Using Kerberos Tokens in Distributed Computing System at IHEP

Xiaowei JIANG[1,2,*], *Chaoqi* GUO[1,2,**], *Qingbao* HU[1,***], *Ran* DU[1,****], *Jingyan* SHI[1,†], and *Gongxing* SUN[1,‡]

[1]Institute of High Energy Physics, Chinese Academy of Sciences
[2]University of Chinese Academy of Sciences

Abstract. The token-based certification method is spreading in the distributed computing system of high energy physics. More and more software and middleware are supporting tokens as one of the certification methods. As an example, WLCG has upgraded all the services to support WLCG tokens [1]. In IHEP (Institute of High Energy Physics in China), the Kerberos [2] token has been used as the main certification method in the local cluster. Naturally, it is selected as the certification method in the distributed computing system. In this case, a set of toolkits were developed or introduced to use Kerberos tokens in the distributed computing system, including token producer, token repository, token transfer and token client engine. The token producer is responsible for creating a token and publishing the token file to the token repository. The token repository stores all the latest token files and a refresh service periodically renews the lifetime of those tokens stored in the token repository. The token transfer brings the token file to the worker node. The token client engine initializes the token environment and renews the token's lifetime on the worker node. With these toolkits, the jobs can run in any worker node in any site and use the Kerberos token to access other services, such as EOS [3] and the XRootd [4] proxy service. In IHEP, the Kerberos toolkit has been deployed in the distributed computing system. Currently, three experiments (LHAASO [5], BES [6] and HERD [7]) are using Kerberos tokens to remotely access the data in EOS or Lustre [8].

1 Background

At IHEP, we are building a distributed computing platform to integrate several Chinese sites. However, several experiments with a long history at IHEP are tightly bounded to the local cluster environment. For example, the BES experiment, one of the biggest experiments in IHEP, bounds to the local resources, including DB, storage service, computing service, etc. So we are using a way called 'Cluster Expansion' to expand the local cluster to the distributed

*e-mail: jiangxw@ihep.ac.cn
**e-mail: guocq@ihep.ac.cn
***e-mail: huqb@ihep.ac.cn
****e-mail: duran@ihep.ac.cn
†e-mail: shijy@ihep.ac.cn
‡e-mail: sungx@ihep.ac.cn

computing cluster that makes BES jobs migrate to remote sites nearly transparently, as shown in figure 1.

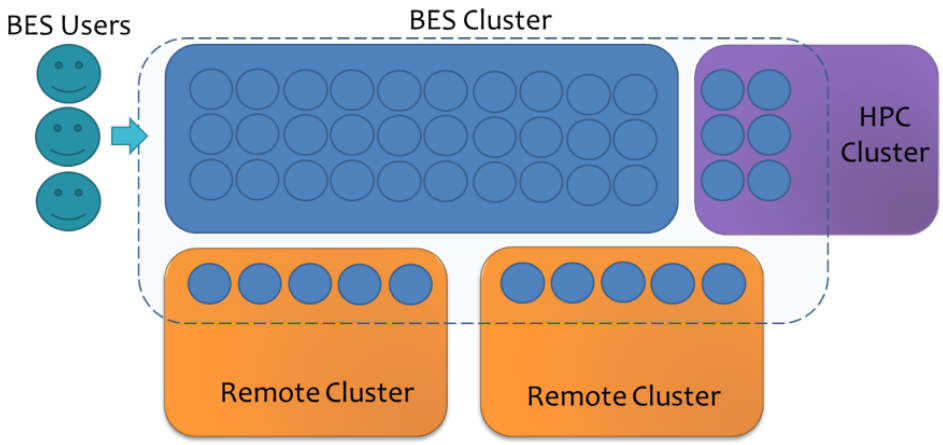

Figure 1. Computing Cluster Expansion (*Example of BES experiment*)

Considering crossing the different sites, a secure certification method is needed. Because the Kerberos token has been used as the main certification method in local computing cluster at IHEP, it is natural to expand to the distributed computing system.

2 Problems

A Kerberos token with a specific time validity is generated when a user logs into the submitter node. Before the validity time expires, the token can be used to access other services, for example the AFS [9] and EOS storage service. For the certification solution based on Kerberos tokens at IHEP, the biggest problem is how to extend the validity time. We need to guarantee the Kerberos token is renewed in the time points or periods:

1. when a job is submitted
2. when the job is idle in job queue
3. when the job is running.

3 Kerberos tokens in Distributed Computing System

As mentioned before, a Kerberos token is generated when a user logs into the submitter node. At IHEP, by default, 2 days of validity time and 7 days of renew time limit will be given when the Kerberos token is created. The token is valid before the validity time expires and the validity time can be renewed before reaching the renew time limit. In terms of the computing system, the Kerberos token is always used by jobs. On the submitter node, the user submits a job and the token is attached as one of the job input files. The scheduling system will send the job and the token to a worker node. The worker node can be a local worker node or a remote worker node. This paper is focusing on the remote site and more specifically on the operation mechanism on the remote worker node. The Kerberos token is stored in the job executing directory and initialized into the job environment when the job arrived at the remote worker node. With the Kerberos token, the job can access storage

services, including Lustre, XCache [10] and EOS in the IHEP case (the figure below shows the use of Kerberos token to access Lustre and XCache). As the certification of Kerberos tokens cover the local computing cluster and the remote sites, it can be potentially expanded to other kinds of services in the future, not only the storage service. Figure 2 shows the basic workflow of Kerberos tokens in the distributed computing system.

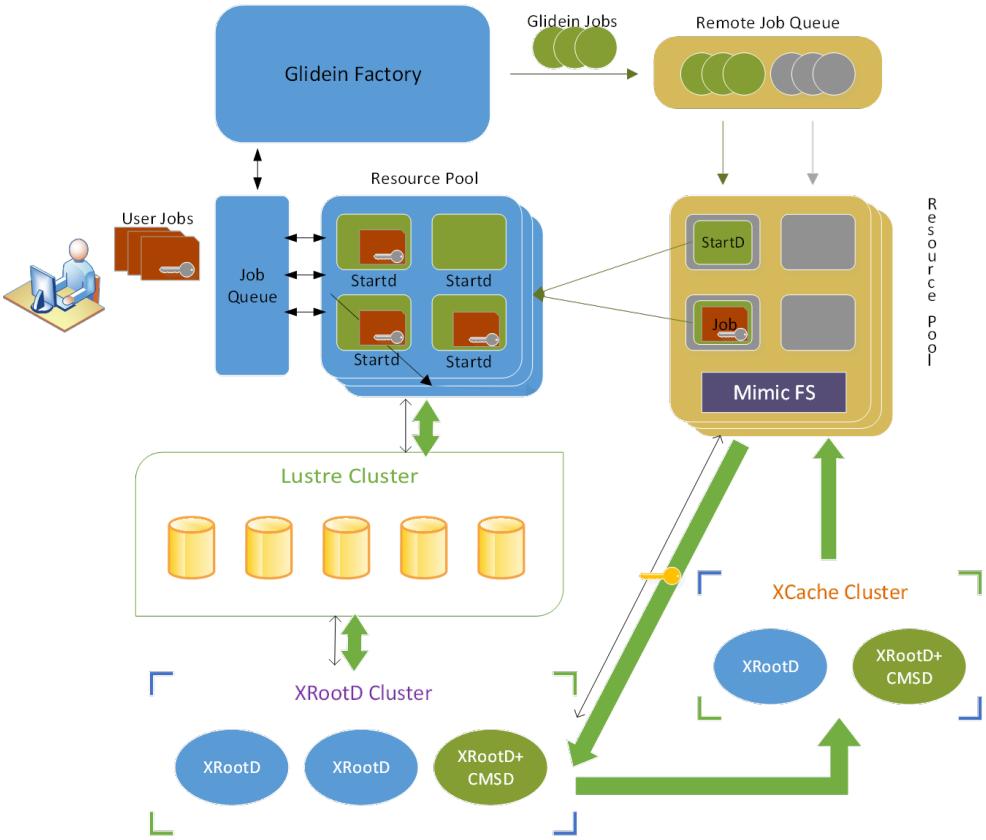

Figure 2. Using Kerberos tokens in distributed computing (*the key icon represents Kerberos token*)

4 Kerberos Tokens Validity Extension

The key problem of using Kerberos tokens in the distributed computing system is how to extend the validity time. The solution in this paper consists of three parts.

1. Extend the validity time when a job is submitted. With HepJob, the job submission tool in IHEP, the token can be transferred to the worker node with the job. When the job is submitted, the token validity time will be automatically extended to 2 days again. This function is implemented as a plugin and added into the experiment job workflow over the HepJob. This function can guarantee all jobs have a new validity time (2 days in the IHEP case) when the jobs are submitted. In addition, the function can also copy the token to the token repository, which is a common directory in this case. The token repository will be referred to in the following step.

2. Periodically extend the validity time when a job sits in a job queue. A job may sit in idle state for a long time when the resources are busy. In an extreme scenario, the idle time can be more than 2 days. That means the job will take a expired token to the worker node. The expired token cannot be fixed and used by the job any more and the job will exit with an error. So the token should be continuously renewed when the job is idle in the job queue. We implemented a simple service to watch the token repository and periodically renew all the tokens in the repository. This can make all tokens newly extended before the jobs are sent to the worker node.

3. Periodically extend the validity time when a job is running on the worker node. When a job starts running on a worker node, it may run for a time longer than 2 days. That means the token will be invalid when a job runs for longer than 2 days and the job will fail. A simple tool called 'token engine' is implemented and launched on the worker node when a job starts running. The token engine starts a service and keeps watching the Kerberos tokens stored in a specific directory and initialized in the job environment. Periodically, with a time interval of an hour in this case, the token engine renews the token until the renew time limit is reached or the job is completed. That can guarantee the token is valid during the job runs on the worker node.

5 Conclusion

The Kerberos token has been used in the distributed computing system at IHEP. With Kerberos tokens, a job can safely access the necessary services from the remote side, especially the storage services including Lustre, XCache and EOS. The key problem is how to guarantee the Kerberos token is valid during the whole lifetime of the job. This paper introduces the solution of extending the validity time of Kerberos tokens in the IHEP case. Currently, all the jobs running in the remote sites are using Kerberos tokens to access storage services. From February to May, there are 2 million jobs using Kerberos tokens in the distributed computing system, as shown in Figure 3.

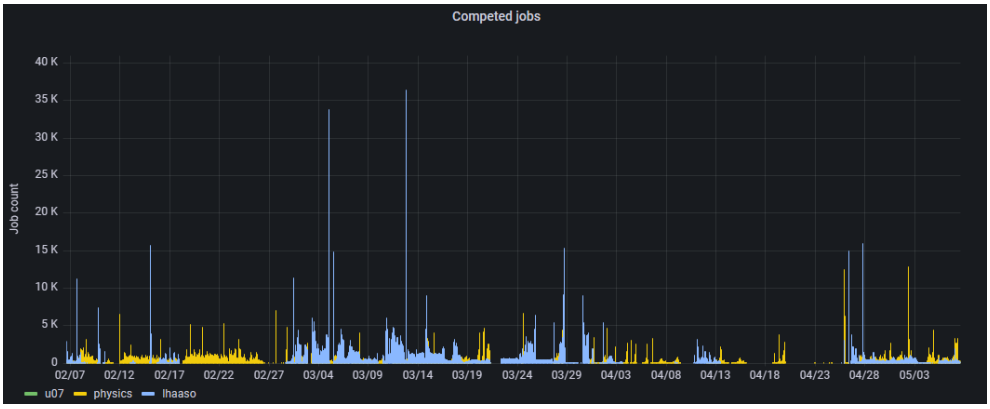

Figure 3. Status of jobs completed in the distributed computing system

The solution is also tested and deployed in the ARM worker nodes, and some of the above jobs used Kerberos tokens in the ARM environment.

6 Acknowledgements

Authors would like to thank all the colleagues who contributed ideas or developments to this work. Besides, this work was supported by several projects of the National Natural Science Foundation of China (No.12175255, No.11805226, No.12105303, No.11905239) and the Youth Innovation Promotion Association of Chinese Academy of Sciences (No.2020017).

References

[1] Bockelman, Brian, et al. "WLCG Token Usage and Discovery." EPJ Web of Conferences. Vol. 251. EDP Sciences, 2021.

[2] Kerberos Introductions. https://web.mit.edu/kerberos/, online, accessed 20-Sep-2023.

[3] Peters, A. Joachim, Elvin Alin Sindrilaru, and Geoffrey Adde. "EOS as the present and future solution for data storage at CERN." Journal of Physics: Conference Series. Vol. 664. No. 4. IOP Publishing, 2015.

[4] Dorigo, Alvise, et al. "XROOTD-A Highly scalable architecture for data access." WSEAS Transactions on Computers 1.4.3 (2005): 348-353.

[5] He, Huihai, and LHAASO collaboration. "Design of the LHAASO detectors." Radiation Detection Technology and Methods 2 (2018): 1-8.

[6] Ablikim, Medina, et al. "Design and construction of the BESIII detector." Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment 614.3 (2010): 345-399.

[7] Gargano, F., and HERD Collaboration. "The High Energy cosmic-Radiation Detection facility (HERD)." European Physical Society Conference on High Energy Physics. 10-17 July. 2019.

[8] Wang, Feiyi, et al. "Understanding lustre filesystem internals." Oak Ridge National Laboratory, National Center for Computational Sciences, Tech. Rep 120 (2009).

[9] OpenAFS Introductions. https://www.openafs.org/, online, accessed 20-Sep-2023.

[10] Hanushevsky, Andrew, et al. "Xcache in the ATLAS distributed computing environment." EPJ Web of Conferences. Vol. 214. EDP Sciences, 2019.