Extending an Open-Source Federated Identity Management System for Enhanced HPC Security

Jennifer Buchmüller*, Simon Raffeiner*, Michael Simon*, Holger Obermaier*,
Peter Weisbrod*, Ulrich Weiß*, and Martin Nußbaumer*
*Steinbuch Centre for Computing (SCC)
Karlsruhe Institute of Technology (KIT), Karlsruhe, Germany
Email: jennifer.buchmueller@kit.edu

***Abstract*—Strengthening the security infrastructure around HPC systems has become an urgent and important task, driven especially by the impact of a recent large-scale attack on the world-wide HPC community by a yet unknown party. Multiple European HPC systems had to be shut down for several weeks in mid-May of 2020 after backdoors were found on the systems. In the aftermath of the attack, two core security issues were identified: the absence of strong authentication, and a widespread practice of insecure handling of SSH key pairs. We present our approach for extending an existing, open source, federated identity management system with user-friendly two-factor authentication (2FA) using Time-Based One-Time Password (TOTP) and centralized, secure SSH key management. A special focus will be put on how we integrated scientific workflows and automation with the new security measures by combining 2FA, SSH key management and security policies in an elegant, secure and user-friendly way.**

In mid-May 2020 multiple European HPC centers discovered a backdoor had been placed on their systems. Unprivileged users were able to gain the highest privileges (root access) through a simple privilege escalation using a so-called set-uid (SUID) executable. To this date, the motives of the attackers are still unclear. On most systems no other indicators of compromise (e.g. botnet clients or cryptocurrency miners) besides the backdoor were found. The vector used for all attacks on these HPC systems was a successful login via SSH, using credentials stolen from legitimate users. The attackers harvested all useful data they could find on the compromised systems, e.g. shell history files, /etc/shadow files containing password hashes, passwords extracted from files and running services, SSH private keys, database files of password stores (e.g. Keepass), configuration files with hard-coded passwords (e.g. for VNC). SSH private keys not protected with a passphrase were used right away for further attacks. However, we discovered hints that the attackers might also have tried to crack SSH passphrases on dedicated systems. A number of user workstations and laptops, as well as some external workflow servers used to automatically send jobs to the HPC systems, have also been found to have been compromised. Most affected HPC systems had to be shut down until a complete re-installation and and evaluation of the security policies had taken place. The two high-performance computers ForHLR II (Tier-2) and bwUniCluster 2.0 (Tier-3) at the Karlsruhe Institute of Technology were put back into operation in mid-June of 2020. During the first (out of three) phase of the recommissioning process coordinated with the other operators in the federal state of Baden-Wuerttemberg, the use of SSH keys was no longer permitted. This caused severe restrictions for the scientific communities, especially on the Tier-2 system, since the HPC systems could no longer be integrated into automated scientific workflows. In our opinion, the enforcement of additional security measures on scientific HPC systems should never interfere with or restrict the scientists working on these systems. Coming up with a solution which would allow us to remove most of the restrictions, especially permitting our users to use SSH keys in a secure manner, was paramount.

Already in the past, the IT service operators of the universities of the state of Baden-Wuerttemberg were heavily dependent on the bwIDM federated Identity Management (IDM) system [1] for authentication and authorization. BwIDM acts as a single point of authentication for all services by implementing a gateway between the university's existing Identity Provider (IdP) servers and the individual services. Service providers are freed from the burden of having to maintain connections to each and every local IdP. Instead the bwIDM regularly imports all user IDs from all IdPs and presents each service with a full set of the information it requires. We use a facade of standardized service, to hide the real complexity of the heterogeneous services. With that we offer users the highest level of convenience by not having to register a new account with every service, and being able to use the same password. Individual organizations still manages what data is exported to bwIDM by the local IdP, so each of them still has the ability to define individual access and security policies for its users. BwIDM consists of a combination of various software components. The following list mentions all services provided by bwIDM and how the components play together:

- A user- and administrator-facing web frontend for registration and user/group management, provided by reg-app.
- An LDAP backend for user/group directory services and password authentication is handled by 13 load-balanced Apache DS and OpenLDAP servers, with reg-app providing the user databases and authentication scripts.
- A SAML/Shibboleth web service is provided by reg-app.
- An OpenID Connect web service is provided by reg-app.
- An HTTP REST interface is provided by reg-app.

Reg-app is an open-source Java application developed mainly at Steinbuch Centre for Computing (SCC) at KIT. It provides users and administrators with an user-friendly self-service web interface for common tasks and already handled security-focused functions like forwarding LDAP password requests to the IdPs, providing the SAML/Shibboleth and OpenID Connect web interfaces and managing individual service passwords for the services. It was therefore predestined as the component where additional security measures like two-factor authentication (2FA) and SSH key management should be implemented. Another option would have been to let every institution implement their own solutions within their respective IdP, but especially smaller universities often do not have the manpower for implementing such extensive changes on the spot. By extending a central component like reg-app, security could be strengthened for all users and services, while individual institutions could later still implement their own solutions. Also focusing regions of expertise improves the overall security concept.

Within just six weeks, reg-app was extended with both new features. The 2FA is based on hardware/software tokens and Time-Based One-Time Passwords (TOTP) [2] managed by a dedicated LinOTP server. This guarantees that an attacker cannot get hold of all credentials and secrets if one the central bwIDM servers is compromised. reg-app provides both the token management functionality as part of its web interface as well as a LinOTP-compatible HTTP interface facing the services. When more organizations in the federation set up their own LinOTP servers later on, the central reg-app will forward the TOTPs to the home institution of a specific user ID like it already forwards other requests (e.g. for LDAP passwords). KIT employees get a TOTP-compatible hardware token from their employer, while for users of other institutions we recommend hardware tokens made by Yubico or software tokens (usually in the form of a smartphone app). Smart cards, as mentioned in [4] were not an option since there is no federated PKI infrastructure and the organizations in the federation do not issue them.

SSH keys are very important for improved usability and especially automation, but the concept has several security issues. Keys do not expire automatically. On the server side, it is impossible to detect if a private key is protected by a passphrase or an additional hardware device on the user side. As a consequence, we disabled the ability for users to store keys in the authorized_keys files in their home directories. Instead the keys have to be managed via the web interface provided by reg-app, so at least minimum standards for key algorithm, key length and an expiration date can be enforced. The interface between the HPC systems and reg-app is implemented using a helper script configured as an authorized key command for the OpenSSH servers. Every time a user tries to log in with an SSH key, a HTTP request to the REST service is triggered and reg-app replies with a list of all SSH public keys it currently deems valid for the given user. This concept does not require modified OpenSSH binaries, additional modules and special apps as earlier solutions did (see e.g. [3]).

In our presentation we want to focus on a number of special features made possible by coupling 2FA and SSH key management. We decided to distinguish between two different types of SSH keys: those for interactive usage (interactive keys) and those for automation (command keys). Interactive keys can be used for normal interactive logins and are unlimited in the sense as to which commands can be executed. This makes interactive keys very powerful, so their usability is limited to a period of one hour after the last successful two-factor login. This means that on the first attempt to access the HPC system, the key will not be accepted by the OpenSSH server because the reg-app HTTP REST interface at first does not include the public in key in its response to the HPC system. The OpenSSH server then falls back into a request for a 2FA TOTP. After a valid TOTP input, the service password is required in a second step. Finally, if this two-step login was successful, reg-app will unlock all registered interactive keys by including them in the response of the HTTP REST interface for the next hour.

The usage of passphrases and TOTPs is contradicting to the interplay between automated workflows and the HPC systems. The second type of SSH keys - the command keys - are therefore special keys which can be used for e.g. scientific workflow systems, continuous integration and interactive data exploration. Command keys are always valid and do not have to be unlocked. This makes these keys extremely attractive to a possible attacker and poses a security risk, so we impose additional restrictions on these keys. They have to be restricted to a single command and to either a single IP address (e.g. the one of the workflow server) or a small number of IP addresses (e.g. the sub-net of the institute). Command keys also have to be checked and approved by an HPC administrator. The validity of a command key is one month after registration, while interactive keys are valid for six months.

In our presentation we will also cover non-technical, but critical aspects such as how to successfully communicate new security measures to users, the relevance of comprehensive documentation, and user support experiences.

References

[1] J. Köhler, S. Labitzke, M. Simon, T. Dussa, M. Nussbaumer, and H. Hartenstein. bwIDM - federated access to it-based services at the universities of the state of baden-württemberg. *Praxis der Informationsverarbeitung und Kommunikation*, 37(1):15–21, 2014.

[2] D. M'Raihi, S. Machani, M. Pei, and J. Rydell. TOTP: Time-Based One-Time Password Algorithm. RFC 6238, RFC Editor, May 2011.

[3] W. Cyrus Proctor, Patrick Storm, Matthew R. Hanlon, and Nathaniel Mendoza. Securing hpc: Development of a low cost, open source multi-factor authentication infrastructure. In *Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis*, SC '17, New York, NY, USA, 2017. Association for Computing Machinery.

[4] Andrew Prout, Anna Klein, Peter Michaleas, Lauren Milechin, Julie Mullen, Antonio Rosa, Siddharth Samsi, Charles Yee, Albert Reuther, Jeremy Kepner, William Arcand, David Bestor, Bill Bergeron, Chansup Byun, Vijay Gadepally, Michael Houle, Matthew Hubbell, and Michael Jones. Securing hpc using federated authentication. pages 1–7, 09 2019.