increasingly deployed as a part of federated storage systems. It is mandatory as a member of the federation to accept requests from users registered with other partners from within the federation. Therefore, it is essential for dCache to be able to establish a trust relationship between itself and the institute that issued the user's credentials.

Bearer Tokens, first introduced by W3C in HTTP/1.0 [19], solved the problem of accessing an associated resource by demonstrating the possession of a secret (e.g. cryptographic key or username/password). They are authenticated assertions and have been implemented in multiple ways, e.g. in OAuth2 [20], OpenID Connect [21], SAML assertions [22] etc. These have been adopted by various services, including dCache, for authentication and authorization. Despite the ubiquitous adoption of bearer tokens, the greater problem of *identity delegation*, simplification of the process for obtaining a *federated identity* (single credential to obtain access to secure resources in multiple organizations) [13] and obtaining decentralized anonymous credentials still remain at large in HEP software stacks [13, 14].

In this paper, we introduce how OpenID Connect can be used to authenticate visiting scientists based on credentials from their home institute (a trusted identity provider - IdP) and demonstrate how identity delegation can be performed in order to achieve data storage and access with dCache managed storage systems. We also demonstrate how a single authorization token based on multiple distinct digital identities (e.g. OpenID Connect, SAML, username/password etc.) can be obtained from a trusted federated Identity Provider (IdP) like INDIGO-DataCloud AAI [23, 24]. At the end, we show how bearer tokens can be combined with more powerful *Macaroons* to obtain a decentralized, flexible, anonymized credential [17].

2. Delegated Authentication

Delegated Authentication is an authentication process in which user authentication and identity management is outsourced to a trusted Identity Provider (IdP)[14, 25]. The IdP provides the user with mechanisms to control attribute release and produces an authentication assertion (delegated token) for the user. This delegated token is an encoded token representing the consent information provided by the IdP and allows the user to perform an approved set of actions and tasks on the resource provider[26].

A classic example where delegation is very useful is when a scientist runs an analysis using a service portal (e.g. Karabo) on a large data set which is on a distributed storage system. Such an operation would require a lot of time, during which the scientist may or may not be online

and the data may be residing on multiple sites. In order for this processing to run without an intervention, the service portal can obtain a delegated token from the IdP on the scientist's behalf. This delegated token can then be used to access data from the storage system and refreshed periodically before it expires.

In a delegated authentication flow (Fig. 1), a scientist making a request against a service portal is redirected to their trusted IdP. The scientist authenticates with the IdP and provides consent to the portal for the reception of delegated token. The IdP, then, redirects back with the encoded delegated token to the service portal, which can be used to perform access to the storage system later.

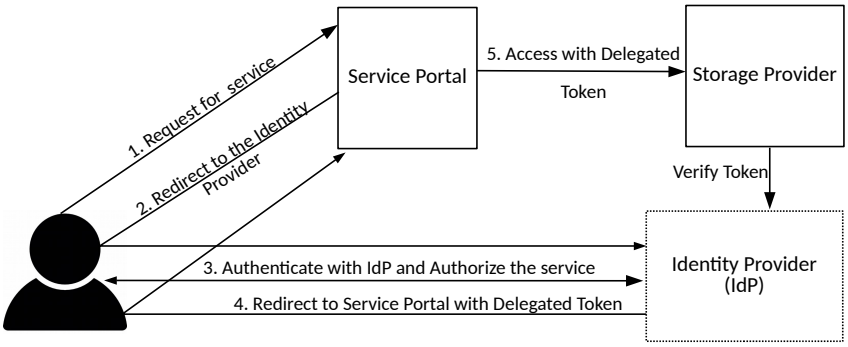

Figure 1: Delegated Authentication

Fig. 1 shows how the delegation of authentication to a trusted IdP works. It improves the security of the authorized services by decoupling the responsibility of authentication from these resource providers [27, 28]. The burden of creating, storing and maintaining user identities for resource providers like dCache is also reduced. Furthermore, the provider can be configured to trust and accept identities from multiple IdPs (e.g. Google, CERN Single-SignOn etc.), hence simplifying the integration with multiple identity management systems.

Popular web-service providers like Google [29], Facebook [30] and Twitter [31] have exposed standardized APIs to 3rd-parties for authentication delegation. Based on OpenID Connect, these HTTP-Rest like APIs [21, 29] have been widely adopted by other Internet services (e.g. digg.com, stackoverflow.com) and can be seen on their websites (Fig. 2).

Since version 2.16 [32], dCache is able to accept requests with delegated tokens and can be configured to work with any IdP (e.g. Google) that supports OpenID Connect. In the Section 4, we will discuss how a token can be delegated to dCache, allowing it to perform *offline access* on a user's behalf.

3. INDIGO-DataCloud AAI

In Section 1, we determined how the prevalence of multiple identity management systems is a problem for scientific communities and collaborating researchers. A *Federated Identity Management* entails an agreement between multiple institutions (*federation*), which allows a user of any member institute to use the same credential for authentication and obtaining access to the secured resources of all of them [13, 24].

The *INDIGO Authentication & Authorization Infrastructure* (INDIGO-AAI [24, 33]) is a standardized authentication and authorization framework to provide a unified federated identity management [34] system, based on OpenID Connect, for INDIGO Services. In addition to federated identity management, it features a standard infrastructure (Fig. 3) to support external authentication mechanisms like SAML and X.509 for INDIGO services not accepting OpenID Connect credentials.

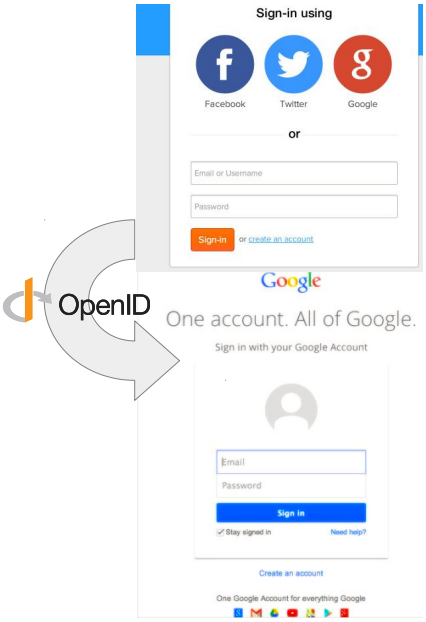

Figure 2: Example of Delegated Authentication with a mocked web-portal and Google

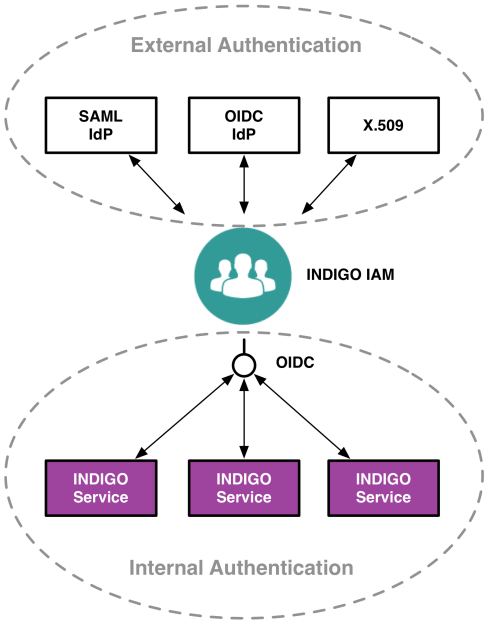

Figure 3: The INDIGO-AAI

The INDIGO-AAI provides the following respectively,

- *Identity Layer* that leverages OpenID Connect for user authentication and retrieval of identity information.
- *Identity Harmonization* layer to allow a user to authenticate with username/password, SAML, X.509 certificates or via an external OpenID Connect provider [24, 33].
- *Authorization* Layer to provide fine-grainular attribute-based authorization and group membership based on Argus Authorization Service [35] and OAuth2 [20] respectively.
- *Identity Provisioning* layer based System for Cross Domain Identity Management (SCIM) [36] to provision, de-provision and manage identities.
- *Delegation and Offline Access* based on OpenID Connect and OAuth2. It provides a partial implementation for token-exchange to perform a controlled delegation of offline access rights [37] to INDIGO services in order to execute tasks on behalf of a user.
- *Token Translation Service* to translate an OpenID Connect token to other types (e.g. SSH Keys, X.509 Credentials) in order to support INDIGO services that rely [38].

The ability of INDIGO-AAI to harmonize external authentication mechanisms, such as SAML and X.509, to OpenID Connect while linking the distinct identities into a single unique persistent identifier, is a key step towards achieving a federation of services. Conversely, the translation of OpenID Connect identities to other forms (e.g. X.509) permits existing services incompatible with OpenID Connect to join the federation.

dCache has supported Kerberos Credentials, VOMS X.509 proxy certificates since version 1.9 [3] and partially supports OpenID Connect as of version 2.16 [32]. With the help of Token Translation Service, it is now possible for dCache to join as an INDIGO service and leverage the INDIGO-AAI to establish a trustful relationship between itself and a member institute within the federation that issued the user's credential.

4. Token Exchange Protocol

The authorization token (e.g. SAML assertions, Bearer Token for OpenID Connect) facilitates the sharing of identity information in heterogeneous environments [37, 39]. These assertions have to be integrity protected and sometimes encrypted to achieve confidentiality. In a heterogeneous environment (e.g. Grid), they have to be exchanged with other services to execute tasks on the user's behalf, since providing a new token to each downstream service is cumbersome. Furthermore, the user should be able to move data between different storage entities seamlessly in a federated storage system.This flow of identities and authorization tokens is achieved using either delegation or impersonation [37].

In case of impersonation, an entity 'B' on receiving a token from 'A', comes into possession of all the identity information and rights in the context in which the token was granted. It is in every sense indistinguishable from 'A' and for all purposes 'B' is considered to be impersonating 'A' [37]. However, it is not desirable that 'A' has to forego all of the identity information and has little control over restricting the usage of the token.

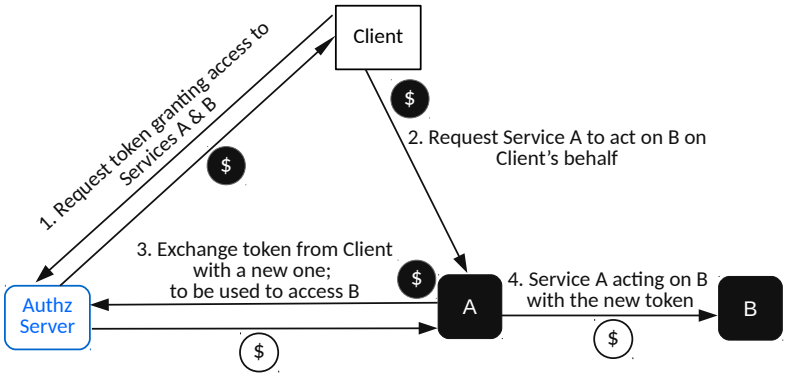

Figure 4: Token Exchange Protocol

In Fig. 4, it is shown that the Client requests for a token of grant type "token-exchange" from the authorization server. It makes a request to service 'A' to act on its behalf on 'B'. 'A' performs a token exchange to receive a new token which is narrower in scope and thereupon uses the new token to make requests to downstream services, e.g. 'B'. This process is also referred to as "delegation" with token-exchange because the two entities 'B' and 'A' maintain their separated identities and 'A' delegates some of its rights to 'B'. It is well understood by the downstream services that 'B' is only acting as an agent of 'A'. In the context of INDIGO-AAI, the authorization server in Fig. 4 can be the INDIGO-IAM and 'A' and 'B' can be INDIGO services. Since version 3.0 [3], dCache supports token-exchange to perform 3rd-party transfers. This allows it to transfer data into and out of itself, on behalf of the user.

5. Authorization Token use cases and limitations

dCache is strongly focused on exploiting the utility of authorization tokens. *Service Portals* are becoming popular as an interface to dCache and other similar backend storage systems. The token-exchange protocol, discussed in Section 4, allows a portal to act on a user's behalf and perform data management and analysis operations without user interaction (*offline access*). In Fig. 4, the entity 'A' can be thought of as a service portal interacting with a dCache storage system 'B'.

Similarly, *sharing* of data in storage systems is highly desirable. The INDIGO-IAM encodes group membership information in the authorization tokens, which enables dCache to provide group-based data access. Moreover, the authorization tokens can be shared between multiple clients/users in impersonation or delegation mode (Section 4), which further simplifies data

sharing between any two entities. Although they are extremely flexible in being generated, shared or encoded with consent and scope, it is important to investigate their limitations.

- The delegation of authentication in OpenID Connect requires the client to be redirected over HTTP from the service to the IdP. Consequently, the authorization token is obtained by the service with a redirection back from the IdP to the client. These redirections during the request and response phase are typically implemented with Javascript, i.e. in a browser. However, it is not straightforward to achieve the HTTP redirection outside of the browser without Javascript.
- The authorization token is relayed with all requests and often propagated with following requests to downstream services. These tokens have no internal protection and rely solely on underlying SSL/TLS for their security. Although, the IdP provides token revocation end-point, they still suffer from token redirect and token reuse threat scenarios [40].
- OpenID Connect delegation provides the user with the ability to restrict the scope of a token within a certain context (e.g. the bearer of this token can fetch user identity information). However, these scopes can not be more fine granular or new restrictions can not be added to existing tokens.

6. Macaroons: Cookies with Contextual Caveats

In light of the limitations mentioned in Section 5, Birgisson et. al. introduced Macaroons, which they define as cookies with contextual caveats [17]. *Macaroons* are bearer tokens (authorization assertion) that enable an application to determine whether the request is authorized [17]. They are safer for *controlled sharing* of credential in decentralized, distributed systems. Although similar to bearer tokens, they add several features which are not available with other token-based authorization schemes,

- *Attenuation*: Macaroons enable users to add "caveats", which are limitations on how the macaroon can be used. A macaroon can carry one or more caveats and each caveat carries a restriction, e.g. who can use it, where can this macaroon be used, what can be done with the macaroon. The holder of an authorization token has full privilege over the token and can perform all tasks on a user's behalf. In contrast, the holder of a macaroon can add caveats to it, hence "attenuate" the macaroon.
- *Delegation*: Whereas bearer tokens also support delegation, a macaroon enables its holder to add a caveat restricting its usage at the intended recipient only. Thus, a macaroon can be attenuated before delegation which makes it safer to pass on to downstream services.
- *Cryptographically Verifiable*: All macaroons carry their own proofs which are cryptographically secure. These proofs are carried in caveats that are constructed using chained HMAC functions. These chained HMAC functions make it easy to add a caveat but cryptographically impossible to remove one. They also simplify the verification of the macaroon by its creator, but make it impossible for others.
- *Third-party Caveats*: Macaroons allow for third-party caveats which are predicates that require a third-party assertion (e.g. group membership). This third-party must certify that the caveat is satisfied in order for the authorization to be successful.

Caveats can be added to a macaroon, before handing it over to the service portal, as shown in Fig. 5. The service portal obtains a macaroon with additional restrictions allowing only write operations from a certain source storage service (from IP address). Furthermore, an additional caveat restricting the time validity of the token can be added. Even if the macaroon is intercepted by a rogue service, any actions originating from it will be discarded since the caveat on IP address cannot be satisfied. This is greatly beneficial for federated services, so that all requests and actions can be tightly controlled.

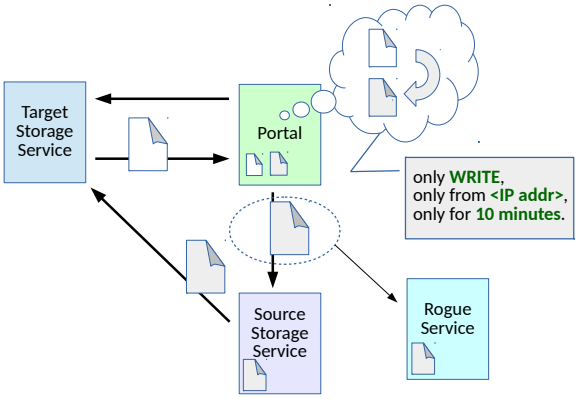

Figure 5: Third-party copy with Macaroons

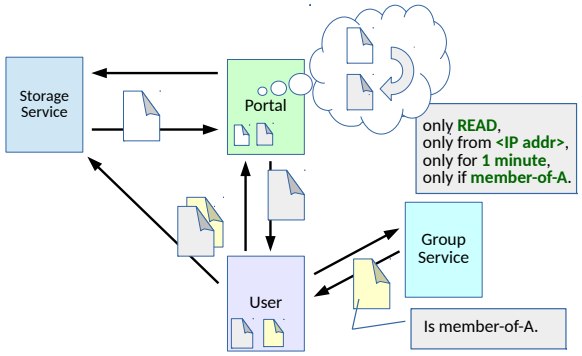

Figure 6: Macaroons with 3rd-party caveat

In dCache, group memberships are quintessential for accessing HEP experimental data and have been implemented with the help of VOs (e.g. VO ATLAS). A user is mapped through VO-RoleMaps to a group which owns the data stored. Similar to the above, such restrictions can be added to macaroons with the help of third-party caveats. In Fig. 6, a caveat for limiting access to user from a certain group is added and can be verified only at a trusted IdP (providing Group Service).

Macaroons can also be used for *anonymized delegation*. For example in Fig. 6, the user can request the portal for an anonymized macaroon. The portal requests the storage service for a macaroon with a third-party caveat on the group membership and hands it over of the user. Subsequently, the user can make a request directly to the storage service with the macaroon. Storage service contacts the Group Service upon receiving the request, which then releases a discharge macaroon verifying the group membership of the user.

With these advantages offered by macaroons, dCache intends to support caveats with IP addresses, paths, actions (e.g. list, download, upload, delete) and time expiration. The dCache RESTful API will be extended to acquire macaroons and add caveats. The support for macaroons is anticipated to be released with version 3.2 and will be initially provided for HTTP/WebDAV end-points.

7. Conclusion

In this paper, we have shown how federated identity management systems coupled with identity harmonization can consolidate a multitude of authentication and authorization mechanisms, such as VOMS and Kerberos, and worked with INDIGO-DataCloud to facilitate the transition of existing infrastructures to novel solutions like OpenID Connect. To this end, we have demonstrated the advantages of OpenID Connect for delegation of authentication and authorization token delegation for offline access. We have extended dCache to support OpenID Connect and perform delegation. Furthermore, we laid out the benefits of adding fine-granular attributes for authorization in macaroons and anonymized delegation.

Acknowledgments

Work described in this paper was funded by INDIGO-DataCloud, DESY, Fermilab and NDGF.

References

[1] Crawford M 2005 URL `https://cds.cern.ch/record/865725`
[2] Heisen B, Boukhelef D, Esenov S, Hauf S, Kozlova I, Maia L, Parenti A, Szuba J, Weger K, Wrona K and Youngman C 2013 (14th International Conference on Accelerator & Large Experimental Physics Control Systems, San Francisco (USA), 6 Oct 2013 - 11 Oct 2013) p FRCOAAB02 ISBN 978-3-95450-139-7 URL `https://bib-pubdb1.desy.de/record/168167`
[3] The dCache project website `http://www.dcache.org/` accessed: 2017-02-08

[4] Haupt A, Leffhalm K, Wegner P and Wiesand S 2011 *Journal of Physics: Conference Series* Doi:10.1088/1742-6596/331/1/012007
[5] Oleynik G *et al. 22nd IEEE / 13th NASA Goddard Conference on Mass Storage Systems and Technologies* pp 73–80 doi:10.1109/MSST.2005.16
[6] Deatrich D *et al.* 2008 *22nd International Symposium on High Performance Computing Systems and Applications* pp 167–171 doi:10.1109/HPCS.2008.27
[7] Behrmann G, Fuhrmann P, Grønager M and Kleist J 2008 *Journal of Physics: Conference Series* vol 119 (IOP Publishing) p 062014 doi:10.1088/1742-6596/119/6/062014
[8] Fuhrmann P and Gülzow V 2006 *Euro-Par 2006 Parallel Processing* (Springer) pp 1106–1113 doi:10.1007/11823285_116
[9] Groeper R, Grimm C, Piger S and Wiebelitz J 2007 *Software Engineering and Advanced Applications, 2007. 33rd EUROMICRO Conference on* (IEEE) pp 367–374
[10] DFN-AAI website `https://www.aai.dfn.de/` accessed: 2017-02-08
[11] Simmel D, Rea S and Stolk A 2012 *Proceedings of 2012 Latin American Conference on High Performance Computing (CLCAR'12)*
[12] Feichtinger D and Peters A J 2005 *6th IEEE/ACM International Conference on Grid Computing (GRID 2005)* pp 172–178 URL `http://dx.doi.org/10.1109/GRID.2005.1542739`
[13] Daan Broeder *et al.* 2013 Federated Identity Management for research collaborations
[14] Birrell E and Schneider F B 2013 *IEEE Security Privacy* **11** 36–48 ISSN 1540-7993
[15] Neuman B C and Ts'o T 1994 *Communications Magazine, IEEE* **32** 33–38
[16] Bencivenni M, Michelotto D, Alfieri R, Brunetti R, Ceccanti A, Cesini D, Costantini A, Fattibene E, Gaido L, Misurelli G *et al.* 2015 *Journal of Grid Computing* **13** 159–175
[17] Birgisson A, Politz J G, lfar Erlingsson, Taly A, Vrable M and Lentczner M 2014 *Network and Distributed System Security Symposium*
[18] Gaignard A and Montagnat J 2009 *HealthGrid 2009* (*Studies in health technology and informatics* vol 147) (IOS Press) pp 257–262 URL `https://hal.archives-ouvertes.fr/hal-00677795`
[19] Berners-Lee T, Fielding R T and Nielsen H F 1996 RFC 1945 – Hypertext Transfer Protocol – HTTP/1.0 `http://www.faqs.org/rfcs/rfc1945.html`
[20] Jones M and Hardt D 2012 The OAuth 2.0 Authorization Framework: Bearer Token Usage RFC 6750 URL `https://rfc-editor.org/rfc/rfc6750.txt`
[21] Sakimura N, Bradley J, Jones M, de Medeiros B and Mortimore C 2014 *The OpenID Foundation, specification*
[22] Organization for the Advancement of Structured Information Standards 2005 Security assertion markup language (saml) v2.0
[23] Ertl B, Stevanovic U, Hayrapetyan A, Wegh B and Hardt M 2016 *High Performance Computing & Simulation (HPCS), 2016 International Conference on* (IEEE) pp 621–627
[24] Salomoni D, Campos I, Gaido L, Donvito G, Antonacci M, Fuhrman P, Marco J, Lopez-Garcia A, Orviz P, Blanquer I *et al.* 2016 *arXiv preprint arXiv:1603.09536*
[25] Morgan R, Cantor S, Carmody S, Hoehn W and Klingenstein K 2004 *Educause Quarterly* **27** 12–17
[26] Architectural Overview of Delegated Authentication `https://msdn.microsoft.com/en-us/library/cc287682.aspx`
[27] Siriwardena P 2014 *Advanced API Security: Securing APIs with OAuth 2.0, OpenID Connect, JWS, and JWE* (Apress)
[28] Li W and Mitchell C J 2016 *Detection of Intrusions and Malware, and Vulnerability Assessment* (Springer) pp 357–376
[29] Google Sign In and OpenID Connect `https://developers.google.com/identity/protocols/OpenIDConnect`
[30] Facebook Login for the Web with the JavaScript SDK `https://developers.facebook.com/docs/facebook-login/web`
[31] Sign in with Twitter `https://dev.twitter.com/web/sign-in`
[32] dCache v2.16.x `https://www.dcache.org/downloads/1.9/index.shtml#server-2.16`
[33] Ceccanti A *et al.* 2016 *22nd International Conference on Computing in High Energy and Nuclear Physics, CHEP 2016* URL `https://indico.cern.ch/event/505613/contributions/2227724`
[34] INDIGO Identity and Access Management `https://iam-test.indigo-datacloud.eu`
[35] Argus Authorization Service `http://argus-documentation.readthedocs.io/en/latest/`
[36] System for Cross Domain Identity Management (SCIM) `http://www.simplecloud.info/`
[37] Jones M, Nadalin A, Campbell B, Bradley J and Mortimore C 2017 OAuth 2.0 Token Exchange Tech. rep. URL `https://tools.ietf.org/html/draft-ietf-oauth-token-exchange-07`
[38] The INDIGO Token Translation Service `https://tts.data.kit.edu`
[39] Jones M, Goland Y, Mortimore C and Campbell B 2015 Assertion Framework for OAuth 2.0 Client Authentication and Authorization Grants RFC 7521 URL `https://rfc-editor.org/rfc/rfc7521.txt`
[40] Tschofenig H and Hunt P 2012 OAuth 2.0 Security: Going Beyond Bearer Tokens Tech. Rep. draft-tschofenig-oauth-security-01 URL `https://tools.ietf.org/html/draft-tschofenig-oauth-security-01`