# Securing HPC: Development of a Low Cost, Open Source Multi-factor Authentication Infrastructure

W. Cyrus Proctor
Texas Advanced Computing Center
10100 Burnet Road
Austin, Texas 78758-4497
cproctor@tacc.utexas.edu

Patrick Storm
Texas Advanced Computing Center
10100 Burnet Road
Austin, Texas 78758-4497
storm@tacc.utexas.edu

Matthew R. Hanlon*
Texas Advanced Computing Center
10100 Burnet Road
Austin, Texas 78758-4497
mrhanlon@gmail.com

Nathaniel Mendoza
Texas Advanced Computing Center
10100 Burnet Road
Austin, Texas 78758-4497
nmendoza@tacc.utexas.edu

## ABSTRACT

Multi-factor authentication (MFA) is rapidly becoming the de facto standard for access to all computing, whether via web, phone, or direct command-line access. HPC centers and other institutions supporting hundreds or thousands of users face challenging cost, licensing, user support, and infrastructure deployment decisions when considering a transition to MFA at scale.

This paper describes our experiences and lessons learned throughout the assessment, planning, and phased deployment of MFA across production systems supporting more than 10,000 accounts. It focuses on the ultimate curation, creation, and integration of a multitude of software components, some developed in-house and built to be compatible within existing HPC environments, and all of which are freely available for open source distribution. We motivate the development of this customized infrastructure by highlighting some of the particular needs of our research community. What follows is an information resource for others when considering their own MFA deployments.

## CCS CONCEPTS

• **Security and privacy → Multi-factor authentication**; *Usability in security and privacy*; *Economics of security and privacy*;

## KEYWORDS

Multi-factor Authentication, Large-scale Deployment, Open Source Infrastructure

*Now at Oracle Corporation


*SC17, Denver, CO, USA*

DOI: 10.1145/3126908.3126957

**ACM Reference format:**
W. Cyrus Proctor, Patrick Storm, Matthew R. Hanlon, and Nathaniel Mendoza. 2017. Securing HPC: Development of a Low Cost, Open Source Multi-factor Authentication Infrastructure. In *Proceedings of SC17, Denver, CO, USA, November 12–17, 2017,* 11 pages.
DOI: 10.1145/3126908.3126957

## 1 INTRODUCTION

October 2016 marked the onset of required multi-factor authentication on all major HPC resources throughout the Texas Advanced Computing Center (TACC). The nine month endeavour leading up to this event resulted in the creation and subsequent operation of an infrastructure comprised of the amalgamation of off-the-shelf and custom-built open source components that provide a tailor-made MFA solution supporting all existing user accounts. This roll out came after extended trials and explorations with commercial two-factor vendors which proved either technically or financially unsatisfactory.

This paper provides a survey of our community's needs that helped to motivate the choices behind the crafted features custom-built to satisfy our diverse user base. Details of the construction, planning, and deployment of the MFA infrastructure are highlighted to underscore the flexibility and functionality that were essential for practical large-scale MFA implementation. We also discuss our experience pre-transition through the first six months of production MFA operation.

We highlight several important concepts for a successful roll out of MFA across HPC resources. An infrastructure must be cost-effective, provide opt-in MFA support to allow users to control when they transition, continue to support automated, non-interactive workloads for specific accounts or IP addresses, all while providing increased levels of assurance to intellectual property and enhancing overall service reliability.

The infrastructure front end of our implementation offers users a choice between three additional, mutually exclusive authentication options beyond traditional, first factor public key or password challenges. Second factor authentication options include: 1.) a free smartphone-based application downloaded via Google or Apple app stores, 2.) SMS text messaging services, or, for a small fee 3.) a key fob with LCD screen. Each of these options provides a six digit,

timed-based one time password, known colloquially as a token code. Users manage their MFA device pairing options within our web-based user portal, which is integrated directly with back end identity management services.

For the second factor, we developed custom Pluggable Authentication Modules (PAM) [16, 20] to integrate cleanly with common data transfer protocols, provide a mechanism for specific accounts or IP addresses to be exempted, allow both public key and password challenges to remain as the first factor of authentication, and are built to support opt-in or mandatory deployments designed for the transitioning of large user bases [21].

The Remote Authentication Dial-In User Service (RADIUS) [33] and Hypertext Transfer over Transport Layer Security (HTTPS) [25] networking protocols connect the back end core infrastructure of LinOTP [11] MariaDB device pairing database and RADIUS servers [23] to provide access control responses to vetted login nodes. Due to their broad adoption and ubiquitous natures, the RADIUS and HTTPS protocols provide for a flexible and extensible framework that can integrate a scalable number of back end components fitted to the size of the user base. Initialized within the PAM modules, RADIUS-compatible login queries invoked by users attempting to connect via Secure Shell (SSH) [34] govern successful authentication in a security-conscious, layered approach.

MFA transition preparation included preemptive communication with targeted user groups to better understand the potential impacts to HPC workflows. Center-wide coordination to prepare front and back end infrastructure, including installation, testing, documentation, and training set the stage for a phased roll out to begin early August, 2016.

Several straightforward strategies aided to adjust common behaviors incompatible in the new MFA paradigm with minimal interruption to productivity. As the phased MFA transition was orchestrated over a two month window, users were able to test MFA in their own time. Those that needed extra time or assistance were granted temporary MFA exemptions while specialized, trusted accounts or IP addresses serving automated, non-interactive workloads on behalf of satellite users continued without service disruption.

## 2 MOTIVATION

The importance of cyber-security to maintain trust, protect intellectual property, and provide reliable services is at an all-time high as the number of remote transactions continues to increase in virtually all forms of business and government [4]. The risks associated with poor cyber "hygiene" are commonly manifested as misrepresentations of identity or attempts to authenticate to gain access to goods or services in which the attacker has no legitimate claim [3]. Single-factor authentication methods, e.g. passwords alone, pose an increasing risk to service providers (SPs), as users tend to reuse passwords across systems providing valuable targets for attackers [28, 31]. Cyber defense measures, including multi-factor authentication, raise the cost to attackers but have largely been neglected until recent years due to implementations being particularly expensive and not easily scalable [28, 32].

A number of businesses and collections of SPs collaborate to provide various standardized platforms in which MFA becomes a service itself [1, 7, 17, 18, 35]. Typically, fees are incurred on a per user basis in a subscription-style business model. MFA solutions of this type can quickly become cost prohibitive when the number of supported end users is taken into consideration at the scales many SPs need. Moreover, many of the off-the-shelf implementations remain proprietary, precluding the ability for feature additions or customization by the intermediary. Other open source possibilities exist [11, 23], but lack the technical capability, e.g. flexible MFA exemption policies, to accommodate all the needs of our use base.

Entry into TACC's HPC systems occurs predominately in two forms, both of which utilize the SSH network protocol. Thousands of users enter directly through SSH clients onto public-facing login nodes. That number again interface through trusted web portals and specialized accounts, i.e. gateways and community accounts. These entities negotiate in an automated fashion on behalf of these users, some of whom do not directly possess accounts for the HPC resources they ultimately consume.

Our challenge was to transition the majority of the center's HPC resources over to MFA in such a way as to provide a higher level of security while attempting to maintain usability for all methods of remote access. To accomplish this, the decision was made to build upon existing open source infrastructure to provide a tailor-made, low cost implementation capable of scaling to our entire user base while accommodating SSH, gateway, and community account access. The goal became to construct an infrastructure that integrated many available off-the-shelf components while maintaining common compatibilities, add in features to support the automated traffic necessary to gateway and community accounts, and provide a means for a tiered, opt-in policy to aid in the migration of the user community from single-factor to multi-factor authentication.

## 3 IMPLEMENTATION

The major front and back end components that comprise our custom MFA infrastructure are detailed in the following subsections. On the back end, LinOTP and RADIUS servers provide the means to validate a user's second factor. PAM modules negotiate challenge-response queries to the remote user via SSH sending a time-based one-time password (TOTP) on to the back end for verification [19]. Users interact directly with front end components including the user portal and a device that they possess that provides the one-time password.

### 3.1 LinOTP Server

LinOTP is an open source OTP-platform developed and maintained by KeyIdentity GmbH [11]. It supports many device token types, authentication protocols, and user directories. In our implementation, a server is set up to provide a repository that keeps track of users and their associated one-time password secret key. The LinOTP user repository is an encrypted MariaDB [8] relational database that extends an existing identity management database reserved for Lightweight Directory Access Protocol (LDAP) [27] queries. When a user account is created, an LDAP entry is generated including a unique user ID that becomes common to both databases.

System administrators utilize a built-in interface that allows for web-based browser interaction via a local Apache [9] service. Admins can view user pairings, re-synchronize tokens, access audit

logs, and clear failure counters associated with consecutive unsuccessful MFA log in attempts on public-facing login nodes.

The LinOTP database is updated via integration with the user portal. Firewall rules isolate the server from the surrounding center's internal network with the exception of trusted RADIUS servers that act as common connectors between login nodes and the LinOTP database.

LinOTP provides an additional configurable security feature to lock out accounts that have experienced multiple consecutive failed token code queries. A threshold of 20 consecutive failed attempts must occur before a user account is temporarily deactivated. As described in more detail in Section 3.4, the second factor can only be attempted once a correct password or authorized public key have been used as a first factor. This effectively filters most illegitimate SSH traffic before the second factor is ever reached. In the event that LinOTP does deactivate a user account, this information is available to staff via an internal website to help troubleshoot with the user.

## 3.2 RADIUS Servers

A handful of servers were set up to accept and proxy requests between authentication agents, i.e. login nodes, and the LinOTP server. These requests are generated by PAM modules when a user attempts to log in and has entered their device token code. The token code is sent using challenge-response functionality of the RADIUS protocol to a server that then negotiates a response from the LinOTP database. Upon validation, an audit log entry is created within the LinOTP database, the provided token code is nullified, and the initiating RADIUS server passes a message back to the authentication agent with a success message. In the event of a token mismatch, the token code remains valid and a failure message is sent instead.

The servers run software released by the FreeRADIUS project, an open source community that serves as the basis for multiple commercial products within Fortune-500 companies and Internet service providers [23]. This common protocol serves as middleware for handling authentication requests and allows for flexible deployment that is capable of load balancing and proxy chaining across servers.

## 3.3 Token Devices

To provide means for a second factor of authentication, users must possess a device that delivers a six-digit, time-based one-time password, i.e. a token code. A code is generated every 30 seconds using the combination of the current time and a secret key. The key is unique to a user and stored in the LinOTP back end database. We offer three methods that a user may choose when pairing a specific device to their secret key: 1.) a free, open source, in-house developed, smartphone-based application downloaded via Google Android or Apple iOS app stores, 2.) SMS text messaging services mediated by Twilio Inc. and served to all common cellphone network providers, or, for a small fee 3.) a key fob with LCD screen sourced from Feitian Technology Company. These methods are known colloquially as a 1.) soft token, 2.) SMS token, and 3.) a hard token respectively.

Both soft and hard tokens are considered "single-factor one-time password devices" while the SMS token is considered an "out of band token" by the National Institute of Standards and Technology (NIST) [2]. Combining one of these three tokens with either a password or authorized public key increases our Level of Assurance, a common measure of security in federal remote authentication guidelines, from a level 2 to a level 3 on a scale from 1 to 4 with 4 being the highest practical remote network authentication assurance.

The smartphone-based application soft token is modeled after an open source release of the Google Authenticator application [12]. The code bases for both Apple and Android platforms were updated, reskinned, and outfitted with the ability to read a quick response (QR) code with the user's smartphone camera. The QR code is provided when pairing the smartphone device and contains the user's unique secret key. After a period of internal testing, the applications were sent and vetted by the application stores to be provided free of cost.

Having stored the secret key on the device, the soft token does not require any sort of remote network communication to provide the user with their token code. The only requirement is that the smartphone keep a time that does not drift more than a time delta of 300 seconds from the LinOTP server's time (timezones are accounted for).

If a user chooses to pair with an SMS token, they provide a US-based phone number tied to a phone compatible with SMS text messaging services. LinOTP was configured to interface directly with the SMS provider, Twilio Inc. [14]. When a user attempts to log in, a null RADIUS response is forwarded to LinOTP which triggers a request to Twilio to send a text message that is then picked up by the user's phone network provider. The SMS is delivered and the user receives their six-digit token code. While the token code is active, if another request is made, LinOTP will not forward to Twilio and instead RADIUS is leveraged to display a response message to the user notifying them that the SMS has already been sent.

Twilio provides SMS text messaging services for a flat rate of $1 per month plus each US-based text message costs an additional $0.0075. Users may incur standard carrier text messaging rates depending on their service. International text messaging services can also be provided but cost more.

Hard tokens were sourced in batch from Feitian Technology Company as an alternative to soft and SMS tokens, both of which required the user to possess a mobile phone [6]. The availability of hard tokens was an important component of our MFA infrastructure as a non-neglible number of users either worked at locations where phones were not permitted, lived outside the United States, or did not own a compatible phone. The single button TOTP hard tokens came pre-programmed with a secret key, all of which were provided at the time of batch purchase.

A limited evaluation of other Initiative for Open Authentication (OATH) TOTP compliant manufacturers and token varieties took place before deciding on the Feitian OTP c200 model. Ultimately, price and the ability to customize the faceplate and backplate of the token fob lead to this choice. A sample set was first ordered, followed by a customized token fob proof. A bulk shipment arrived 5 weeks after initial purchase.

Users were able to acquire the hard tokens online via a web-based store for a fee of $25 to help cover the cost of the device, shipping and handling, as well as staff time for processing. Once the hard token is received by the user, the user uses the serial number on the back of the token to pair within the user portal. The tokens have proved to be reliable during the first six months of MFA production operation and have been shipped throughout China, Germany, the United Kingdom, Switzerland, France, Spain, and the United States.

Lastly, a fourth token type, not available to the public, is used specifically for training accounts. These accounts are managed and maintained for training activities at workshops, tutorials, conferences, and institutions to help expose users to HPC resources who otherwise do not possess an individual account at the center. LinOTP provides the capability to set a static, six-digit token code for individual accounts. Before each training session, accounts are assigned a random six-digit number such that the participants may step through the multi-factor authentication process without needing to set up or possess a device tied to their identity directly. This compromise allows users to experience how to log in to the center's systems without needing the extra time and instruction to provide full accounts. The static token codes are easily regenerated once the training session is finished.

### 3.4 Pluggable Authentication Modules

The Pluggable Authentication Modules (PAM) library has existed as a standardized API for system entry services since 1996 [26]. New authentication methods may be added by installing new PAM modules and updating authentication policies controlled via configuration files. Used throughout our center as well as the Linux, BSD, and Sun communities [16, 20, 29], PAM was the natural choice for integrating MFA into existing infrastructure.

In all, four new PAM modules were created: a module 1.) to check the success of SSH public key authentication, 2.) to check if an MFA exemption has been granted, 3.) to check if an MFA token code was correct, and 4.) a module specific for use on Oracle Solaris operating systems that combine the public key and MFA exemption checks to accommodate differences in PAM stack processing logic [21]. To illustrate, Figure 1 provides a representative PAM stack incorporating these modules (outlined in bold blue) to determine if SSH entry via MFA will be granted or denied. SSH would be configured to test for an authorized public key and then hand off the authentication decision, including password check, if necessary, to PAM. These modules would be customized by the system administrator to determine how system entry will be allowed.

The first PAM module in the stack (Figure 1 – "Public Key Success?") has been constructed to determine if a user has utilized public key authentication successfully via SSH as their first factor of authentication. This module searches recent local secure system entry logs to determine this information. If confirmation was found in the system authorization logs, the user need not be prompted for their password and is instead directed to the MFA exemption module. Information about the state of public key authentication is not provided from SSH to PAM. This module is the only mechanism known to provide this information and plays a crucial role as one of the necessary components for a practical MFA implementation.

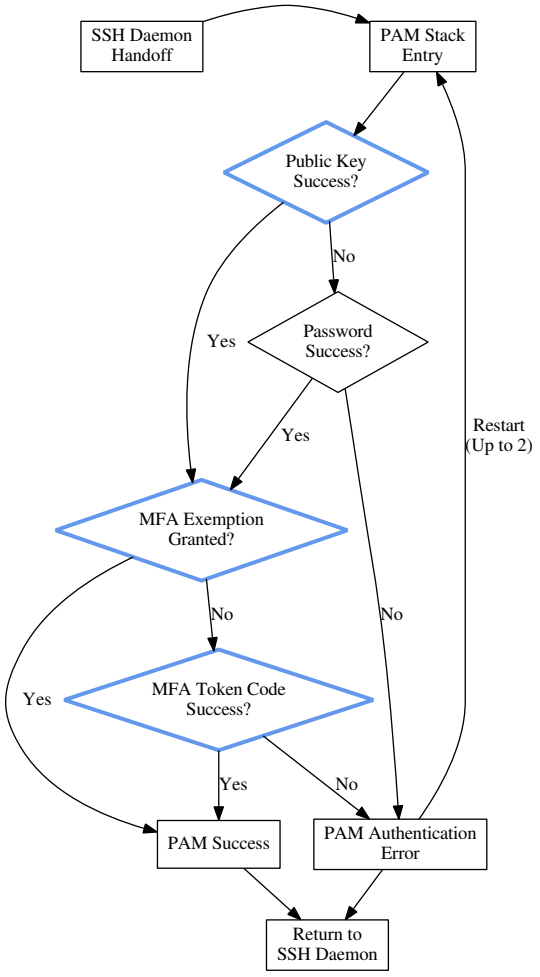


**Figure 1: A representative Linux PAM authentication stack for SSH log in with MFA. Open source, in-house created PAM modules are highlighted by bold blue borders. This decision tree illustrates how considerations for first and second factors of authentication are integrated together to provide flexible and powerful configuration options that satisfy the needs of our HPC user base.**

In the event that authorized public key authentication has not been set up by the user or that the verification has failed, an existing PAM module instead ensures that the user enters an appropriate password as their first factor of authentication. If the password entry is incorrect, the PAM stack is restarted and the user is prompted once again for a password, up to a maximum of two more times before SSH disconnect. Only when a correct password is entered or an authorized public key is provided does the processing of the second factor of authentication begin. This helps to mitigate brute force attacks coming from outside the center's internal network.

To begin the second factor, the user's information, including username and remote IP address are compared with an existing configuration file that contains white and blacklists specific to the second factor of the MFA process (Figure 1 – "MFA Exemption

Granted?"). If one of the pieces of information matches a particular list entry, appropriate action is taken to grant or deny an MFA exemption for the user. If an exemption is granted, no further action by the user is required to gain SSH entry into the system. If an exemption is denied, the user will then continue on to be prompted for their MFA token code by another module. In the event that a user account is outfitted to use public key authentication and the account has been granted an MFA exemption, log in may occur uninterrupted. This is of particular use in allowing our trusted gateway and community accounts to continue automated, non-interactive transactions on our HPC systems.

MFA exemption control lists may be configured for each system individually and can be updated at any time during production. Changes take effect immediately upon write to disk. The configuration file extends typical PAM access configuration syntax and allows for either permanent exemptions or for temporary variances that will automatically expire if the date has passed. Individual accounts, specific IP addresses or IP ranges, or any combination of the two may be targeted for MFA exemption with or without an expiration date. Additionally, special "ALL" keywords can be set in the date, account, and IP address fields to allow blanket policies that cover either all time, all user accounts, all IP addresses, or any combination thereof. By default, all accounts are subject to multi-factor authentication and are denied an MFA exemption. System administrators must add in MFA exemption policies explicitly. This mechanism allows for dynamic, powerful, and scalable configurations to accommodate specific requests that could not otherwise be similarly entertained by other MFA implementations that we are aware of.

The module that provides the second factor of authentication via device token code challenge-responses to the user does so using the RADIUS protocol (Figure 1 – "MFA Token Code Success?" and Figure 2). These API calls communicate with RADIUS servers in a round-robin fashion to provide load balancing and resiliency if specific RADIUS servers are unavailable. If the token code entered by the user is valid, the RADIUS server will return success, which in turn, will exit the PAM stack and grant SSH entry. The token module queries for existing LDAP entries on the authenticating user to distinguish between possible authentication routes, shown in Figure 2. If a user has an SMS device pairing, a null request is first sent to the LinOTP back end to initiate a text message to be sent to the user. Once this has occurred, or if the user has paired with either soft or hard tokens, a challenge-response is presented to allow the user to enter their six-digit token code. This code is delivered to the LinOTP back end database for validation. If the user-generated token code matches the LinOTP-generated token code, success is reported back to the PAM module, which then exits successfully releasing back to SSH for entry into the system.

The PAM module that validates token codes may operate in any of four modes controlled within a PAM stack configuration file. This "enforcement" mode gives system administrators fine-grain control of a four tier, opt-in MFA enforcement policy. This is designed to assist with the transitioning of large user bases from single-factor authentication to multi-factor authentication. Any of these modes may be set during production operation and are in effect as soon as written to disk.

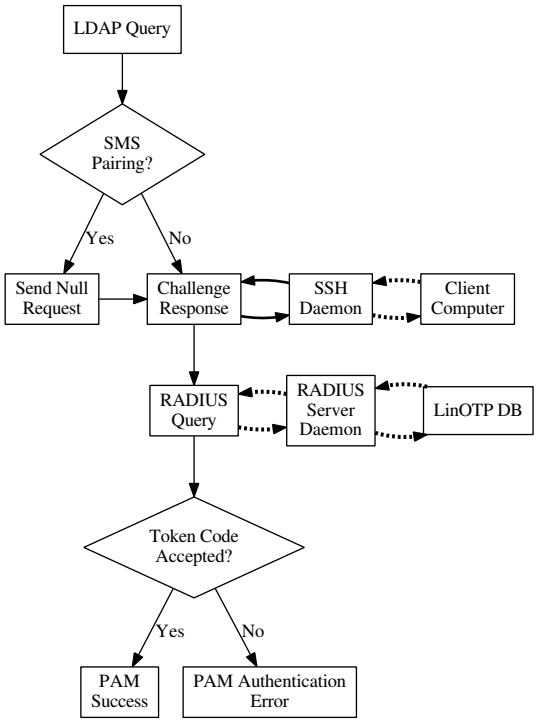


**Figure 2: In-house created PAM token module decision tree when MFA is active, i.e. "full" mode. An LDAP query is used to check the user's MFA pairing type. The user is sent a challenge-response query to enter their token code. The entered code is validated on the LinOTP back end. Either success or authentiation error is returned from the token module. Dotted arrows represent remote requests while bold arrows indicate interaction outside PAM back to the SSH daemon.**

The first mode, i.e. "off", deactivates the token module entirely, exiting with success. This effectively drops the system back to single-factor authentication.

The second mode, i.e. "paired", checks to see if a user account has paired an MFA device with an LDAP query. If a device pairing is found, the user is prompted for their token code. If no device pairing is found, the token module exits successfully without denying entry to the user.

In the third mode, i.e. "countdown", the token module once again checks via an LDAP query for a user's device pairing. If a device pairing is found, the user is prompted for their token code. If no device pairing is found, a message is sent via challenge-response to the user to inform them that they have "x" number of days to pair a device and that they may visit URL "y" for more information. The time delta between a configured deadline date and the current date are used to calculate "x". The deadline date and URL ("y") are configured by the system administrator in a PAM configuration file as the deadline date in which MFA will be mandatory and the user portal tutorial web page for setting up their MFA device respectively. Further, the user must press return to acknowledge

that they have read and received this statement. The token module then exits successfully without denying entry to the user. If the configured countdown date expires, the token module will default to the fourth mode.

The fourth mode, i.e. "full", checks for a user's device pairing type via LDAP. Regardless of whether a device pairing is found, the token module prompts the user for a token code. The response is sent to the LinOTP back end for validation. If no valid pairing is found, entry is denied into the system. Lastly, if any configuration errors occur, the token module defaults to the fourth enforcement mode.

We are not aware of any similar configuration options within other MFA implementations where users may opt-in to MFA simply by a device pairing. This allows users to start MFA when they are ready and only after their own extensive testing. Section 5 details how the phased transitional periods played out as these features were tested with thousands of users who were able to control their own MFA device pairing and unpairing via the user portal.

Within each HPC system, an MFA exemption is configured to allow any SSH traffic to move freely from IP addresses that are a part of that particular system. This allows users to move back and forth freely within login and reserved compute nodes without being prompted for a second factor of authentication. Additionally, remote storage systems are configured to accept SSH traffic from all HPC systems within the internal network. This allows for batch transfer of files to remote storage systems from shared file systems attached to either the login or compute nodes. This feature is an integral requirement in our implementation of MFA that satisfies the needs of users to transfer files remotely in a non-interactive manner as their jobs run without their presence.

### 3.5 Portal Integration

Users manage their own MFA device pairings via our web-based user portal. This is the same portal through which they manage their accounts, allocations, and other interactions with the center. After a successful log in to the user portal, the user's current device pairing status is checked. If the user has an MFA device set up, the log in continues normally. If no multi-factor device is configured, then the user is directed to an interstitial page, i.e. a "splash screen", explaining the multi-factor authentication requirement for system entry and prompting the user to configure an MFA device. Users can dismiss the prompt, but they are re-prompted upon each log in.

All users share a single, post-registration process for managing their multi-factor device pairing. This decouples the already existing account registration process from the MFA management process. Separating these two inherently complex operations benefits portal developers and users alike, promoting streamlined code verification while minimizing potential user support issues. Additionally, this provides a straightforward mechanism to incorporate existing, pre-MFA user accounts smoothly without the need for potentially convoluted MFA retrofitting. Finally, there exist scenarios where a user may wish to set up an account with the center that would not immediately require multi-factor authentication, e.g. subscribing to a user news RSS feed or registering for in-house training. This design choice enabled staff to focus on ensuring the MFA transition process was as smooth as possible for all users.

The user portal MFA device pairing functionality itself is set up as a portlet application as part of the center's overarching Liferay Portal deployment [13]. This application shepherds communication between the LinOTP back end, the user and their multi-factor device, and the center's identity management back end. The information sharing between components is minimal to maintain an information firewall between different pieces of the multi-factor authentication process. When a user first sets up an MFA device pairing, information is exchanged between the MFA device and the LinOTP back end to coordinate the token secret key that will be used to generate token codes. After a successful pairing setup through the portal, the portal notifies the identity management back end that the user has configured multi-factor authentication and which method the user has configured, i.e. soft, SMS, or hard token.

The portlet application communicates with the LinOTP back end via an administrative interface, which is available as a Representational State Transfer (REST) interface. The portal back end authenticates to the admin API using HTTP Digest Authentication over a TLS-secured connection. Through this interface, the portal is able to perform all necessary operations to manage user token information in LinOTP.

The MFA portlet application is hardened to handle form resubmissions and replays that could otherwise raise errors or cause problems during the MFA pairing process. The pairing process itself is a stateful operation between the browser client and the portal back end. This means that the complete pairing process occurs without a page refresh. If a user refreshes in the middle of the process, e.g. after requesting a token but before confirming it, the process is aborted and the user will have to restart from the beginning. This also protects against using the browser's back button to go back to the pairing setup page after a successful pairing.

When users initiate a device pairing, they choose whether to pair using a soft, SMS, or hard token device already in their possession. During a soft token pairing, the user is shown a QR code which contains the user's secret key encoded as an image that can be scanned by the mobile application for import. After scanning the QR code, the mobile application immediately presents the user with a six-digit token code. The user then enters the displayed code in the portal to confirm receipt. Upon successful validation of the token code with the LinOTP back end, the identity management back end is notified that the user has paired using a soft token device.

During SMS device pairing, the user is prompted to enter a ten-digit, US-based phone number. Upon submission, the phone number is sent to the LinOTP back end to create an SMS pairing. The LinOTP server generates and stores a secret key for the user with the associated phone number. The portal then triggers the LinOTP server to send a token code to the user via SMS text message. The user enters the token code in the portal to confirm receipt. Upon successful validation of the token code with the LinOTP back end, the identity management back end is notified that the user has paired using an SMS token device.

When a user orders a hard token, the device, having already been configured in the LinOTP back end, is shipped to the user. Upon receiving the hard token device, the user then visits the portal and

enters the serial number of the hard token into the pairing setup page. This confirms that the user is in possession of the device. To complete the pairing process, the user is then prompted to enter the current token code into the portal, which is then validated by the LinOTP back end. This ensures that the hard token device is working properly after shipment. Once validation of the token code is successful, the identity management back end is notified that the user has paired using a hard token device.

The portal application supports users needing to update their MFA device pairing, e.g. after getting a new mobile phone or a new phone number. Ideally, users will still have in their possession the current active token device. To begin, a user visits the portal and selects to remove their current multi-factor device pairing. If they are using the soft or SMS token device pairing, the user is prompted to enter the current token code displayed in the mobile application or received via SMS, respectively. Once the token code is successfully validated by the LinOTP back end, the portal affirms that the user is in possession of the current MFA device and then proceeds to disassociate the paired device and notifies the identity management back end that the user has unpaired the device. After removing the MFA pairing, if the user does not pair a new device, they will again receive notifications to set up multi-factor authentication when logging in to the portal.

In the case that a user no longer has access to the current active token device, e.g. a broken mobile phone, we support an out-of-band unpairing process. The user is sent an email to their associated account email address that contains a signed URL. Following the URL in the email ensures that the user is in control of the email address on file for the account and will allow the user to remove the current MFA pairing.

Support is not provided for the unpairing of a hard token device via the portal. Instead, users utilizing hard tokens for MFA are asked to submit a request directly to the center's user support ticketing system to permanently disable the hard token's use. This allows staff to keep an audit of which hard token fobs are currently active.

# 4 PLANNING

Deployment at this scale necessitated center-wide engagement and coordination between all groups to prepare and communicate for an MFA roll out to occur before the end of 2016. The steps outlined in this section provide a rough chronological frame of reference beginning in May and ending in October.

## 4.1 Information Gathering

To help audit the methods by which users were logging in, a script was installed throughout major systems to create a log event upon successful entry with explicit information pertaining to the user's current shell properties and whether a terminal session (TTY) had been initiated. These messages were aggregated over a period of months to get a sense on how many users' workflows would be impacted by the implementation of MFA.

From the logs, a non-negligible number of user accounts, on the order of hundreds, clearly were automating log ins. In this case, a minority of users were responsible for the majority of entries via SSH into the systems. This confirmed that further exploratory investigations needed to be conducted to ascertain what typical automated workflows from this subset of the user community were being carried out.

Users were ranked by the number of log in events in a fixed time period. Any known gateway or community accounts, where log in behavior could be inherently understood, were filtered out and contacted separately. As a small sample but good point of reference, staff members, who generally tend to be quite active on the systems, served as threshold cutoffs. Any user more active in log ins than this threshold were separated out to be targeted for inquiry into the nature of their workflows.

A sample of users from this subset were preemptively contacted via email in several waves by HPC staff to understand how users were interacting with the systems in such a way as to generate the volume of log in events seen. The far majority of these log in events were not invoked with a TTY. That is to say, most of the log in events were scripted and run on a remote host or involved some sort of remote data movement (e.g. via SFTP, SCP, rsync, or non-interactive SSH commands). In some limited cases, this inquiry led to the discovery of groups of users that were sharing accounts. This provided an opportunity to promote better security practices within our user community.

## 4.2 Preparation

Transitioning to include a second factor of authentication for all users logging in to the center's systems via SSH was formidable not only from a staff perspective but also from an infrastructure perspective. The decision to provide a tiered, opt-in strategy with increasing levels of communication targeted at users not signed up for MFA was designed to help alleviate the number of user support tickets open at any given time. This was to enable staff members, who are roughly outnumbered by SSH users a hundredfold, to more effectively prepare and handle incoming support inquires.

To begin, on the back end, LinOTP and RADIUS servers were deployed to allow internal staff testing in regards to configuration and integration with existing LDAP infrastructure. PAM modules were written and tested to accommodate a tiered, opt-in adoption approach. During scheduled routine maintenance, SSH and PAM infrastructure was installed throughout the login nodes on several systems. This provided the first production implementation that allowed staff members to test all major components together without need for further service interruption.

Development web pages for tutorials and pairing infrastructure within the user portal were created, while the existing user account management database was updated to store the current state pertaining to user's MFA pairing status. Interfaces between the LinOTP server, LDAP servers, and the user account database endpoints were constructed for the user portal to orchestrate the pairing and unpairing of an individual's device used to provide the token code associated with their second factor of authentication.

An internal wiki served as a collection point of information for staff to communicate an overview on MFA, important dates, procedural instructions, and strategies for assisting users in transitioning their workflows to MFA. The document grew to house the collective experience of staff interacting and troubleshooting with users who either had concerns about MFA or had workflows incompatible with our MFA implementation.

As the MFA infrastructure matured, core staff began inviting other internal staff to opt-in and provide feedback on technical and communication issues. This phase occurred over a month starting in July to identify corner cases, pairing issues, and allowed for updates to the language used in the user portal, tutorial materials, on the command-line, and within the smartphone applications.

Staff training sessions helped clearly define responsibilities and protocols across the center to mitigate confusion, incorrect information, and associated delay when answering support inquiries. Moreover, this enabled staff to possess enough working knowledge that even if the answer was unknown they were empowered to seek more information independently before needing to elevate questions.

In August, with both front end and back end services vetted, the first communications to the public were sent out via portal user news and mass email. Users were invited to opt-in to MFA with the understanding that it would become mandatory on September 27th, 2016. While targeted users had already been contacted, others were encouraged to proactively reach out to staff members via our ticketing support system if they had concerns or questions related to MFA. This provided an opportunity for individuals to try out MFA before it was mandatory and determine if there would be any significant changes to their workflows. Staff were instructed in a number of strategies to help support users' transitions. Depending on the scenario, if it was clear that more time would be needed to make the full transition, staff could request a temporary variance from MFA for specific users beyond the September 27th deadline.

In late August, any new users signing up for access to center resources for the first time began receiving instruction on how to pair an MFA device. This included updating automated mail deliveries with tutorial information and links within the user portal. Additionally, from this time, any user who logged into the web portal would be prompted by the interstitial page described in Section 3.5 to pair an MFA device.

Not communicated to the public were the transition dates in which messages would begin to appear on the command-line prompts of users who had not signed up for MFA yet. These were designed to provide information, first, passively with a message displayed for interactive shells that would appear only when the user initially logged in. In the second phase, users who had not paired were given a second prompt after having entered their password in which acknowledgement via a return key press was mandatory. In the last phase, MFA itself became mandatory and an updated SSH banner with instructions was put in place to greet all incoming users. Those who had not paired a device would not be able to enter the systems.

## 5 ROLL OUT

Beginning in July, back end components had been installed on several major HPC systems for full scale beta testing. User portal pages were up, but hidden while pairing device mechanisms were integrated and tested. PAM modules were in place and set to the "paired" opt-in mode described in Section 3.4. Staff members helped to debug and polish front-facing media for the first public announcement regarding multi-factor authentication.

In parallel, targeted users as described in Section 4.1 were contacted to determine what comprised the majority of their SSH login traffic. The majority of traffic from these users was scripted, non-interactive workflows usually moving data or checking on job status. During this time period through September, staff members worked with these users to come up with strategies on how they could continue their research with as little disruption as possible from MFA.

Sometimes, strategies as simple as utilizing a workload manager's email capability to notify a user on job start or completion was set up instead of using a cron job on their remote client that logged into the system to perform this same task. For some, creating communal, group-shared directories with proper default umask and/or Access Control Entries (ACEs) allowed users to share data directly instead of accessing a shared account to log in. Workload manager job dependency options enabled users to automate and submit more jobs without needing to make an interactive decision. Several SSH and SFTP connection clients support a keyboard-interactive challenge-response second factor. Popular choices that were tested and either worked out of the box or needed minimal settings updates included PuTTY [30], Bitvise [15], WinSCP [24], Filezilla [22], and Cyberduck [10].

Helping to set up user cron jobs on HPC login nodes that performed tasks such as data transfer, backup, synchronization, or job management were popular choices. Some users opted to reverse their SSH connection if there was no MFA or firewall rules precluding the reversal. Others set up SSH File System (SSHFS) connections to mount a remote HPC file system on their local machines. Perhaps most popular of all was the adoption of SSH multiplexing which allowed for one connection to be established via MFA and subsequent connections to the same host to utilize the already existing SSH connection [5].

The public transition period from single-factor authentication to multi-factor authentication lasted just under two months from August 10th to October 4th. This period is broken into phases 1 and 2 that correspond to the "paired" and "countdown" modes of the PAM token module described in Section 3.4. The transition from phase 1 to phase 2 occurred on September 6th. After October 4th, MFA was mandatory and is denoted as phase 3. These phase numbers appear throughout Figures 3 - 6.

Figure 3 displays the total number of unique users logging to in HPC resources broken down by day. A steady increase of users using MFA throughout phases 1 and 2 suggest that adoption was well spread over this time period. By the time MFA was mandatory in phase 3, the number of users using MFA was near its maximum for the remainder of the year. A noticeable discontinuous increase does occur on September 7th, the day after phase 2 begins. A decline in unique users is noted during the winter holiday.

Figure 4 provides insight into the SSH traffic on all HPC resources during the MFA transition. Traffic is first broken into two categories: internal and external. Internal traffic is a label used to denote SSH traffic with an origin IP address coming from inside the center's IP range to a login node. External traffic is any SSH traffic with an origin IP address coming from outside the center's IP range to a login node. Next, traffic can be separated into two further categories. Either the traffic utilized multi-factor authentication or it did not.

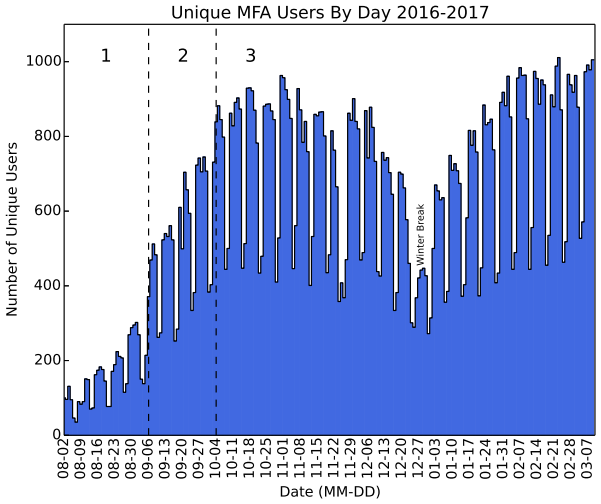


**Figure 3: Number of unique MFA users broken down by day. Voluntary, opt-in MFA adoption trends are shown in phases 1 and 2 before becoming mandatory in phase 3. Most users had already paired an MFA device before the mandatory deadline of October 4th.**

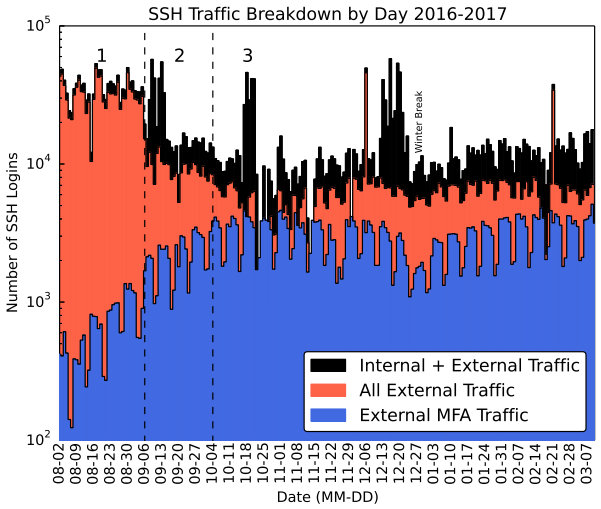


**Figure 4: SSH traffic broken down by day. Traffic is categorized as internal or external to denote connections originating either inside or outside the center's network space respectively. Traffic either authenticates with an MFA device, or does not. Blue bars represent external connections utilizing MFA. Red bars represent all external connections, including those that do not use MFA either because of an MFA exemption or because it occurs during a voluntary, opt-in phase. Black bars represent all MFA internal login traffic as well as external traffic.**

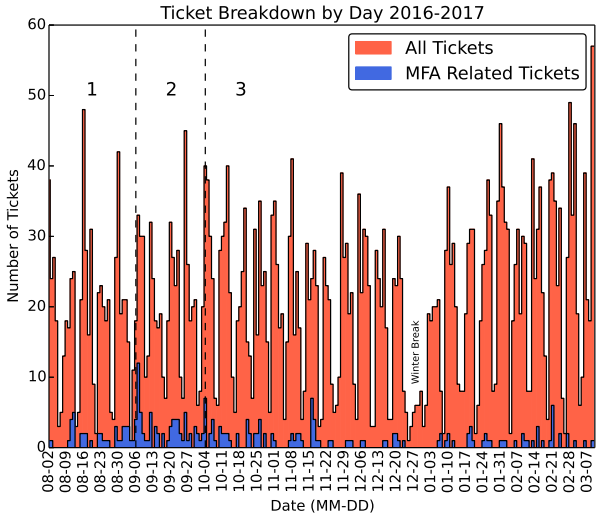


**Figure 5: Number of user support ticket inquiries broken down by day. MFA inquiries became a consistent but relatively small amount of the ticket load throughout phases 1 and 2 while waning after the beginning of phase 3. After the transition, inquiries were generally either from new users or those who wished to change their MFA device pairing.**

The reasons for why a connection would not utilize MFA vary, but include two typical possibilities. One, if the traffic was initiated during phase 1 or 2, MFA was voluntary and therefore could be bypassed. Two, if a specific account or IP range had been granted an MFA exemption, either as a gateway or community account, or as a user who needed a temporary variance, the SSH traffic did not require MFA. This traffic occurs throughout all three phases. In the figure itself, the blue bars show external SSH traffic utilizing MFA. Red bars denote all external SSH traffic, including traffic that did and did not use MFA. Black bars depict both internal and external SSH traffic. By design, most all internal SSH traffic was exempted from MFA as described in Section 3.4. The difference between black and red bars is the amount of internal SSH traffic occurring on login nodes. This traffic was not particularly affected by the transition to MFA. The differences between red and blue bars gives an indication of the amount of automated, non-interactive SSH traffic present on the systems. It is clearly seen that there was a significant decrease in this type of traffic once phase 2 began. Even after the beginning of phase 3, automated, non-interactive traffic continues to account for a significant portion of login events.

While there was a marked decrease in external, non-MFA SSH traffic, particularly at the beginning of phase 2, user support queries did not increase as much we expected, shown in Figure 5. From August to the end of the year, MFA-related user support tickets comprised an average of 6.7% of all inquiries. During January to March of 2017, MFA inquiries averaged only 2.7% of ticket inquiries.

The number of newly initialized MFA device pairings generated per day is given in Figure 6. Increases in the number of device pairings can be correlated to the initial announcement on August 10th and also to the days immediately following transitions from one

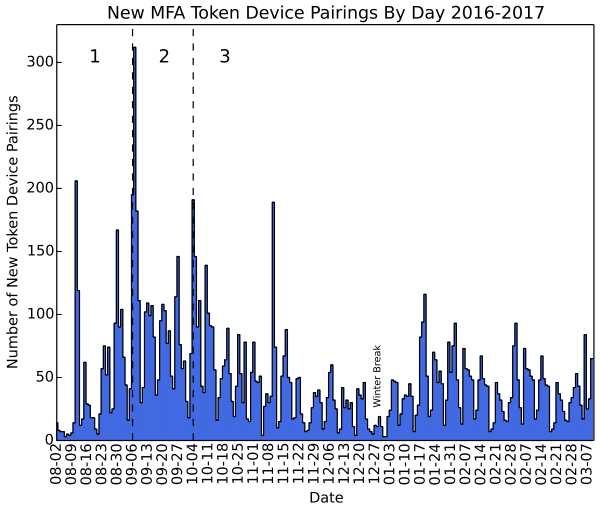


**Figure 6: Number of new token pairings broken down by day. Several of the increases directly correlate with mass announcements (08-10) or a change in the MFA pairing policy as described in Section 3.4 (09-06, 10-04). A majority of users chose to pair an MFA device before the mandatory phase 3 started.**

**Table 1: Percentage breakdown of current token device pairing types. More than 95% of users tend to utilize a mobile device with either the customized, open source mobile application, i.e. "soft token", or via SMS text message with the "SMS token".**

| Token Device Pairing Type | Breakdown (%) |
|---|---|
| Soft | 55.38 |
| SMS | 40.22 |
| Training | 2.97 |
| Hard | 1.43 |

phase to another. October 4th, the day MFA was made mandatory, ranks fourth in the total count of newly initialized pairings while September 7th, the day after phase 2 began, ranks first. New device pairings slowly declined until the end of the year. Beginning with the Spring semester, new pairings once again increased and have shown a slight declining trend since.

The current percent makeup of token device pairing types is given in Table 1. The far majority of users depend upon the smartphone-based application or SMS text messaging service to receive their token codes. Little to no issues have been reported for any of the device pairing types once users have successfully paired. In a handful of cases, an SMS text message will arrive delayed. Logs indicate that the user's network carrier had failed to deliver the message until subsequent retries delivered the token code in an expired state.

## 6 CONCLUSIONS

We describe our experience deploying a multi-factor authentication infrastructure throughout a large number of HPC production resources serving an eclectic research base, from individual users accessing via SSH to portal accounts interacting through science gateways. To be successful, the MFA implementation needed to be a cost-effective solution, integrating harmoniously with existing production HPC services, and providing the right balance of control, flexibility, and security to meet both the needs of the center and those of the user community. Throughout the planning, development, and rollout of MFA, a few key insights we learned are highlighted in the following paragraphs.

A transition to MFA during the production lifetime of an HPC system requires a level of control and deployment flexibility not available or not available at a reasonable cost from commercial vendors. Providers of MFA services with a cost model based on the active number of users per month is intractable at the scale we require. Our approach was to supplement existing, open source components to provide the MFA solution that was right for us.

Accommodating the needs of individual researchers, community, and gateway accounts prompted the development of a powerful MFA access control list mechanism. Researchers who needed extra time and assistance to prepare for MFA were easily granted temporary exemptions while trusted hosts and specialized accounts were whitelisted to continue important automated services, uninterrupted by MFA or its deployment, at the behest of thousands of satellite users. This in-house tool provides dynamic, scalable, and agile configuration options not available in other implementations.

User managed opt-in capabilities during a phased transition allow researchers to prepare and rigorously test complex workflows for a new MFA log in paradigm in their own time instead of being subject to coarse enforcement mechanisms mandated on system administrators' schedules with no recourse for incompatibilities. Assessments of HPC resource access patterns enabled staff to target user groups that were believed to be particularly in need of this transitional functionality. Controlling when to pair and with what device over a window of time empowered staff and users alike to preemptively strategize and communicate knowledge on how best to integrate MFA into a typical HPC workday.

With over half a million successful log ins and counting, this paper details the collection of software and hardware components integrated together to provide a scalable, production-ready, multi-factor authentication solution built for HPC systems with broad user communities. This software infrastructure is freely available for open source distribution and is ready to be grown to incorporate new features including geolocation services, dynamic risk assessment, or biometric security.

## ACKNOWLEDGMENTS

The authors gratefully acknowledge National Science Foundation support under award number ACI-1134872 (Stampede). We would like to thank our colleagues Carrie Arnold and Walter Scarborough for their help in preparing our smartphone-based applications for Android and iOS platforms. The authors would also like to mention our colleagues R. Todd Evans, Antia Lamas-Linares, and Carlos Rosales-Fernandez for their comments and suggestions for this

manuscript. Next, we wish to thank Chris Hempel and the User Services group for their for invaluable feedback on user perceptions and roll out. Finally, we wish to acknowledge all TACC staff members for their assistance in completing this milestone.

## REFERENCES


[1] FIDO Alliance. 2017. FIDO Alliance. (2017). https://fidoalliance.org/about/overview

[2] William E. Burr, Donna F. Dodson, Elaine M. Newton, W. Timothy Perlner, Ray A.and Polk, Sarbari Gupta, and Emad A. Nabbus. 2017. Electronic Authentication Guideline. Technical Report NIST SP 800-63-2. Information Technology Laboratory, National Institute of Standards and Technology.

[3] Michal Choras, Rafal Kozik, Andrew Churchill, Artsiom Yautsiukhin, and Ben Brewster. 2016. Are We Doing All the Right Things to Counter Cybercrime? Springer International Publishing, Cham, 279–294. DOI: https://doi.org/10.1007/978-3-319-38930-1_15

[4] Abdullahi Chowdhury. 2016. Recent Cyber Security Attacks and Their Mitigation Approaches – An Overview. Springer Singapore, Singapore, 54–65. DOI: https://doi.org/10.1007/978-981-10-2741-3_5

[5] OpenSSH Community. 2017. SSH Multiplexing. (2017). https://en.wikibooks.org/wiki/OpenSSH/Cookbook/Multiplexing

[6] Feitian Technologies Company. 2017. Feitian. (2017). https://ftsafe.com

[7] Duo. 2017. Duo Security. (2017). https://duo.com

[8] MariaDB Foundation. 2017. MariaDB. (2017). https://mariadb.org

[9] The Apache Software Foundation. 2017. Apache. (2017). https://www.apache.org

[10] Iterate GmbH. 2017. Cyberduck. (2017). https://cyberduck.io

[11] KeyIdentity GmbH. 2017. LinOTP. (2017). https://www.linotp.org

[12] Alphabet Inc. 2017. Google Authenticator. (2017). https://github.com/google/google-authenticator

[13] Liferay Inc. 2017. Liferay. (2017). https://www.liferay.com

[14] Twilio Inc. 2017. Twilio. (2017). https://www.twilio.com

[15] Bitvise Limited. 2017. Bitvise. (2017). https://www.bitvise.com/ssh-client-download

[16] Linux-PAM. 2017. Linux-PAM. (2017). http://www.linux-pam.org

[17] InCommon LLC. 2017. InCommon. (2017). https://www.incommon.org

[18] RSA Security LLC. 2017. RSA Security. (2017). https://www.rsa.com

[19] David M'Raihi, S Machani, M Pei, and J Rydell. 2011. Totp: Time-based one-time password algorithm. RFC 6238. RFC Editor. https://tools.ietf.org/html/rfc6238

[20] Oracle. 2017. Oracle Solaris Administration: Security Services. (2017). http://docs.oracle.com/cd/E23824_01/html/821-1456/pam-1.html

[21] W. Cyrus Proctor. 2017. OpenMFA: TACC OpenSSH pluggable authentication module project for multi-factor authentication. (2017). DOI: https://doi.org/10.5281/zenodo.841211

[22] FileZille Project. 2017. FileZilla. (2017). https://filezilla-project.org

[23] The FreeRADIUS Server Project and Contributors. 2017. The FreeRADIUS Project. (2017). http://freeradius.org

[24] Martin Přikryl. 2017. WinSCP. (2017). https://winscp.net/eng/download.php

[25] Eric Rescorla. 2000. Http over tls. (2000).

[26] Vipin Samar. 1996. Unified Login with Pluggable Authentication Modules (PAM). In Proceedings of the 3rd ACM Conference on Computer and Communications Security (CCS '96). ACM, New York, NY, USA, 1–10. DOI: https://doi.org/10.1145/238168.238177

[27] Jim Sermersheim. 2006. Lightweight directory access protocol (LDAP): The protocol. (2006).

[28] Yogendra Shah, Vinod Choyi, and Lakshmi Subramanian. 2015. Multi-factor Authentication as a Service. In 2015 3rd IEEE International Conference on Mobile Cloud Computing, Services, and Engineering. IEEE, IEEE, San Francisco, CA, USA, 144–150.

[29] Dag-Erling Smørgrav. 2017. Pluggable Authentication Modules. (2017). https://www.freebsd.org/doc/en/articles/pam

[30] Simon Tatham. 2017. PuTTY. (2017). http://www.chiark.greenend.org.uk/~sgtatham/putty/latest.html

[31] M. Theofanos, S. Garfinkel, and Y. Y. Choong. 2016. Secure and Usable Enterprise Authentication: Lessons from the Field. IEEE Security Privacy 14, 5 (Sept 2016), 14–21. DOI: https://doi.org/10.1109/MSP.2016.96

[32] Sheldon Whitehouse, Michael T. McCaul, Karen Evans, and Sameer Bhalotra. 2017. From Awareness to Action: A Cybersecurity Agenda for the 45th President. Technical Report. Center for Strategic & International Studies.

[33] Steve Willens, Allan C Rubens, Carl Rigney, and William Allen Simpson. 2000. Remote authentication dial in user service (RADIUS). (2000).

[34] Tatu Ylonen and Chris Lonvick. 2006. The secure shell (SSH) transport layer protocol. (2006).

[35] Yubico. 2017. Yubico. (2017). https://www.yubico.com