Cipherise Technical Brief

Version 4.0 - September 2021

Table of Contents

Audience

This brief provides IT and security professionals an overview of a typical Cipherise installation, outlining each key feature of the system, and how all these features work together to provide cryptographically secure, convenient authentication.

What is Cipherise?

Cipherise is a revolutionary product that achieves what was previously thought impossible in the world of technology, by addressing the competing objectives of increasing the level of security on a platform and removing complexity for end users, without compromising on either. This was the challenge that we set ourselves and which Cipherise delivers.

Cipherise From The User Perspective

The Cipherise experience for the end user is centered around the Cipherise mobile application. The app makes the experience of logging in to a system extremely simple.

To commence an authentication process, using the Cipherise app on their mobile phone, the user simply scans a QR code on their computer screen (usually presented on the log in page). This triggers an out of band request from the app to the Cipherise Server to authenticate the user.

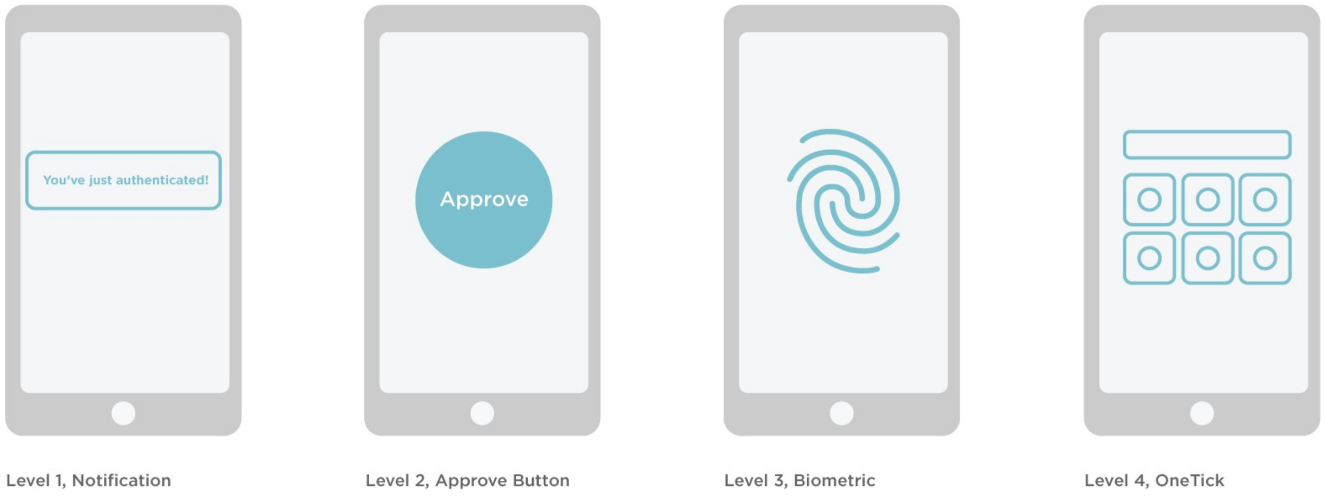
Depending on the risk profile of the system the user is attempting to authenticate to, the Cipherise Server will present the user with one of four different challenge types on their mobile phone (Refer Figure 1), such as a request for a biometric.

If the user completes the challenge correctly, the Cipherise Server will provide positive verification to the application server and the user will be presented with an authenticated session on their computer*.

The whole process is complete in about 3 seconds. No username or password is typed into a keyboard and no user credentials are ever transported or shared.

* Note: It is possible to initiate an authentication or approval process using triggers other than the scanning of a QR code. For example, Cipherise can also provide authentication for mobile apps using a "push button" trigger instead of scanning a QRcode.

The description above demonstrates how we have achieved the vast improvement in usability, but obviously, under the covers there is a lot more technical detail. To understand how we have also met our goal to significantly improve security, please read on.



Cipherise Architecture

How does Cipherise work?

Cipherise offers a new and unique solution to the many existing problems around authentication. It offers the convenience of a single simple secret for an end user to access a range of registered services, yet also enforces the highest levels of Multi-Factor Authentication in a way that doesn't negatively affect the user experience. In fact, the level of friction for a user at the point of logging in to a service is decreased as the user no longer has to type a username and password.

One of the key conceptual differences between Cipherise and existing authentication mechanisms is that the authentication process is decentralised, in that Cipherise moves the authentication process from a centralised server to a user's mobile phone.

The impact of this change is enormous. As there is no longer a need for a user's password to be matched against a centrally stored copy of that password. Organisations no longer need to maintain a database of usernames and passwords. This change significantly reduces an organisation' s risk exposure to identity theft, as there is no possibility of a security breach through typical password based attack vectors

A key component of the Cipherise Platform is the Cipherise Server. The Cipherise Server acts as a messaging broker between the Cipherise Mobile Application and the system which the end user is attempting to log in to.

All communication between the mobile phone and the Cipherise Server is underpinned by Public Key Infrastructure (PKI) with RSA encryption. This process allows the Cipherise Server to confirm that it is communicating with an authorised device.

Each time a user enrolls to use Cipherise on a new system a new set of Public/ Private Keys is created for that specific system on the user's mobile phone. Four pairs of keys are created, one for each of the different challenge types.

After the key pairs are created for a new system, the private key remains on the user's mobile phone (encrypted and stored in the hardware protected enclave) and the public key is shared with the Cipherise Server.

The Cipherise Server also determines, based on pre-configured rules, the type of challenge to be issued to the user on their mobile phone when they trigger an authentication process. Four challenge types are employed, refer to Figure 1 above. More detail on these challenges will be provided later in this document.

When a user successfully solves an authentication challenge on their mobile phone, a Private Key specific to that level of challenge, is unlocked from the secure enclave of the mobile, for the specific system that they are attempting to log in to. This key is then used to digitally sign a response to the Cipherise Server.

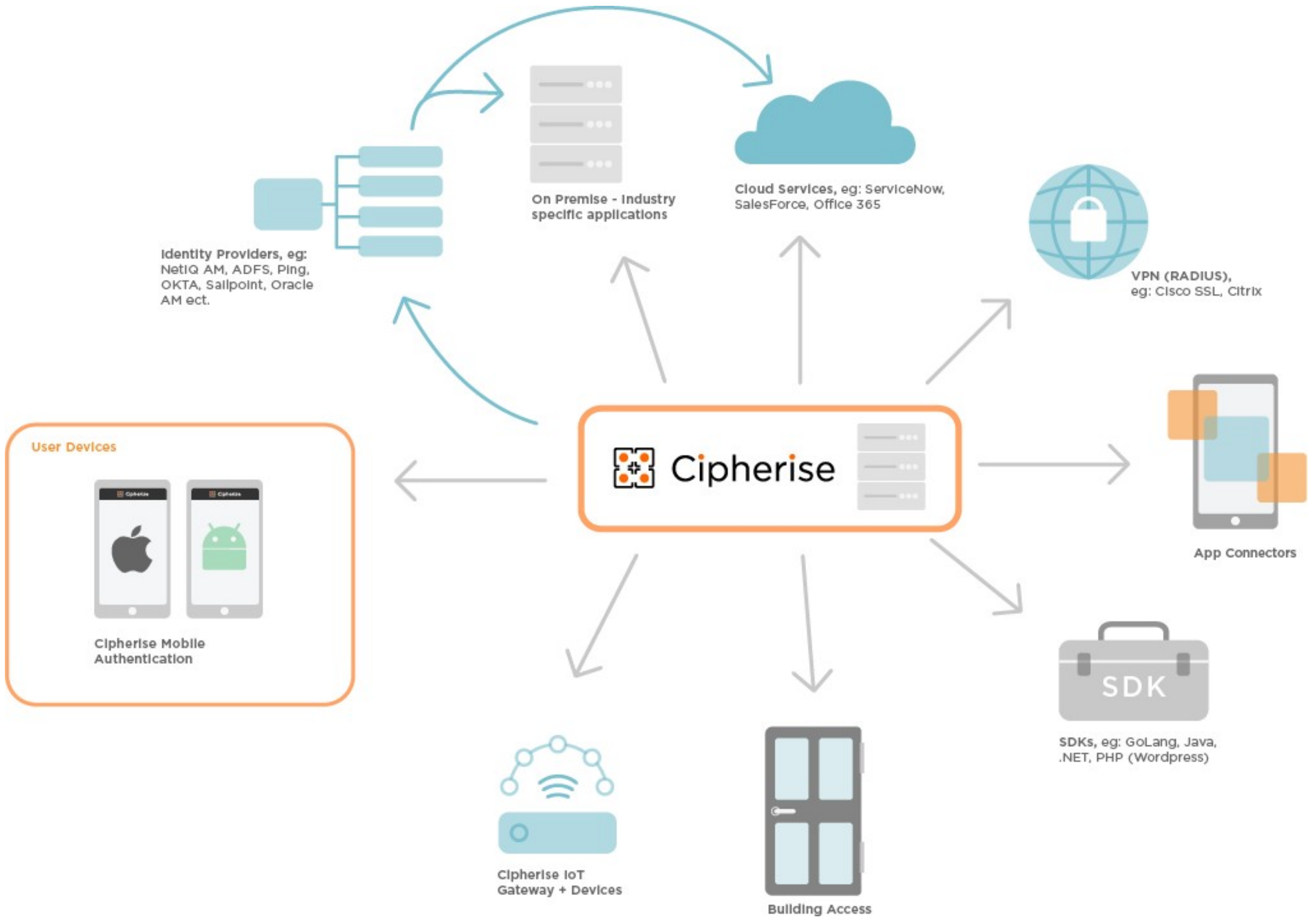
With this process, if the Cipherise Server receives a correctly digitally-signed response to the challenge it issued, both of the following must be true:

1. The response was received from the specific trusted device to which the challenge was issued.
2. The challenge was correctly solved by holder of that trusted device.

Cipherise Components

The Cipherise platform is comprised of three main components:

- **Cipherise Mobile Application**
- **Cipherise Server**
- **Cipherise Integration Server**


Identity Providers, eg: NetIQ AM, ADFS, Ping, OKTA, Sailpoint, Oracle AM ect.
On Premise - Industry specific applications
Cloud Services, eg: ServiceNow, SalesForce, Office 365
VPN (RADIUS), eg: Cisco SSL, Citrix
User Devices
Cipherise Mobile Authentication
Cipherise
App Connectors
SDK
SDKs, eg: GoLang, Java, .NET, PHP (Wordpress)
Cipherise IoT Gateway + Devices
Building Access

Cipherise Mobile Application

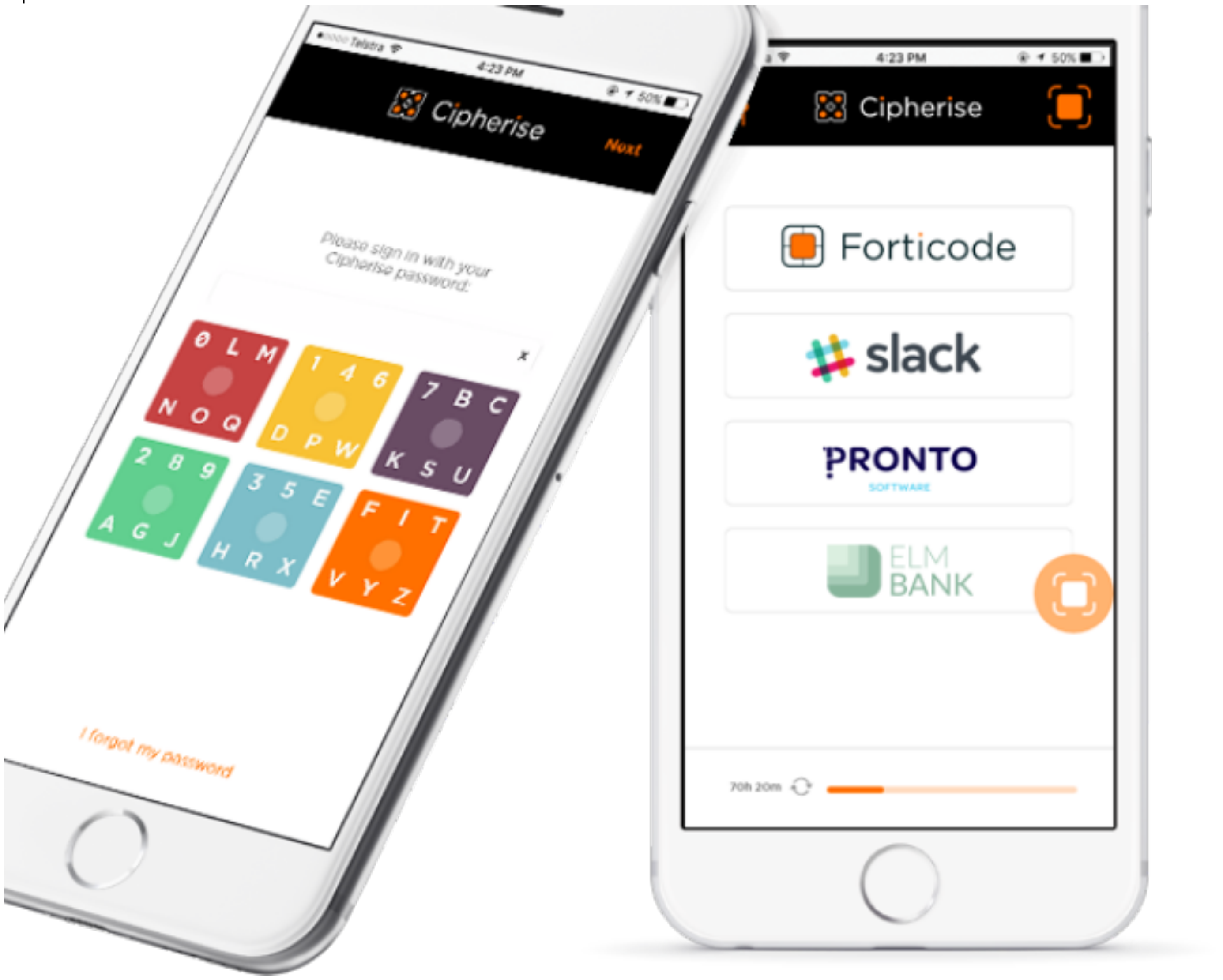
The Cipherise Mobile Application is the core user interface layer of the Cipherise platform. It offers a simple way to securely login to an application or approve an action, places user experience at the forefront and protects the user through almost invisible security parameters.

Available through either Apple's App Store or via Google Play, the Cipherise Mobile Application is designed around a self-service model. Users use a familiar method to install it onto their personal mobile device(s), setup their secret via the One Time Cognitive Keyboard (OneTiCK) method and can immediately enrol to Service Providers.

Using Cipherise, a user will never directly type their password into a screen, therefore they are completely protected from forms of endpoint (or password) attack such as key-loggers, brute force, observation, recording, phishing, and man-in-the-middle attacks. And due to the protection provided by OneTiCK, a user can use a simple and memorable secret to access all Digital and Physical Application services without any reduction in their personal security. (Please refer to later in this document for a deeper description of OneTiCK).

The Cipherise Mobile Application acts as an aggregator or provides a federation service, for multiple services in much the same way as an Identity Provider but manages these multiple forms of access from the user's mobile phone. Each enrolled service is underpinned by four private keys (one for each authentication level) which are unlocked upon successfully solving a challenge and exchanged with the Service Provider under the industry standard PKI framework and RSA cryptography.

The Cipherise Mobile capability is also available as an OEM SDK for Android and iOS mobile operating systems allowing a custom build and experience.



Cipherise Server

The Cipherise Server is the core engine within the authentication platform. It manages the routing of authentication requests originated from client applications and servers, via the Connectors, to the registered Cipherise Mobile App layer on an identified user's mobile phone. The Cipherise Server manages three core processes:

1. **Enrolment** – the process of 'binding' a user's locally managed identity(managed within the Cipherise Mobile App) to a Service.
2. **Authentication / Validated Intent** – the process of receiving a request from a Service for an identity assertion, and the subsequent fulfilment of this request via the Cipherise Mobile App back to the Service.
3. **Revocation** – the process of removing (or suspending) the binding between a user and their registered Services. This can be triggered via the Cipherise Dashboard (e.g. by an Administrator) or by a user through self-service within the Cipherise Mobile App.

The Cipherise Server interacts directly with the Cipherise Mobile App, and via the Cipherise Connector(s) to various systems, platforms and services to which users need to authenticate. An implementation can also contain multiple Cipherise Servers for load balancing and redundancy, and these can be distributed geographically to ensure network lag is reduced.

Cipherise employs Public Key Infrastructure (PKI) to securely manage connectivity between the Connectors and the Cipherise Mobile App. The Cipherise Server itself is not an explicitly trusted component of the system – all authentication communication is independently verifiable using public / private key pairs, and thus there is no opportunity for a malicious actor to subvert Cipherise's authentication process even via a network level intercept.

The Cipherise Server is essentially stateless in that it retains no session information, or ongoing persistence for an authentication process. It only manages a user identifier and the relationship of one (or many) Cipherise Mobile Apps. No user credentials or private information is ever held within the Cipherise Server, and thus there is no risk of personally identifiable information being mined from this server.

The Cipherise Server also exposes an API that is instantiated via a suite of Software Development Kits (typically used to develop custom connectors). It also provides database independence and runs across various versions of the Linux OS family. For ease of deployment and management, the Cipherise Server is deployed through docker containers into the Forticode's cloud infrastructure.

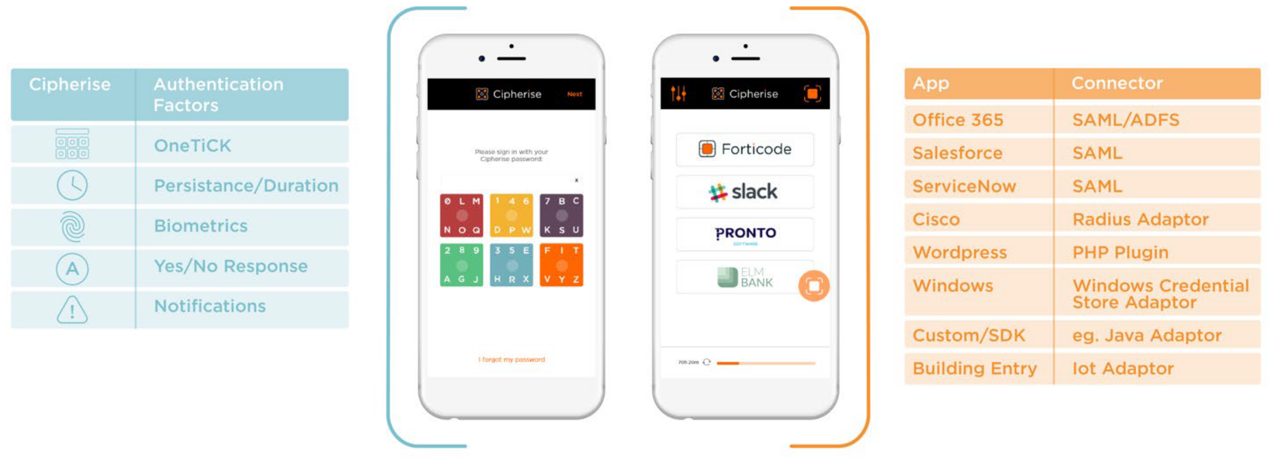
Cipherise Integration Server

Cipherise can be implemented by employing Restful API or SDKs. However, doing so would consume development time and may delay deliveries. Cipherise Integration Server (CIS) is designed to address this issue.

The Cipherise Integration Server fully implemented all Cipherise features out of the box. It also has integrations with most mainstream implementations out there. This makes CIS ideal to be configured with existing implementations in order to benefit from Cipherise's security and convenience without spending many development hours. A typical integration with known services usually takes under 10 (ten) minutes to configure. Others might take a bit longer, however it is still much quicker than developing from scratch. Additionally, CIS **audits** all activities and is available for analysis on the fly. It can also be queried externally to be fed into another system for further analysis and alert.

Moreover, CIS is supported by Forticode and new features and integrations are consistently being introduced. If new integration methods are required or unlisted in this document, please send a feature request to Forticode support.

Note - sometime we call these integration methods as connectors.



Cipherise	Authentication Factors
	OneTiCK
	Persistance/Duration
	Biometrics
	Yes/No Response
	Notifications

App	Connector
Office 365	SAML/ADFS
Salesforce	SAML
ServiceNow	SAML
Cisco	Radius Adaptor
Wordpress	PHP Plugin
Windows	Windows Credential Store Adaptor
Custom/SDK	eg. Java Adaptor
Building Entry	Iot Adaptor

Sample integration methods we support includes:

Credential Store

In general, enterprise implementations usually utilises credential store to keep users records. CIS consumes those records to be used for enrolment and authentication to the Cipherise platform. CIS is also able to query live records to validate authentication processes. CIS also support mapping of fields in the credential store to be used as attributes.

This then allows the maintenance of authorisation and roles to be administered from the existing credential store that the administrators are familiar with.

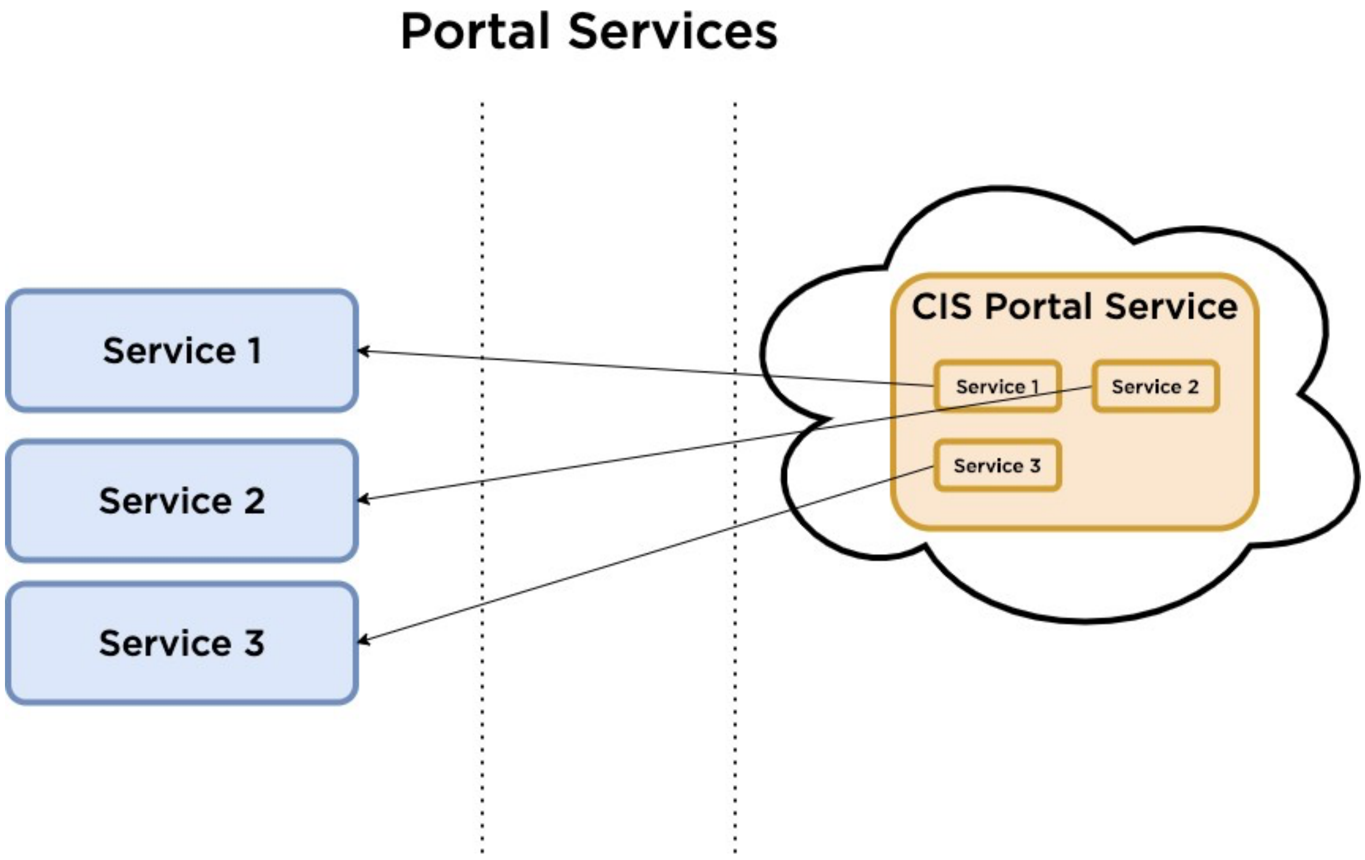
Administrator's access to CIS can be configured to load from a credential store. Administrators can then add credential stores and assign them to different services. Essentially CIS supports multiple credential stores to be configured at the same time.

The following credential stores are supported and tested in CIS:

- Local Store (CIS stores user records)
- LDAP(S)
    - AD
    - DS389
    - Azure
    - Oracle Identity Manager

Portal

CIS supports **portal functionality to allow all available applications for a user be in one place for easy access**.

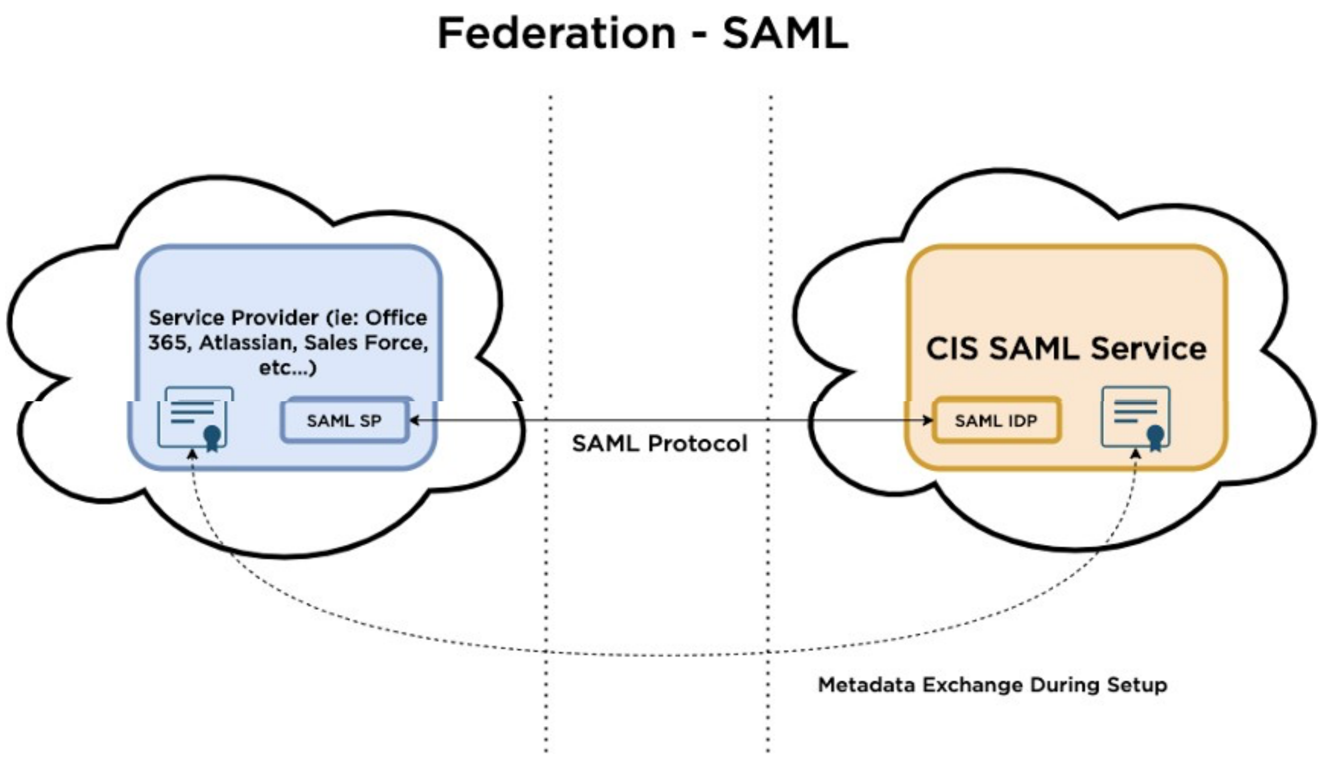


Federation Methods

CIS supports **SAML2.0** as an Identity Provider (iDP). It will integrate with any application that implements SAML 2.0 as a Service Provider (SP).

Some of the popular SAML2.0 applications that was tested to be working with CIS is as follow:

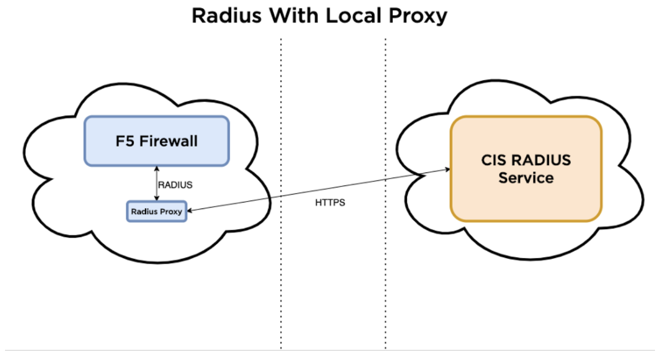
- AWS
- Microsoft 365
- Sales Force
- Atlassian
- Slack
- GitLab
- F5
- Shibboleth
- VMware Horizon
- Thycotic Secret Server
- Skills Base
- Wordpress (via SAML 2.0 Plugin like Mini Orange)

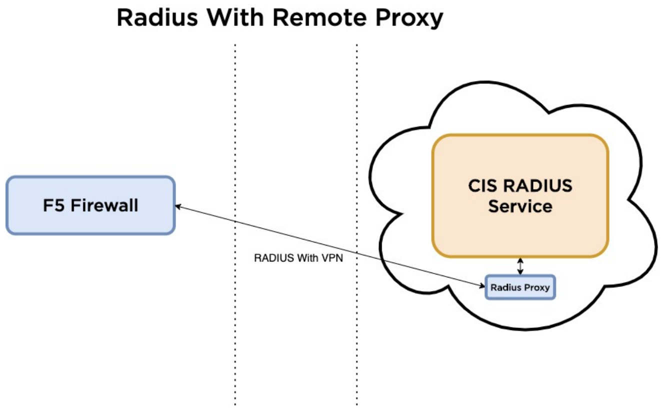

Federation - SAML
Service Provider (ie: Office 365, Atlassian, Sales Force, etc...)
SAML SP
SAML Protocol
CIS SAML Service
SAML IDP
Metadata Exchange During Setup

CIS also support **OpenID Connect** as an Identity Provider (iDP).

RADIUS

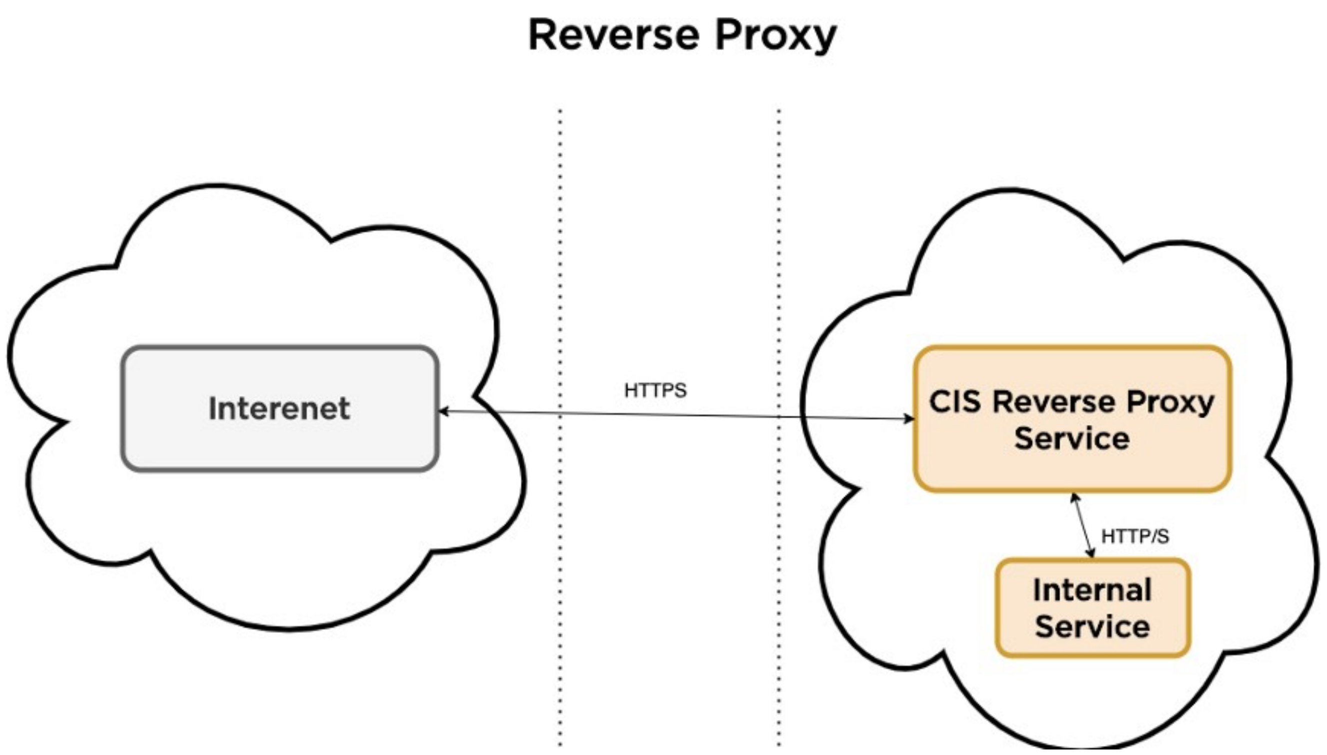
The Cipherise Integration Server Radius proxy allows Cipherise authentication through any **RADIUS** client. Moreover the Proxy that runs on the same network as the RADIUS client talks to CIS using HTTPS allowing deployment to be done on separate network, yet keep security intact.





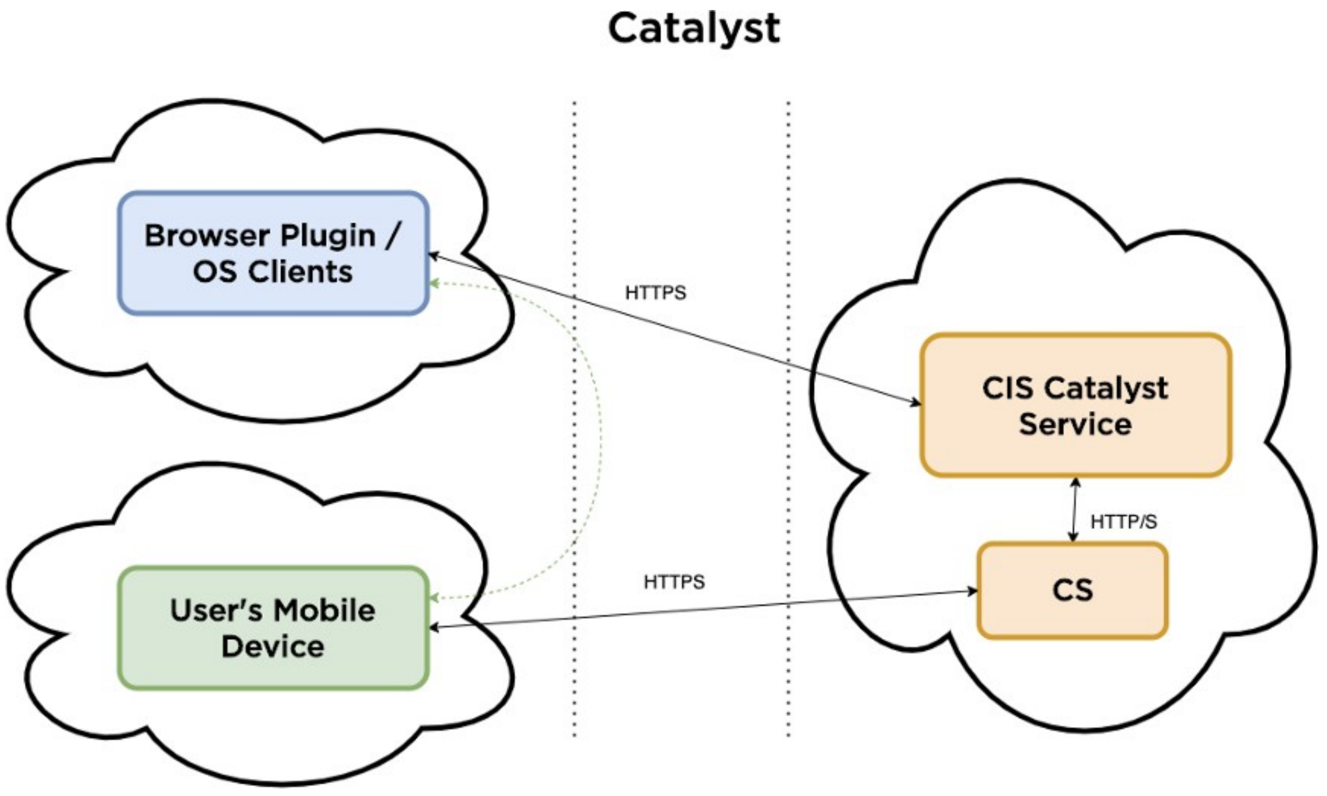
Reverse Proxy

The Cipherise Integration Server supports **reverse proxy** functionality allowing internally hosted services available externally and be protected with Cipherise. This negates the need of Virtual Private Network (VPN) to access internal service yet still be secured by the Cipherise platform.



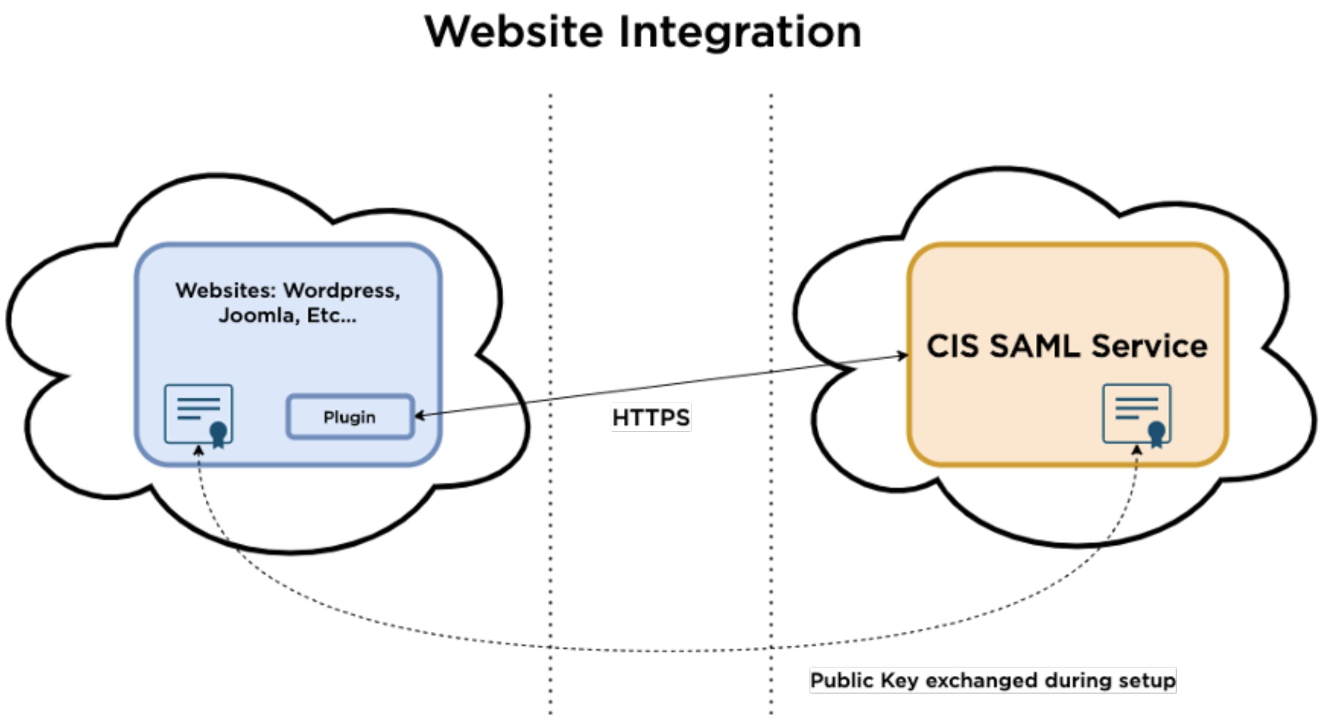
Catalyst

The Cipherise Integration Server **Catalyst plugin allows password plug capabilities** to allow logins to operating systems and websites. User credentials are being saved in user mobile applications and sent back as a payload encrypted and utilised at the end point.



Website Integration

Cipherise Integration Server allows **website integration just by applying few lines of code**. A plugin utilising these code is available for **WordPress** and soon be available for other content management system.


Website Integration
Websites: Wordpress, Joomla, Etc...
Plugin
HTTPS
CIS SAML Service
Public Key exchanged during setup

Single Sign On

Cipherise Integration Server keeps track of session and manage them. It then leverages the Federation Methods to authenticate users into the service. Administrators are able to configure behaviours as how users gain access to the application under **Single Sign On** depending on risk factor.

Cipherise Security

A core design goal with Cipherise is to offer best in class security, but at the same time ensure the user experience is incredibly simple and seamless. There is no point having yet another solution that addresses the security dimension but fails to understand that people just won't accept extra steps when trying to login. The Cipherise security model is comprised of the following logical elements that are facilitated through the platform architecture and authentication process:

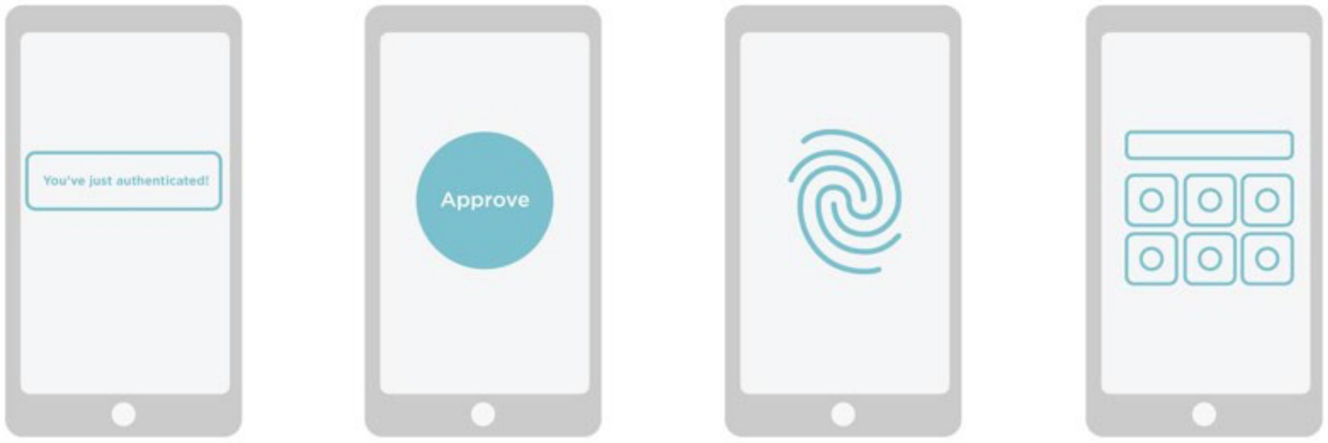
1. Authentication Challenge Levels
2. One Time Cognitive Keyboard
3. Public Key Infrastructure
4. Decentralisation
5. Device Specific Security
6. Persistence
7. Notifications
8. Contextual Messaging
9. Advanced OneTiCK Method

Authentication Challenge Levels

Cipherise supports the concept of adaptive authentication through aligning the type of user interaction required based upon the risk profile of the service that they are attempting to access.

Four levels of user 'challenge' are supported, and these are presented within the Cipherise Mobile Application as part of an authentication request. Solving the presented challenge unlocks the private key for that level associated with the requesting service to be used to sign the subsequent identity assertion.

It is important to note that ALL authentication challenge levels require the completion of the OneTiCK challenge in advance, to maintain "persistence" (described below) within the Cipherise Mobile Application, and any additional verification factors are 'layered' on top of this initial validation of 'something you know'.



Level 1, Notification   Level 2, Approve Button   Level 3, Biometric   Level 4, OneTick

The four levels of user challenge are:

L1. Notification

No user action required, but the user receives a notification that an assertion has been made –based upon the something you know factor you provided periodically. Typically only used for a very low risk service. This Notification ensures that even in processes where no specification is required to authenticate, the user is notified.

L2. Push

The user is presented with a Push button within the Cipherise Mobile App. Pressing this button indicates possession and validates the something you have factor and adds to the something you know factor. Typically only used for a low risk service.

L3. Biometric

The user is presented with a biometric challenge (e.g. fingerprint, face scan) within the Cipherise Mobile App. Satisfying this challenge validates both the something you have factor and the something you are factor, as well as the something you know factor provided periodically based on security posture and risk.

L4. OneTiCK

The user will be asked to enter their secret, using OneTiCK. If completed successfully, this validates the something you have factor, and actively enforces an immediate re-proving of the something you know factor. This challenge level is recommended for higher risk services or where the highest level of confirmation regarding a user's identity and intent is required.

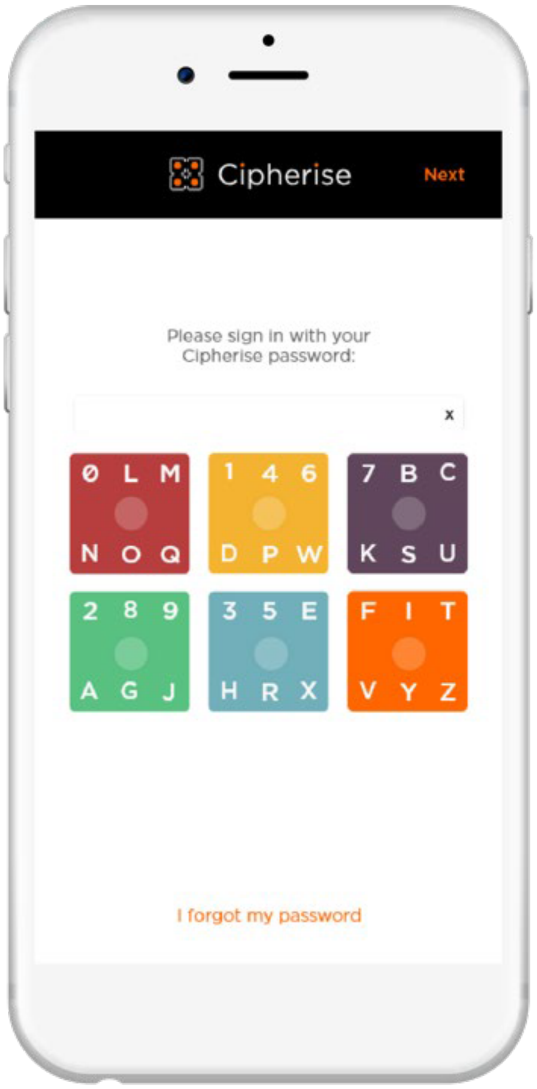
One Time Cognitive Keyboard (OneTiCK)

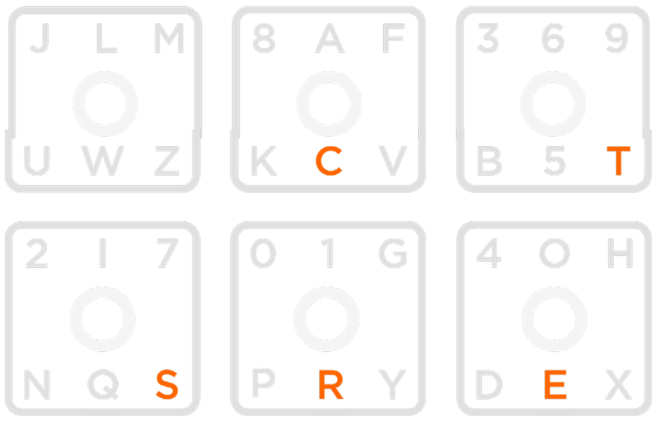
The One Time Cognitive Keyboard is a core element within the Cipherise security model. It is based upon a globally patented abstraction method that means a user can authenticate without ever having to directly type their secret.

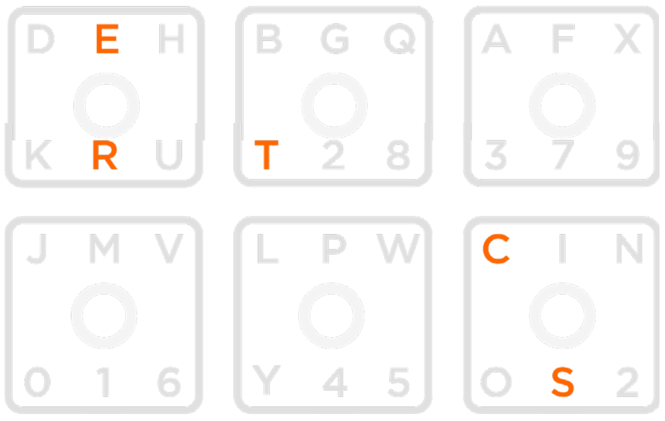
The user can also be assured that whatever they enter cannot be reverse engineered to reveal their secret. A key strength of OneTiCK is that it natively protects against a range of traditional password entry exploits, including record and replay (e.g. key-loggers), brute force, observation, and man-in the middle attacks.

Every time a OneTiCK challenge is presented on a user's mobile phone the characters presented on the six coloured buttons are randomised. Due to this, the user action of entering in their secret will result in a different pattern of key presses every time.

Using the example below and based upon the user's secret being the word "SECRET", the two response patterns are illustrated. Even though the secret stays the same, every interaction is different.







Through the abstraction provided by OneTiCK, and in combination with the other security elements of the solution, Cipherise enables users to have a simple secret with as little as 5 characters. The need for ridiculous password complexity rules such as length greater than 8 characters, inclusion of uppercase, numbers or special characters, and frequent password changes, no longer exists, because Cipherise protects the user at the time of entering their secret and never transmits it anywhere.

Persistence

As mentioned above, all 4 user authentication levels are dependent upon an initial solving of the OneTiCK challenge to maintain persistence. The specific user authentications outline above will not be successful without OneTiCK proving that the user knows their secret within the Cipherise platform (i.e. their single simple password).

Cipherise employs a concept of persistence within the Cipherise Mobile Application.

Every time the Cipherise Mobile App is opened, it first checks to see whether the OneTiCK persistence is valid. This is a user defined period that is designed to improve usability, and to ensure every subsequent user challenge is layered on top of an original validation of the something you know factor, fulfilled by entering their secret into OneTiCK and thus solving the indeterminate cipher.

If OneTiCK persistence has expired, the user will be asked to re-enter their secret through OneTiCK, and this will then recharge the persistence for the defined period.

Note that OneTiCK persistence does not affect a Level 4 authentication request. For this level, the user's persistence is ignored, and they must revalidate the something you know factor by entering their secret into OneTiCK. It's important to note that the level of intention, and thus the security posture and policy is determined by the Service provider.

Public Key Infrastructure (PKI)

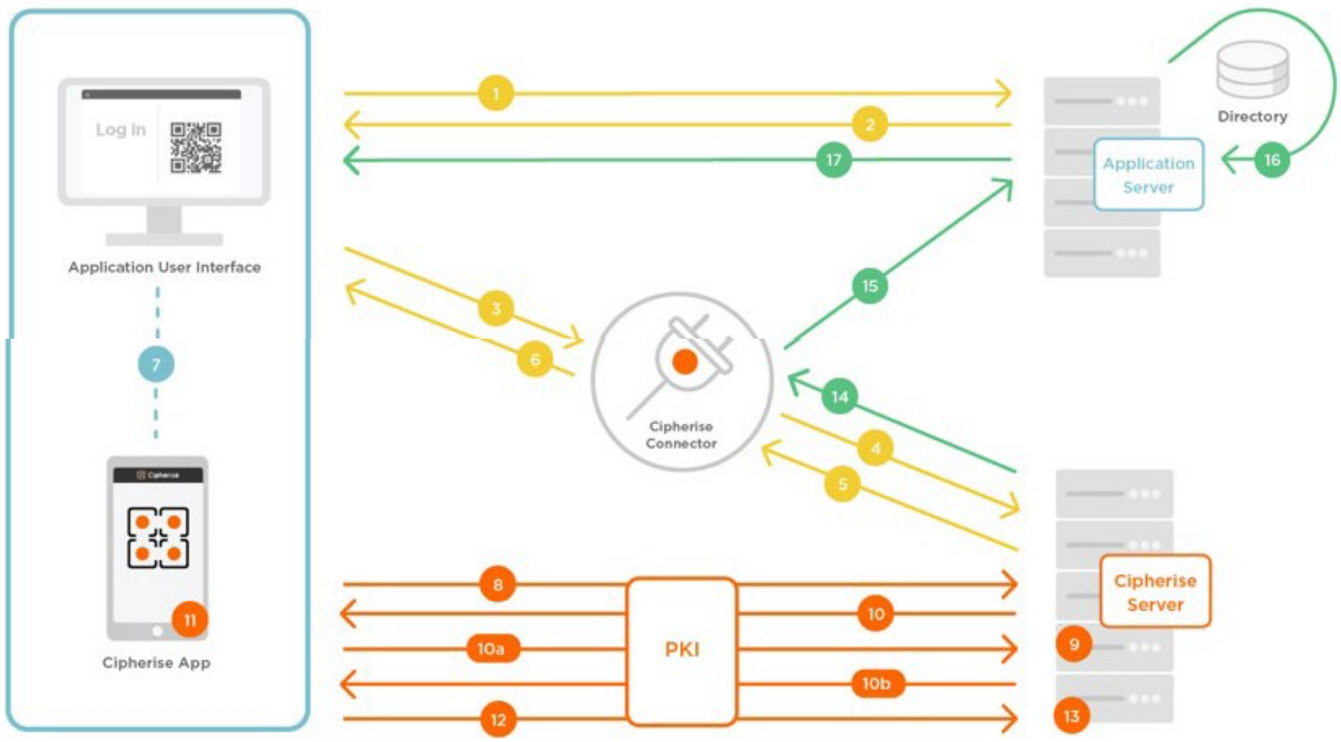
Cipherise uses Public Key Infrastructure (PKI) to build trust relationships between the multiple layers within an authentication transaction.

Each authentication request made from a service to Cipherise requires validation of that Service with a public- private key pair, and every assertion driven by a user's response through the Cipherise Mobile App is also validated across the tiers using a public private key pair.

The Bi-Directional authentication mechanism enables the Cipherise Mobile App to validate if the received authentication request is from the "valid" Service Provider. Bi-Directional authentication is initiated by Service Providers in their authentication ceremony. This way authentication is validated from both the Mobile Application and the Service Provider, in both directions, securing unprecedented levels of trust.

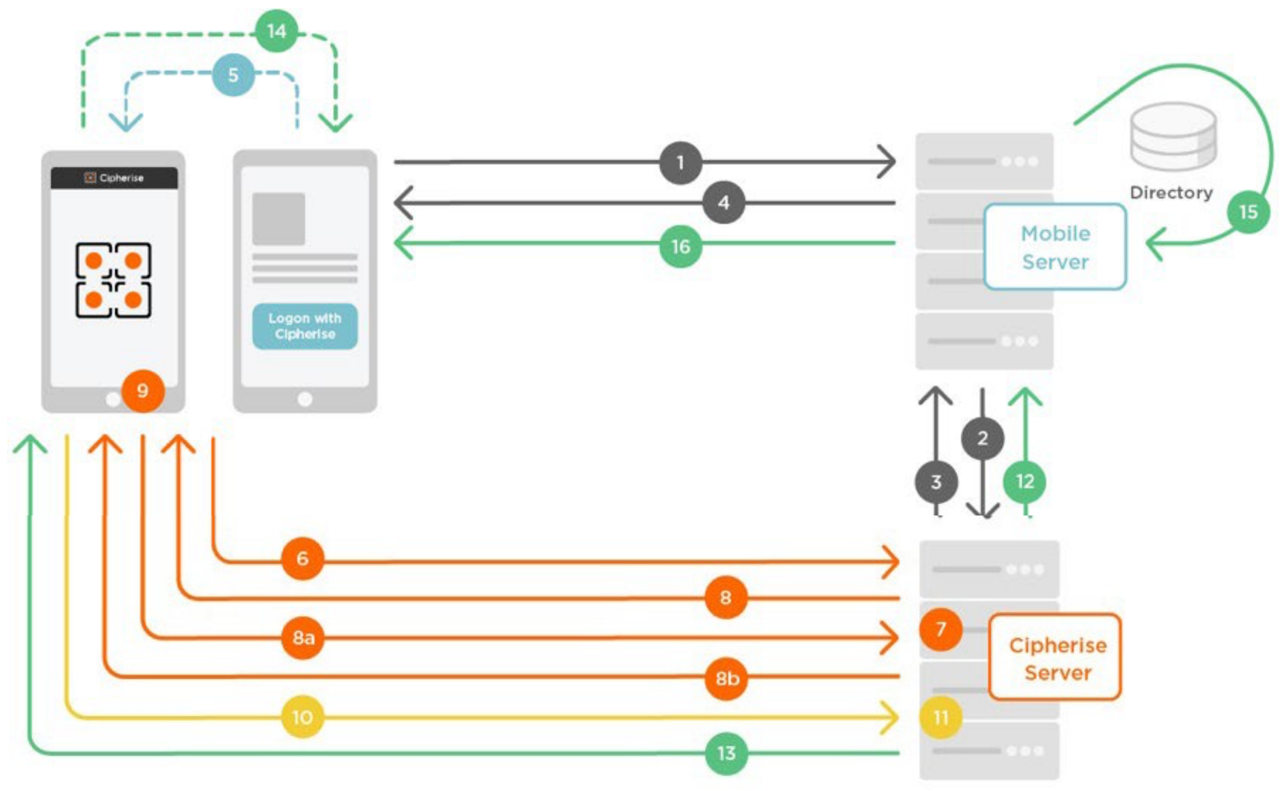
The response assertion's private key is unlocked based upon a user's successful completion of an authentication challenge. By solving the OneTiCK request, as an example, the private key for the requesting service, and the required authentication level, is used to digitally sign the assertion response.

Cipherise Desktop Logic Flow



1. User access the Application platform via opening a webpage on a computer.
2. Application Server returns the Cipherise enabled login page.
3. The Cipherise Login page makes request to CipheriseConnector for QR code.
4. The Cipherise Connector makes a call to the CipheriseServer.
5. The Cipherise Server returns a QR code to the CipheriseConnector – contains a random number (a token).
6. Cipherise Connector passes this QR code back to theApplication Login page as a response to the original request (as a PNG image) – long poll started.
7. User scans QR code with their CipheriseMobile App
8. Cipherise App decodes QR code, and sends request to Server to understand what application is requesting authentication.
9. Cipherise Server takes token, and matches it to theApplication access request – interrogates rule base to understand what authentication level is required.
10. Cipherise Server pushes a signed authentication request to the user's mobile phone – validated by the public key for the Application stored in the phone –and a nonce.
    a. On receipt of authentication request, user's mobile initiates a challenge for Cipherise Server to validate Service Providers identity.
    b. Service Provider solves presented challenge and send it back to Cipherise Server to forward it to user's mobile.Cipherise Mobile Logic Flow
11. User 'solves' the challenge locally on their phone –unlocks the private key for the Application and the authentication level requested.
12. The Cipherise App uses the private key to sign a nonce, and sends this one-time message back to the Cipherise Server along with the enrolled username.
13. Cipherise Server validates the message using theApplication's public key.
14. Cipherise Server sends a successful authentication response message to the Cipherise Connector including the validated username.
15. Cipherise Connector passes this authentication request to the Application Server (protected byTLS).
16. Application matches the supplied username against its user table, and logs the asserted user into the platform.
17. User is logged in with the Application defined role.

Cipherise Mobile Logic Flow



1. On a Mobile open application event, the Mobile App places a request to Cipherise for a token under the API through the Mobile Server
2. On a Mobile open application event, the Mobile Server places a request to Cipherise Server for a token under the API
3. On a Mobile open application event, the Cipherise Server returned token URL to the Mobile Server under the API
4. On a Mobile open application event, the token returned with URL for the locally installed Cipherise App.
5. When token is returned to App, URL facilitates context switch to Cipherise App-Cipherise App is launched.
6. Cipherise App sends token back to Cipherise Server.
7. Cipherise Server validates token as valid, looks up authentication level requirement for the Mobile App(based upon initial registration).
8. Cipherise Server sends signed Nonce to Cipherise App –validated within Cipherise App as being from a known source.
   a. On receipt of authentication request, user's mobile initiates a challenge for Cipherise Server to validate Service Providers identity.
   b. Service Provider solves presented challenge and send it back to Cipherise Server to forward it to user's mobile.
9. On receipt of valid response from Cipherise Server,User solves presented challenge with Cipherise App based upon level issued by Cipherise Server.
10. Upon successfully solving local challenge within Cipherise App, retrieve local PrivateKey for Mobile App for the specific authentication level – sign the Nonce and return to Cipherise Server.
11. Cipherise Server validates response withPublic Key for the Mobile user, the Mobile service and Mobile authentication level.
12. Cipherise Server provides signed Nonce to the Mobile server to create session for the approved username.
13. Cipherise Server sends instruction to Cipherise App to context switch back to Mobile App.
14. Cipherise App 're-opens' Mobile App.
15. Mobile Server validates supplied username against its directory.
16. Mobile App polls the Mobile Server for an active session – once received, the user is logged in.

Decentralised Authentication

One of the most significant differentiators provided by Cipherise with respect to other authentication platforms on the market comes from the fact that all Cipherise authentication transactions are decentralised and completed on a user's mobile phone, completely separate from the service being logged in to.

This is significant for two key reasons:

1. It addresses the challenge that the traditional perimeter can no longer be protected - there is no way to control the physical device or access points used to access a service. It is almost certain that an access request will – at some point – be made from a compromised or infected machine, and thus we must assume every keystroke is being recorded. If any credential information is entered through such a device, it is only a matter of time before an unauthorised user will gain access.
2. There are problems associated with centralised credential storage. Virtually all current authentication platforms rely upon a 'submit and match' process for granting access – a user submits their password, and this is matched against the service's centrally held copy. This is problematic as not only are the repeated password copies creating multiple opportunities for interception, the core credential repository is a significant data-at-rest vulnerability.

There have been increasingly frequent cases of a breach where an attack is based upon gaining a copy of the centralised credential store, taking it offline, and then using brute force techniques to isolate specific user identity information. Typically, these attacks are after privileged user information, however the rest of the database gets dumped, sent to the darknet, and all users are affected.

The decentralised nature of Cipherise removes this risk by never requiring passwords or access credentials to be stored centrally. The actual authentication process is completed locally on a user's mobile phone, and thus there is nothing central to be attacked or breached.

Device Specific Security

Cipherise leverages the current (and future) biometric capabilities of the Apple and various Android based hardware platforms to support the Level 3 authentication challenge. The design future proofs the solution to alternative iterations of the biometric capabilities as these become available from the hardware manufacturers.

It is also important to note that the entire Cipherise authentication process is also protected by the native device capabilities themselves. For example, to compromise a user, you would first have to steal their phone, then get through the protection layer for that phone. You would then have to open Cipherise, solve the OneTiCK challenge, then create an authentication request within a service, and then successfully solve the presented challenge (e.g. fingerprint).

To attempt to achieve this would be an extremely technically challenging and time-consuming process for a hacker. The hack would also need to be completed before a user realised that their mobile phone was missing and had the device revoked from their services.

Due to these factors, the potential payoff for a hacker and the window of time available to take advantage their exploits have been reduced to almost zero.

1. Steal the user's phone
2. Get through the phone's native security
3. Open Cipherise with OneTiCK
4. Initiate an authentication request within a service
5. Validate Cipherise request for biometric
6. Validate response with Cipherise server using public key.

Notifications

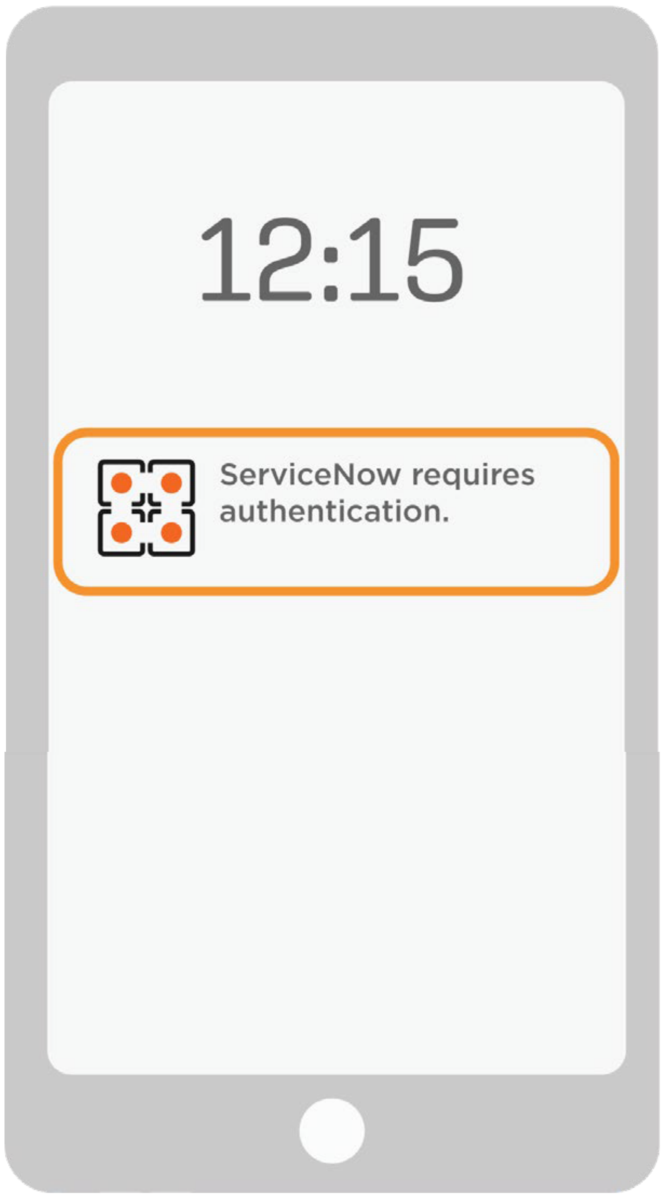
When Cipherise is being employed as the authentication mechanism for a service, it ensures that a user is actively involved every time their identity is being authorised to access the service.

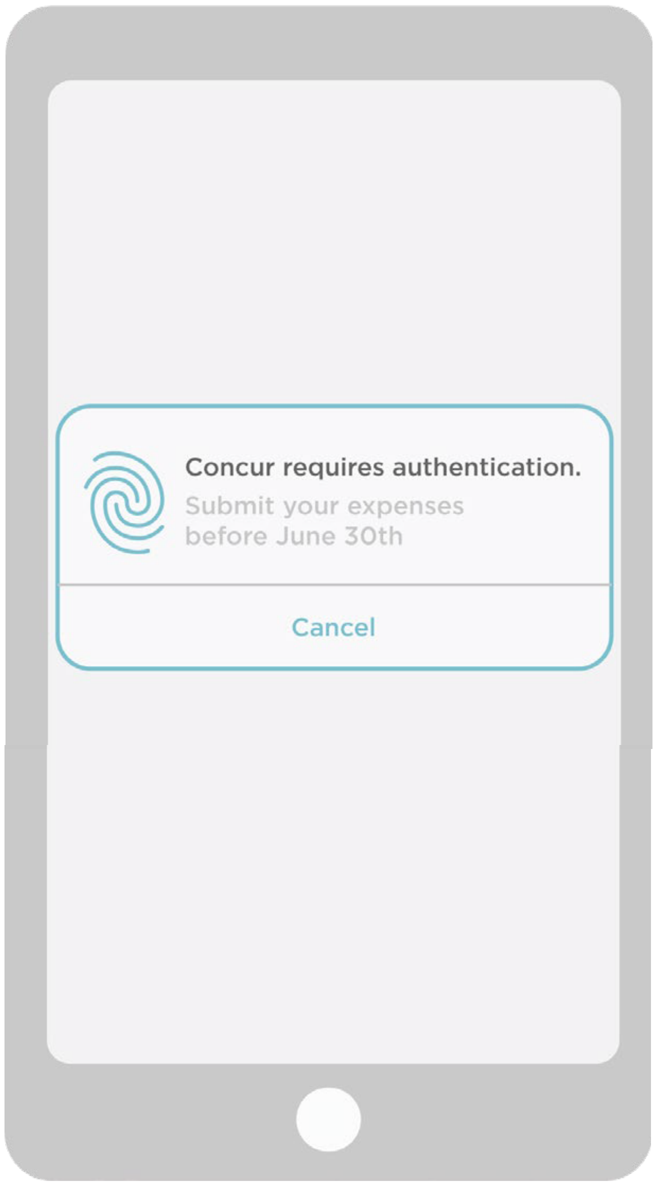
Unlike password based systems where passwords can be shared or stolen, and there is no way to know that your identity has been stolen and used, all authentication processes protected by Cipherise offer an active Notification process.

Notifications provide a real time prompt to the 'owner' of an identity that someone is attempting to login to a system or approve a transaction as them. If a user receives a notification for an action which they did not trigger, that user can immediately block that process and, if configured, post an alert to an external security monitoring platform.

Contextual Messaging

Cipherise creates the unique ability for a Service Provider to provide a targeted contextual message that is relevant to the person authenticating themselves to the service. The real power behind this capability is that the individual has a vested interest in this service and has a very high likelihood of acknowledging the content of the delivered message. The use case for the message content is unlimited but can span areas such as: OH&S, Compliance and Attestation, Training, Service Availability, Public Safety Announcements, Marketing and Advertisements.




Concur requires authentication.
Submit your expenses before June 30th
Cancel

OneTiCK Advanced Methods

Users can elect to choose extended features of the Cipherise system to provide additional protection - by means of alternative OneTiCK Method to employ the best possible protection against observation.

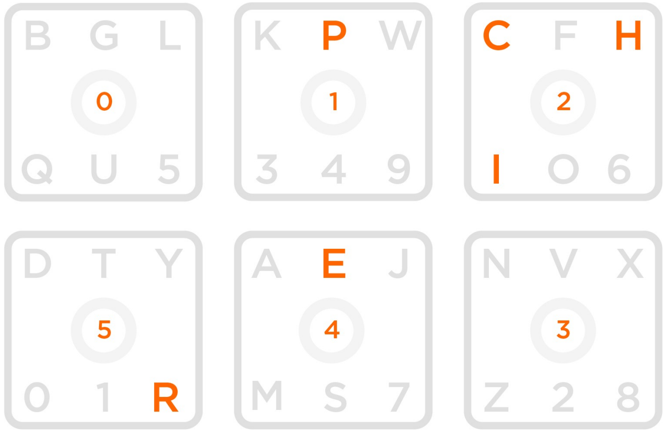
Wildcard Method

The Wildcard Method allows a user to incorporate one or more wildcard characters into their secret. When incorporated, Wildcard characters introduce additional entropy and randomness as there is no correlation between what button is pressed, and the abstracted secret password. The Wildcard allows users to better disguise their secret, affording an observation difficulty similar to a much longer secret, while remaining simple and convenient for the user.

Users who wish to employ this method create the Wildcard when they are choosing their secret. Wildcards are based on the sequence of buttons pressed when entering the secret, so the user must indicate when they will enter the Wildcard in the sequence.

For example, suppose a user wished to apply a Wildcard to the secret CIPHER. To do so, they select the Wildcard input one or more times while creating their secret. If the user wanted to add a single Wildcard between the I and P in CIPHER, they would press the keypads corresponding with C, and I, then the Wildcard character (*), then the keypads corresponding with P, H, E, and R. The newly created secret including the Wildcard is now 7 characters long.

Now, given the OneTiCK keyboard in the example below and the secret CI*PHER, the user would select the following tiles:



1. Find C (tile2). Tap tile 2.
2. Find I (tile 2). Tap tile 2.
3. Wildcard. User can press any tile. For demonstration, select tile 5, tap tile 5.
4. Find P (tile 1). Tap tile 1.
5. Find H (tile 2). Tap tile 2.
6. Find E (tile 4). Tap tile 4.
7. Find R (tile 5). Tap tile 5.

Rotation Method

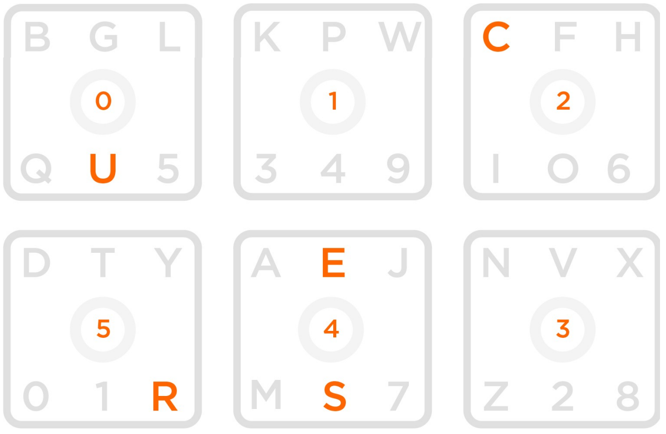
Rotation is an advanced OneTiCK method that allows a single keypress to define a rotation scheme for all subsequent keypresses when entering your secret. The first keypress during the entry of your secret tells Cipherise how many tiles you intend to rotate in a clockwise direction, e.g. Yellow would indicate a clockwise rotation of 1 tile, whilst Purple would indicate a clockwise rotation of 3 tiles.

For example, if you select your rotation key as Yellow or “1”, it would then mean that all subsequent keypresses are to be rotated clockwise by one tile from the correct letter response.

As a specific example, let’s assume that rotation is being applied to the secret SECURE.

Therefore, if the user wanted to rotate clockwise by 1 tile, the input would be:

1. Rotate keys by 1 - tap tile 1.
2. Find S (tile 4). Tap tile 5.
3. Find E (tile 4). Tap tile 5.
4. Find C (tile 2). Tap tile 3.
5. Find U (tile 0). Tap tile 1.
6. Find R (tile 5). Tap tile 0.
7. Find E (tile 4). Tap tile 5.



Cipherise Payload

A central component of data security is protecting personal information from misuse or loss. The Cipherise Payload mechanism enables organisations to protect the personal information they hold from misuse, loss and from unauthorised access, modification or disclosure. Cipherise Payload functionality is a very powerful platform feature enabling Cipherise users an ability to control access to their encrypted personal information, consumed by a specific Service Provider's server. Cipherise users can give temporary access from the Cipherise mobile application by releasing a decryption key to the service provider for a specific piece of information saved in an encrypted format at the Service Provider's server.

The Service Provider will only be able to decrypt and use personal information about an individual, if authorised by the owner of that data. The decrypted information can be in the form of a medical report, a copy of driver's license, driving history report etc.

Payload functionality is an optional add-on which can be activated by a Service Provider. Service Providers enroll users to their systems with optional Payload functionality, in order to consume this functionality requested at the time by the Service Provider and authorized by the Cipherise Mobile application user. Once a user is enrolled with the Payload add-on, the Service provider can initiate a Payload request for that user. The user can elect to approve or decline the request.

The Service Provider presents the user one of four different challenge types on the Cipherise Mobile Application based on the required level of authorisation. The authorisation request can be in the form of a simple notification, an approval button, a biometric or OneTick challenge. If the Cipherise user completes the presented challenge correctly, the Cipherise Server will provide positive verification to the Cipherise Payload Service Provider along with an authorization and approval key to decrypt the stored data at the Service Provider's end for their use. The user can elect to deny access to their stored data by actively declining or allowing the presented challenge to expire.

Cipherise Deployment

Cipherise is designed to be deployed as a sovereign Software as a Service (SaaS) solution in combination with a client side Cipherise Connector to allow integration into the applications and services requiring authentication. Forticode hands the core Cipherise Server virtual cluster to the client to run on their own infrastructure choice. Forticode can assist with provisioning and support as required. Users access the authentication facility through the Cipherise Mobile Application. This is provisioned via a self- service model via either the Apple App Store or Google Play. The Cipherise Integration Server is deployed within client infrastructure, whether this be on-premises, within a private cloud, or even in a client subscribed public cloud instance. It requires connectivity with the Cipherise Server, typically via TLS, therefore firewall configuration changes may be required to facilitate this depending on your hosting arrangements. A trust relationship must also be configured between the Cipherise Integration Server and the Cipherise Server.