PROCEEDINGS OF SCIENCE

DIRAC: OIDC/OAuth2 based security framework

A.Tsaregorodtsev[a,*] , **A.Lytovchenko**[a]

a Aix-Marseille Univ, CNRS/IN2P3, CPPM
163 Avenue de Luminy, Marseille, France

E-mail: atsareg@ccpm.in2p3.fr, alytovchenko@cppm.in2p3.fr

The DIRAC Interware is the framework for building distributed computing systems which allows to integrate various kinds of computing and storage resources in a transparent way from the user's perspective. Up until recently, the client communications with DIRAC were based on a custom protocol using X.509 PKI certificates. Following the recent move towards OIDC/OAuth2 based security infrastructure, the DIRAC client/server protocol was enhanced to support both proxy certificates and tokens. The new framework has components for user authentication and authorization with respect to the DIRAC services. It also has a Token Manager service for maintaining long-living tokens necessary to support asynchronous operations on the user's behalf. The tokens now can be used to access computing resources such as HTCondorCE and ARC Computing Elements as well as cloud sites. Enabling access to the storage resources and other third-party services is currently under intensive development.
In this paper we describe the architecture of the DIRAC security framework together with various aspects of its implementation. The choice of the solutions is largely motivated by the requirement of continuity of the DIRAC services already in production and transparency of changes for the end users. The usage of OAuth2 tokens in dedicated or multi-community DIRAC services as well as the necessity to support multiple Identity Provider services is discussed. We also provide an outlook of future development plans with the goal to achieve a complete, scalable and user-friendly security framework for the DIRAC Interware project.

International Symposium on Grids & Clouds (ISGC) 2023 in conjunction with HEPiX Spring 2023 Workshop, ISGC&HEPiX2023
19 - 31 March 2023
Academia Sinica Taipei, Taiwan

*Speaker

PoS(ISGC&HEPiX2023)029

1. Introduction

A general tendency of the modern science is an ever-increasing need of research communities in computing and storage resources. The race for getting access to more and more computing power started with the launch of the LHC Collider at CERN, Switzerland, where the 4 LHC experiments produced unprecedented data volumes. The solution for the processing and analysis of the very large amounts of data was found in the concept of computational grid which brought together hundreds of traditional computing centers around the world into a single coherent infrastructure. Later, other types of computing resources were progressively made available to the LHC experiments and other scientific communities, such as High Performance Computing (HPC) centers, private and public clouds as well as standalone computing clusters available to the researchers. The need to operate scientific workflows consistently using distributed heterogeneous resources resulted in the development of dedicated software tools allowing to present the variety of computing sites as a single coherent system to the users.
The DIRAC Interware project [1] is one of the software frameworks which allows to build and operate distributed computing systems of arbitrary scale. It started as a solution for one of the LHC experiments (LHCb) but later it was adopted also by other High Energy Physics and Astrophysics Collaborations such as Belle II, ILC/CLIC, Juno, CTA and others [2]. Several distributed computing infrastructures are offering DIRAC services to their scientific communities, for example, EGI, GridPP/UK, IHEP/China and JINR/Russia.
One of the main elements of any distributed computing infrastructure is a commonly adopted security system defining and imposing common rules of access to shared resources according to policies agreed by all the partners. The grid security solution was chosen to be based on the X.509 PKI infrastructure where each user should obtain a certificate from a corresponding Certification Authority (CA). The management of user communities is performed by means of the community VOMS services to register user membership, their groups and roles which determine the user rights to access common resources. This system is now being replaced by another one which is based on the actual *de facto* industry standard technologies OAuth2 and OIDC. The migration from X.509 to OIDC/OAuth2 based solutions have multiple advantages for the users and providers of resources and services but it is also a challenge for developers of distributed computing frameworks.
The DIRAC Interware project developed its own support for the OIDC/OAuth2 based security. This development has several specific requirements stemming from the fact that DIRAC services support multiple user communities each possibly having different Identity Providers services. In this paper we will describe the DIRAC security model and its implementation based on X.509 and OAuth2 technologies in Section 2. The new security framework changes the way user management is performed which is discussed in Section 3. Accessing third party resources (Computing Elements and clouds) is presented in Section 4 followed by discussion of the development status and further plans outlook in Section 5. Finally, the Conclusion summarizes the new DIRAC security framework realization.

2. DIRAC security model

2.1 Security based on X.509 certificates

The current DIRAC security model based on the X.509 certificates has the following elements and steps to be performed by the users. Before starting to use DIRAC services, users should be registered in the Registry (Fig.1). Each user record in the Registry contains the username and the user's certificate Distinguished Name (DN). One or more certificates can be associated with the username. The users are also registered in one or several DIRAC groups. Each group is belonging to a Virtual Organization (VO) and has Properties which are defining rights of the group members with respect to DIRAC services.

The Registry is populated with user records either manually by a VO administrator or by a special software Agent (VOMS2CS Agent) which is synchronizing the contents of the Registry with the user data obtained from the VOMS services of the VOs served by DIRAC. The user information in the VOMS service, the user's groups and roles, are mapped onto the DIRAC groups. As a result of the synchronization, users can be added, deleted or suspended in the DIRAC Registry, they can be also added to or removed from one or more DIRAC groups.

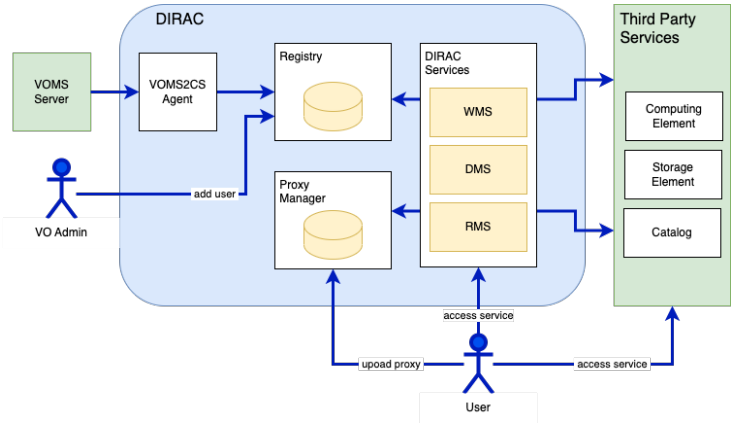

Fig.1 DIRAC Security Model

In order to access DIRAC services, users should create a proxy of their certificates. This is done by a special command where the user specifies the desired group to be used for the subsequent interactions with the DIRAC services. As a result, the obtained certificate proxy is generated with an extension specifying the user group after checking the user group membership. This group is taken into account to evaluate user's rights when interacting with the DIRAC services over a custom secure access protocol (DIPS).

Many users related operations in DIRAC are performed asynchronously without the user's presence, for example, executing user's jobs from the DIRAC Task Queue or storing user's data. These operations are performed with credentials of the users which are the owners of the corresponding jobs and data. In order to enable DIRAC services to perform user's operations, users are uploading long-living certificate proxies to the Proxy Manager service which stores them securely. Later on, the Proxy Manager service provides short-living user proxies for other DIRAC services, Workload, Data and Request Management Systems (WMS, DMS, RMS), which thus can perform operations on the user's behalf.

2.2 Transition to OIDC/OAuth2 based model

Upgrading the DIRAC security framework from using X.509 certificates to the OAuth2 tokens is a challenging development task for several reasons. First of all, DIRAC services in production should ensure a smooth transition for the users without disruption of their activities and their main usage patterns. It is practically inevitable that in the transition period DIRAC should ensure user authentication and access rights control based on *both* certificates and tokens. Only after all the user communities and all third-party services are enhanced with the token support, the possibility to use certificates can be dropped from the DIRAC software stack.

The DIRAC services are often supporting multiple different VOs simultaneously. This results not only in the fact that VOs adopt tokens at a different pace but also can use different Identity Provider services, such as EGI Check-In service [3] and WLCG IAM services [4]. These services use different user profiles which complicates mapping them onto the DIRAC security framework entities (DIRAC users and groups).

The following DIRAC components should be updated for supporting tokens.

- The DIRAC client/service protocol should support tokens for the user's authentication and control of access to DIRAC service;

- The User Management should be based on the information obtained from Identity Provider service rather than from VOMS services;
- The Token Management should be introduced in order to provide valid tokens for asynchronous operations on the user's behalf;
- Connectors to third-party services should be developed which support both certificates and tokens based access.

In the following sections we will describe in more details the upgrade and implementation of each of the components.

2.3 User authentication with tokens

Before users can start accessing DIRAC services, they have to pass authentication or login into the system. In other words, they have to obtain credentials necessary to access the services. In DIRAC, both Command Line (CLI) and Web user interfaces are available in which the login steps are slightly different.

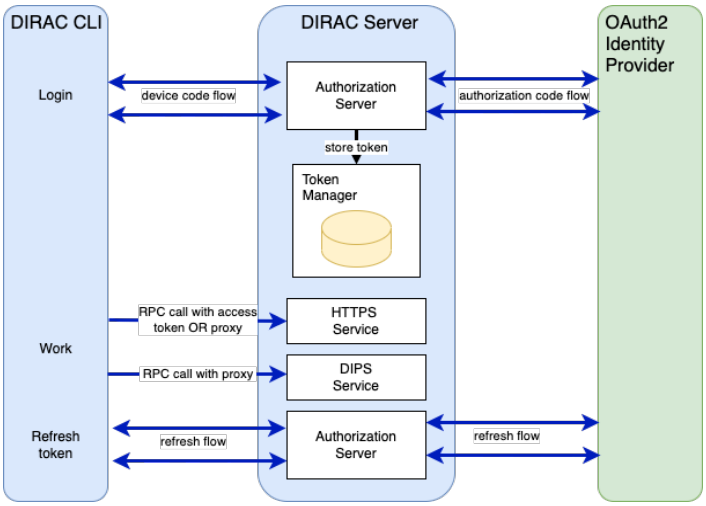

Fig.2 User authentication with tokens

In the case of CLI, the key component in the authorization process is the DIRAC Authorization Server (AS) with which the user interacts using a special *dirac-login* command (Fig.2). The AS is implementing a standard OAuth2 Authorization Server interface. However, DIRAC does not play the role of Identity Provider itself. Instead, the AS receives the user's requests and guides the user for subsequent authentication and authorization steps. At some steps the AS communicates with a selected Identity Provider service using appropriate authentication flows in order to obtain valid tokens suitable for the user's working session. In particular, the following steps happen while the user's login process:

1. With the *dirac-login* command, the user initiates *device code flow* with the DIRAC AS;
2. The AS proposes a choice of several Identity Provider services to the user;
3. The AS initiates *authorization code flow* with the chosen identity provider;
4. After a successful authentication and authorization (AA) by the Identity Provider, the AS receives user's access and refresh tokens and stores them into the Token Manager database (see more details below);
5. The AS proposes a choice of DIRAC groups of which the user is a member;
6. After the group is selected, the AS is obtaining a new user's access token from the Identity Provider with the scope limited to the one corresponding to the chosen group using *refresh token flow*;
7. As a final result, the AS returns the scoped access token to the user.

With the obtained token, the user can now access DIRAC services which are aware of the user's access rights. The access token can be later refreshed to ensure continuous user working session. In the case of the DIRAC Web Portal interface, the steps are slightly different. The user is presented a choice of configured Identity Providers and the AS initiates *authorization code flow* with the chosen service. After the successful AA step, the user's refresh token is stored in the Token Manager database and the browser receives a secure cookie for the current working session. The session data will allow the Web Portal to interact with other DIRAC services on behalf of the user with the proper user identity and the chosen DIRAC group.
The described AA architecture with an intermediate AS is justified for several reasons:

- It serves as a single entry point for all the DIRAC users from different VOs possibly using different Identity Provider services. In this case, there is no need for the DIRAC clients to have complicated local configuration, the users are guided by the AS for all the necessary choices based on the centrally managed configuration;
- The AS stores long refresh tokens in the Token Manager database transparently for users. Otherwise, this would require a separate user action;
- AS assists users in choosing the right DIRAC group eligible for the user's configured group membership and the chosen Identity Provider;
- The user can request and obtain from the AS together with the token also a proxy certificate since AS can interact with the DIRAC Proxy Manager service. This is an essential feature for the transition period.

In the whole, the developed AA procedure allows users to stay completely within the familiar DIRAC ecosystem avoiding confusion with multiple AA services, methods and credentials.

2.4 Token Manager service

The Token Manager service stores the long user's refresh tokens in its database in order to provide later on necessary credentials for asynchronous user operations (Fig.3). This functionality is similar to the Proxy Manager service which stores long user certificate proxies that can be used to generate appropriate user credentials to access third-party computing or storage services.

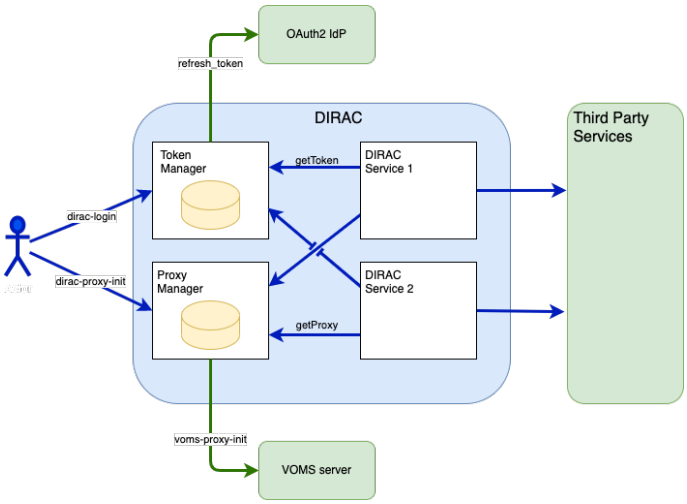

Fig.3 User credentials Management

The Token Manager service provides an interface that can be used by other DIRAC trusted services to obtain tokens suitable for a particular user operation. To do that, the Token Manager employs a *refresh token flow* with the appropriate Identity Provider for which the DIRAC service is registered as a client having *offline access* scope. The user access tokens can be requested with a limited scope necessary for only a specific operation.

The Token Manager service can also obtain tokens with a *client credentials flow*. In this case the tokens obtained are not belonging to a particular user but rather represent the DIRAC service with respect to third-party services. This can be quite useful for the DIRAC Pilot Factory, the central element of the Workload Management System based on the concept of pilot jobs. Provided that the DIRAC service is trusted by the sites offering computing resources, the use of this type of credentials simplifies significantly the credential management procedures because no stored refresh tokens are needed in this case.

2.5 DIRAC client/server communications

The DIRAC client/server protocol is now undergoing a significant modification. The custom protocol DIPS is replaced by the HTTPS based protocol in order to have a modern standards-based solution which is supported by multiple mature software libraries. Therefore, it was decided that the token support will be only enabled for the HTTPS version of the protocol which will also support the X.509 certificates.
The implementation of the DIRAC services API is providing a transparent use of both protocols where the appropriate choice is done automatically based on the configured service endpoint URL. This allows to migrate services from DIPS to HTTPS implementation progressively keeping the overall continuity of the system operations, which is especially important for multiple DIRAC extensions.

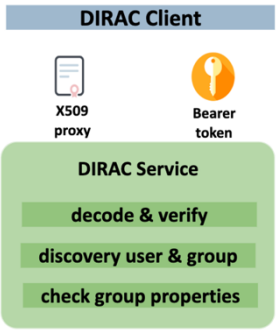

Fig.4 DIRAC Client/Server protocol

The implementation of the AA part of both protocols is illustrated in Fig.4. For each client request, the client credentials are first decoded and verified. Then, the user identity and group are evaluated based on the DIRAC Registry data. The necessary input is taken from the presented client credentials:

- In the case of X.509 proxy certificates, the username is derived from the certificate DN and the DIRAC group extension. The latter is checked for validity according to the current state of the user record in the Registry;
- In the case of OAuth2 token, the username is associated with the “sub” claim of the token and the DIRAC group is determined based on other claims which are Identity Provider dependent.

The DIRAC group evaluation in the case of authentication with tokens relies on the information present in the tokens themselves. The possibility of token introspection mechanism is considered to be unacceptable as it requires contacting the Identity Provider service upon each client request and implementation of a complex caching mechanism on the DIRAC server side. This will certainly present scalability problems. Therefore, both Check-In and IAM type Identity Provider services allow for creating tokens with group information embedded in the token as a special scope. It is important to mention that only groups requested at the token creation will be included and not all the groups eligible for the user. This is made possible with a *parametric scope* mechanism which is used in the *refresh token flow* as described above. An example of the token scope to DIRAC group mapping in the case of Check-In Identity Provider is the following. The token scope:
eduperson_entitlement:urn:mace:egi.eu:group:registry:biomed:role=member#aai.egi.eu
corresponds to the *biomed_user* DIRAC group. Similarly, the token scope:
wlcg.groups:/wlcg/pilot would correspond to the *wlcg_pilot* DIRAC group in the case of IAM type Identity Provider.
It is important to ensure unambiguous mapping of the scopes to DIRAC groups which should be properly taken into account in the VO configuration both on the Identity Provider and the DIRAC side.

Once the username and the DIRAC group are evaluated, the already existing DIRAC mechanism for determining user rights to access specific service interfaces *AuthManager* is applied in both certificate and token cases. As a result, the user is either granted or denied access to the DIRAC service.

3. User management

Currently, the user management in DIRAC is built around the Registry where the usernames, credentials and group membership are recorded. The DIRAC groups are defining the user rights with respect to the DIRAC services but they are also used to denote different activities within a given VO. This allows to define policies for different activities, for example, assign higher or lower priorities to certain groups. The groups are also reflected in the Accounting of consumed resources. Technically, the Registry is implemented as a section of the DIRAC Configuration Service (CS).
On the other hand, the DIRAC user communities use also external services to describe their users and rights. Therefore, the static DIRAC Registry should be kept up-to-date with the user data in the VOMS or Identity Provider services which is a rather challenging task. In order to cope with an increasing number of the DIRAC user communities, the design of the Registry is updated with the following goals in mind (Fig.5):

- The primary user management is completely delegated to the corresponding Identity Provider services. No preliminary user registration will be required, DIRAC will fully trust the information presented in the user credentials issued by the approved Identity Providers. Extra information can be obtained from Identity Providers through the corresponding UserInfo endpoints;
- The Registry will still keep definitions of the DIRAC groups and VOs in order to define user rights with respect to DIRAC services, as well as to describe VO policies and accounting of resources consumed under the DIRAC control;
- Instead of a static Registry a new DIRAC Registry service is introduced. It will maintain a dynamic cache of the user information coming from various community management services.
- The user information in the cache is normalized to keep it in a unique format independently of the source. New user's information is automatically added to the cache upon the first access to DIRAC services with valid credentials;
- The DIRAC services use the same interfaces for accessing static and dynamic Registries to ensure continuity in the migration to the new design.

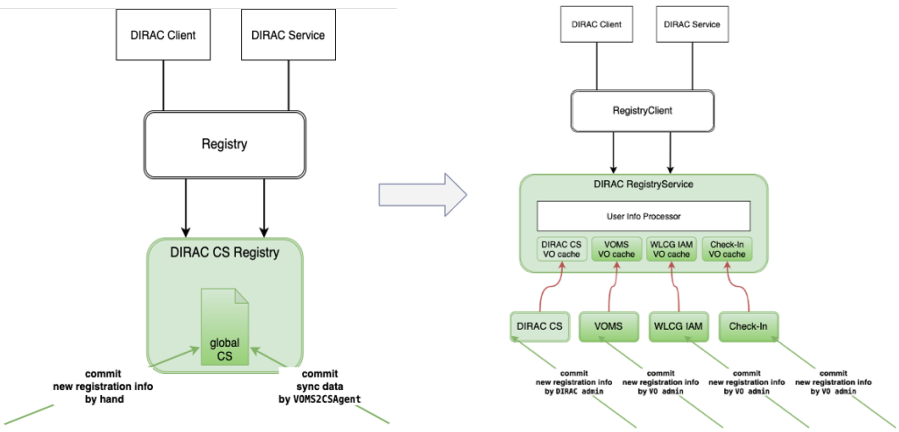

Fig.5 Migration from static to dynamic DIRAC Registry

The development of the dynamic DIRAC Registry is the work in progress. It will allow to significantly simplify the work of VO managers as they will have to maintain a single description of their communities in dedicated services with appropriate specially designed interfaces.

4. Accessing computing resources

DIRAC is accessing sites - providers of computing and storage resources - in order to execute user jobs and manage user data. The DIRAC Workload Management System (WMS) is based on the pilot job architecture. In this architecture, the computing worker nodes are, first, reserved by the pilot jobs which are inspecting the local environment, install the necessary software and then request user payloads from the central Task Queue. The user jobs are then executed under the control of the watchdog process. The pilots are obtaining credentials of the payload owners for the operations needing access to third-party services, e.g., to data storage services.
The DIRAC Pilot Factory, the component which submits pilot jobs to computing sites, is instrumented with the possibility to use tokens for those sites that are already upgraded for this user authentication method. At the same time, the use of X.509 certificates is still possible as well since during the transition period the mixture of sites supporting X.509 certificates and tokens is inevitable.
The main computing resources for DIRAC services are grid sites with HTCondorCE [5] or ARC Computing Element (CE) [6] services and cloud sites with the Openstack management system [7]. Now all these services employ token-based user authentication.

4.1 Grid Computing Elements

In the case of HTCondorCE and ARC CEs the corresponding DIRAC connectors are upgraded to use tokens for authentication when submitting pilot jobs. Each CE is configured to accept either certificates or tokens. The Pilot factory is using this information to obtain proxy certificates or tokens from the Proxy Manager or Token Manager service correspondingly as described in the previous section. In the case of tokens, two options are available:

- Use tokens of a user responsible for pilot job submission in the given community. Those tokens are obtained from the Identity Provider service through the *refresh token flow* mechanism starting from the refresh token stored in the Token Manager database;
- Use tokens belonging to the DIRAC service itself. In this case, the tokens are generated in the *client credentials flow* based on the credentials that the DIRAC service owns as a client of the Identity Provider service.

Tokens obtained by the client credentials mechanism are conceptually similar to the use of so-called “robot” X.509 certificates where one certificate is representing a whole user community. Generating such tokens is simpler and less error prone. So, if it is accepted by Identity Providers and user communities, this is the preferrable solution. DIRAC allows to register and use multiple clients of the Identity Provider service, for example, to represent different communities or different types of activities if necessary. The preferred way is to reuse the same client credentials for pilots of all the user communities. It is important to mention that tokens obtained with the client credentials mechanism are still generated with the scopes limiting them for the use by only a specific user community. This allows the computing sites use this information in order to map the jobs to proper local accounts according to their locally defined policies.

4.2 Cloud sites

Multiple cloud sites accessed under the DIRAC control require token-based authentication, for example, EGI Federated Cloud sites [7]. These are mostly sites using OpenStack cloud management services [8]. The corresponding DIRAC connector is used to instantiate dynamically virtual machines (VM) and run pilot jobs in them thus making these resources similar to any other grid site. However, this connector was not updated to use token authentication while interacting with the sites via their REST service interfaces. Instead, it applies another connection method called "Application Credentials" provided by the OpenStack API. The Application Credentials are generated by a user upon a successful authentication with the cloud site using tokens. These credentials are limited to a given project and user. Their properties are registered in the secure configuration of the DIRAC cloud connectors which now can instantiate VMs without a complex procedure of obtaining tokens on demand from the Token Manager service. Therefore, this method provides a simple and secure replacement for the tokens-based access to cloud sites.

4.3 Other computing resources

Other types of computing resources (HPC sites, standalone clusters) are usually not requiring access with authentication by tokens. In most cases, the access is done through and SSH tunnel using a specially registered DIRAC user account. However, if in the future some of these sites will require token-based authentication, this will be added to the corresponding DIRAC resources connectors.

4.4 Pilot Framework

Multiple distributed pilot jobs interacting with the central DIRAC services altogether compose the Pilot Framework. In this framework, secure interactions are performed not only during the pilot job submission but also throughout the whole life-cycle of the pilot jobs. Pilot jobs are communicating with the central DIRAC services to request user payloads, report their status and log their actions. In particular, the pilot jobs are able to obtain credentials of the payload owners to be used for all the payload related operations.
The infrastructures providing DIRAC services, e.g., EGI or WLCG, are not imposing any specific requirements on the implementation of secure connections performed within the Pilot Framework like the one of DIRAC. The choice of technology for its realization is left to the DIRAC developers. The current implementation is based on X.509 certificates. This is a well-established technology supported by the current and future versions of DIRAC as described in previous sections. Therefore, it is not planned to modify it to use tokens or any other method for secure connections. However, the user payload operations requiring access to third-party services, e.g., data storage services, will be performed with the credentials required in each particular case – either certificates or tokens.

5. Status and outlook

The OIDC/OAuth2 security framework in DIRAC is largely a work in progress. However, the main components are already available in the production software releases.
Several user communities using DIRAC in the EGI infrastructure started to configure their VO descriptions in a way compatible with their descriptions in the DIRAC Registry. The community users are registered with their Check-In identifiers. The user registration is done manually, for the moment, by administrators who receive mail notifications when a user makes an attempt to login into the DIRAC Web Portal. This process will be automated as soon as the DIRAC user

communities will have proper description of their users including user groups and roles. The already registered users can obtain their credentials in the CLI or Web interfaces with the DIRAC guidance using their Identity Provider service. In particular, in the CLI interface users can obtain proxy certificates while authenticating with their OAuth2 provider and without the need to install their certificates. This turned out to be very useful, for example, for the users of Jupyter notebooks where the DIRAC client software is preinstalled. The DIRAC Authorization Server is fully functional to support the user authentication process.
The Token Manager service is deployed and is used to serve tokens for the DIRAC Pilot Factory. Several HTCondorCE and ARC sites allowing access with tokens are configured in the EGI Workload Manager DIRAC service to run user jobs in production.
However, there are many components still in development or planned to be developed. Some of them were already mentioned in previous sections:

- The dynamic Registry service described above will allow to fully delegate management of DIRAC users to their respective Identity Provider and Community Management services;
- The user access tokens in the CLI interface are supposed to be short-living and therefore need a transparent refresh mechanism in order not to disrupt user working sessions. Several approaches to this problem are discussed by developers and an appropriate solution will be implemented;
- There are many communities that cannot provide X.509 certificates for their users but have access to grid sites that do not support tokens yet. For these communities DIRAC developers are requested to develop a connection to the RCAuth service [8] which can generate valid proxy certificates on the fly after a successful authentication with an Identity Provider service. For DIRAC users this mechanism will be made transparent and hidden behind the standard DIRAC authentication interface;
- It is planned to develop connectors to the SE services with tokens that carry scope information about user rights to access certain paths in the storage namespace. In the DIRAC Data Management System, the user ACLs are stored in the File Catalog service. Therefore, for user data operations it will be necessary to generate special tokens with storage access scopes evaluated from the File Catalog ACLs.

In the whole, there will be a need to follow the evolution of the OIDC/OAuth2 Identity Provider services. DIRAC developers are working in close contact with the Check-In and IAM developers and administrators in order to provide sufficient support for all the use cases of the DIRAC user communities.

6. Conclusion

The DIRAC Interware Project was started as a grid solution for the LHCb experiment at LHC and is now adopted by several High Energy Physics and AstroPhysics collaborations as well as by several multi-VO grid infrastructures. The Project is now undergoing a serious upgrade to introduce a new security framework based on the OIDC/OAuth2 industry standard technology. The new framework integrates Identity Provider services, such as EGI Check-In and WLCG IAM, to provide all the necessary tools for describing scientific communities, their structures, policies and user rights. Tokens provisioned by Identity Provider services can now be used for secure client/server communications with distributed DIRAC services using an HTTPS based protocol. Tokens are carrying scope information which determines user rights with respect to the DIRAC services. The new Token Manager service is developed to maintain and provide tokens to other DIRAC components for performing asynchronous operations on behalf of the users.
The DIRAC Workload Management System is instrumented with connectors to HTCondorCE, ARC and cloud computing services to enable Pilot Factories to submit pilot jobs with the token-based authentication. Several HTCondorCE sites are used with tokens in production in the EGI Workload Manager service. A smooth transition to the new security framework is ensured by

keeping compatibility with the X.509 certificates which can be used interchangeably with the tokens.
The migration to the new security framework is a continuous work in progress with many components still in development. This includes dynamic DIRAC Registry service which will allow to delegate user management completely to the Identity Provider services; access with tokens to third party services, such as SE services, RCAuth and some others. Much effort is devoted to following the evolution of the Identity Provider services, assisting user communities in describing their users and policies, providing convenient interfaces hiding the complexity of the underlying system for the users.

7. Acknowledgments

This work is supported by the EGI-ACE project which receives funding from the European Union's Horizon 2020 research and Innovation program under grant agreement no. 101017567.

References

[1] A Tsaregorodtsev and the DIRAC Project, DIRAC Distributed Computing Services, 2014 *J. Phys.: Conf. Ser.* **513** 032096, **DOI** 10.1088/1742-6596/513/3/032096, http://diracgrid.org

[2] F Stagni *et al,* DIRAC in Large Particle Physics Experiments*,* 2017 *J. Phys.: Conf. Ser.* **898** 092020 **DOI** 10.1088/1742-6596/898/9/092020

[3] EGI Check-In service, https://www.egi.eu/service/check-in/

[4] Indigo Identity and Access Management (IAM) service, https://github.com/indigo-iam/iam, WLCG Common JWT profile - https://github.com/WLCG-AuthZ-WG/common-jwt-profile

[5] HTCondor Compute Entrypoint, https://htcondor.org/htcondor-ce

[6] ARC, Advanced Resource Connector, https://www.nordugrid.org/arc

[7] EGI Federated Cloud, https://www.egi.eu/service/cloud-compute

[8] OpenStack cloud manager, https://www.openstack.org

[9] RCAuth service, https://rcauth.eu/