initiative for open authentication

OATH Reference Architecture Release 1.0

Initiative for Open AuTHentication (OATH)

CONTENTS

Open AuTHentication Technical Working Group

ActivCard Inc.

Aladdin Knowledge Systems Inc.

Assa Abloy ITG

AudioSmartCard

Authenex

Aventail Corp.

Axalto Inc.

BMC Software Inc.

Boojum Mobile Inc.

Citrix Systems Inc.

Discretix Inc.

Diversinet Corp.

Gemplus Corp.

IBM Corp.

Livo Technologies

LJM Systems

RedCannon Security Inc.

SafeNet Inc.

Signify

Venafi Inc.

VeriSign Inc.

1. Executive Summary

This document specifies version 1.0 of the reference architecture for the Initiative for Open AuTHentication (OATH). The OATH Reference Architecture document describes a high level technical framework for open authentication, as envisioned by the OATH member companies.

This document intends to capture the overall vision for OATH as well as provide a high level technical roadmap. It is targeted towards– decision makers and technical architects from OATH-member and non-member vendor organizations, IT managers and architects from enterprises that are looking to deploy strong authentication solutions, and finally other standards organizations that share all or part of the OATH vision.

The work has been driven by the following key guiding principles:

- **Open and royalty-free specification:** Establish an open and royalty-free specification for strong authentication by leveraging existing open standards when possible, and otherwise leading standardization efforts through well-established technical standard bodies.
- **Device Innovation and Embedding:** Specify technology building blocks that enable low-cost, multi-function authentication devices (e.g., tokens, smart cards) and transform today's mobile devices (e.g., cell phones, PDAs, laptops) into strong authentication devices.
- **Native platform support:** Facilitate native support (platform connectors) for strong device and user authentication in application development and identity management platforms. Leverage existing infrastructure building blocks such as LDAP directories and AAA servers.
- **Interoperable Modules:** Finally, enable best-of-breed solutions through a framework of interoperable components. An enterprise would have the option of buying modules - both software and service-based; and devices – both hardware and embedded from different vendors to put together into a comprehensive authentication solution.

An open reference architecture such as the one specified in this document will serve as a powerful mechanism to foster competition and innovation among all key solution constituents such as device manufacturers, identity management vendors, security service providers, and application developers. It will consequently lower the complexity and cost of deploying strong authentication by applications helping realize the OATH vision of universal strong authentication.

The rest of this document is structured first to provide context for this work in the OATH Vision and Goals, and Usage Scenarios section. We then introduce the high level Authentication Framework. The next four sections do a deeper dive into the four areas that OATH is planning to focus on viz. Client Framework, Validation Framework, Provisioning and Management Framework and finally Common Data Models. The document concludes by describing an Example Deployment Scenario and summarizing the OATH Focus Areas.

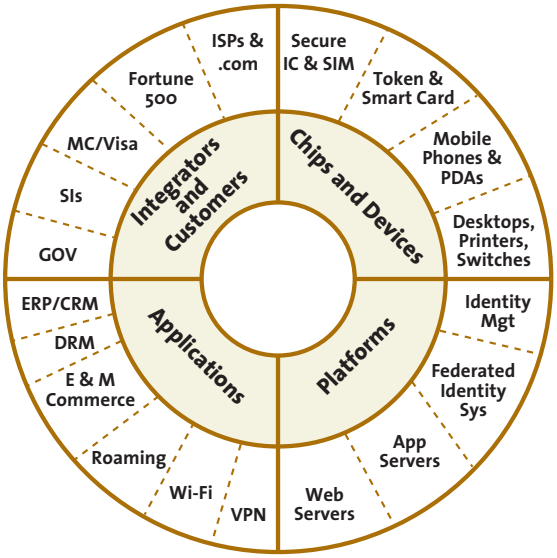
2. OATH Vision and Goals

More and more, organizations are leveraging the benefits of e-business by opening up their networks and applications for access by a wider constituency of employees, business partners and customers. The traditional method of securing that access, the static password, is coming under increased scrutiny by IT and business executives as they come to the realization that password-based access can no longer meet their information security or regulatory compliance requirements. A stronger digital identity must be issued to these information users in order to ensure that valuable information is not accessed, modified, or stolen by unauthorized users and to ensure that the security, privacy and reporting requirements of various industry regulations are firmly met.

For this reason, OATH believes that the voyage toward strong digital identity must start with strong authentication.

Strong authentication is the first pillar of trusted networks where identities can be securely shared and trusted across independent partners. It is the foundation for a more secure network, where all people and all devices are strongly authenticated in an open, interoperable, and federated environment.

To drive adoption of strong authentication across the entire user community— from corporate employees, to Internet users, to people accessing everything from health care records to government services—the industry must collaborate to lower the complexity of and financial barriers to strong authentication. Open technical standards and deployment profiles that promote interoperable solution components are powerful mechanisms for lowering complexity and cost. Therefore, the development of an open and royalty-free specification for strong authentication will be OATH's initial focus. Open, universal strong authentication is intended to provide all key constituencies (device manufacturers, identity management vendors, security service providers, and application developers) with a common framework for strongly authenticating users and devices.



Open Strong Authentication Ecosystem

Open authentication aims at the following goals:

- Establish an open reference architecture for strong authentication by leveraging existing open standards when possible, and otherwise leading standardization efforts through well-established technical standard bodies.
- Propagate low-cost, multi-function authentication devices (e.g., tokens, smart cards).
- Transform today's mobile devices (e.g., cell phones, PDAs, laptops) into strong authentication devices.
- Propagate device credentials, strong authentication algorithms, and authentication client software across many network end points (e.g., desktop computers, servers, switches, Wi-Fi access points, set-top boxes).
- Build around well-established infrastructure components such as directory and RADIUS servers.
- Facilitate native support (platform connectors) for strong device and user authentication in application development and identity management platforms.
- Leverage federated identity protocols as a powerful propagation and integration mechanism.
- Increase the breadth of packaged applications (e.g., enterprise resource planning (ERP), material requirements planning (MRP), and customer relationship management (CRM) application connectors) that support strong authentication.
- Enable best-of-breed solutions through interoperable components.

To be effective, a specification must be jointly defined and published by key industry partners that share the vision of universal strong authentication. By laying the ground for ubiquity, integration, and interoperability, an open architecture can decrease the risk and complexity of deploying strong authentication products. In turn, the promise of reduced risks and costs will drive adoption across enterprises, service providers, and governments around the world. Ultimately, by making strong authentication (all users, all devices) part of the network fabric, the entire user community will benefit. Last but not least, by increasing the trust of the network end points, new types of secure interaction will become possible.

3. Usage Scenarios

In this section we present some scenarios that require strong authentication.

Scenario 1: Remote Access

Salespeople, care workers, engineers and travelling executives all need secure access to the corporate network when 'on the road'. These users will demand the most flexible range of access methods including:

- VPN over wireless when their laptop/PDA can connect to a Wi-Fi hotspot
- VPN over broadband connection from laptop when at home
- Wireless e-mail and other lightweight apps from PDA, Blackberry or other handheld device
- Web access to e-mail and other web-enabled apps from Internet café or other insecure PC

These users must be able to use a single set of secure authentication credentials that can be used at all the access points that the enterprise has enabled.

Scenario 2: Online Banking

With the increasing number of incidences of online identity theft, phishing and credit card theft, several banks have already deployed (mainly in Europe) or are considering deploying strong authentication as one of the key components of a multi-pronged strategy to combat these threats.

The user will use some form of hardware token (most likely an OTP token) in addition to the username and password to access the online banking web-site. Additionally, the user may also be required to authenticate and/or digitally sign at a transactional level. The key requirements for this use case are that these authentication credentials should be easy to use (diverse user population), easy to deploy and cost-effective.

Scenario 3: Telecommuting

In this use case Home-based staff accessing office network over DSL or Cable broadband home Internet connection. The connection is typically secured using an IPSEC or SSL VPN tunnel to provide the user with essentially the same working experience as if they were in the corporate office. Increasingly both computing and VOIP telephony services are being delivered to the remote employee over the broadband connection, allowing all calls, e-mail and other services to be forwarded to the user as though they were in the office.

The security of the remote location cannot be policed by the organisation so it is critical that the user is strongly authenticated before s/he is allowed access to the corporate assets: the user's identity serves as a personal key to the corporate kingdom. It is essential that other members of the user's family, friends, and housemates cannot impersonate the authorised user.

Scenario 4: Client and business partner extranet

Specific individuals at your key clients and business partners need to be granted deep and broad access into your core business systems, typically through web portals. They need to be securely authenticated – it is no longer sufficient to rely on just the IP address of the remote network to validate their identities.

These individuals may be logging in from any web enabled system: a corporate desktop or home PC for example, and so there should be no requirement for any form of reader device to be plugged into the client system.

Scenario 5: e-Government

There is an increasing demand for 'joined-up' government where local authorities, central government departments, law enforcement, healthcare and other agencies all need to communicate more closely to provide the citizen with more coherent and personalised service.

To comply with privacy regulations it is essential that only the specific individuals at each agency who are responsible for each citizen's case should be allowed access to citizens' personal case files, and this access must be independently auditable.

Secure authentication of all officials and citizens is fundamental to the secure delivery of e-Government services, and delivery of a wide range of low cost, flexible OTP credentials is critical to the success of this revolution in public sector services.

Scenario 6: 24x7 IT infrastructure support

Increasingly all SME, corporate and public sector IT systems have to be operational 24x7 to support the non-stop demands of their workforces, clients and consumers. IT staff and outsourced support engineers need to be able to gain instant access to core network infrastructure and servers from a remote office, home or other location they happen to be at when their alert pager beeps.

Access privileges give some of these users substantial power over organisations' entire IT infrastructure, and if any of their identities are stolen, then the thief has complete control. These users must therefore be issued with the strongest possible forms of authentication credentials, and they must be educated to care properly for these credentials.

Scenario 7: Wireless roaming

Wireless users increasingly require 'Anywhere access' throughout corporate offices and from public WiFi hotspots. This use case is analogous to roaming on cell-phone networks. The end-user can seamlessly roam on a guest network. In this case the guest network will communicate with the user's home network to authenticate the user. The home network could be a corporate network or a wireless service provider that the user subscribes to.

Scenario 8: Desktop Logon

In this use case, the user is required to present strong credentials (typically at least two factors) to log into a desktop, which may be a Windows, LINUX or UNIX console. The user is typically required to present a password or a PIN; 2nd factors could be smartcard tokens, OTP tokens, or biometric samples.

4. Authentication Framework

In this section we present the high level architecture for an open authentication system. The picture below shows the high-level logical blocks of a generic authentication system. We describe these blocks briefly in this section. Some of the blocks viz. client framework, validation and provisioning are discussed in further details in the next section.

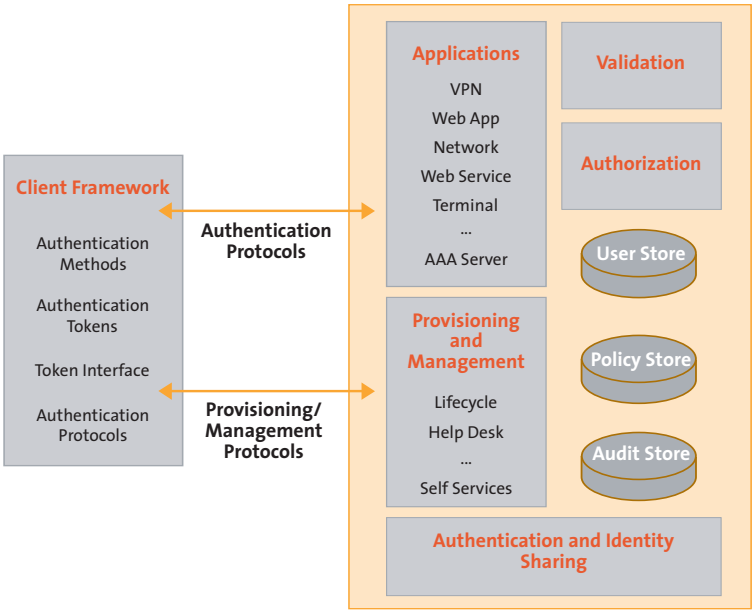


Figure 1: OATH Authentication Architecture

- **Client Framework:** The client framework enables a range of authentication methods, tokens and protocols to be supported when deploying strong authentication across an enterprise or service provider infrastructure. The client framework allows for standards-based integration of multiple forms of strong authentication implemented using either existing or new authentication token technologies, and communicated using standard authentication protocols.

- **Provisioning and Management:** The provisioning and management module is responsible for provisioning and managing the entire life-cycle of software modules and/or security credentials to an authentication device. The purpose of this process is to bring the device from a "clean state" to a state where it can be used as an authentication token e.g. provisioning of certificates, or provisioning of an instance of an OTP token in software or a connected token. The goal of the Provisioning architecture is to accommodate secure and reliable delivery of software and/or security credentials to any client device, using standards-based provisioning protocols or programmatic interfaces.
- **Validation:** The validation module is responsible for validation of the different types of authentication credentials. Various applications that need authentication (such as VPN gateways or web applications) communicate with the validation module using standard validation protocols to authenticate the end-user. The goal of the Validation framework is to enable vendors to write custom validation modules and enables enterprises to deploy multiple types of authenticators in the same infrastructure.
- **Applications:** This includes any application that may need to strongly authenticate the end-user e.g. VPN, Web application, Network router, WI-FI Network, Web Service, etc. The application communicates with the end-user (client framework) using one of the standard based authentication protocol. Once the application has the end-user's strong authentication credential, it can talk to the validation module using one of the supported validation protocols to validate the end-user credentials.
- **Authorization:** Once the user is authenticated, the application may consult an authorization module before granting the user access to the requested resources. The authorization module is responsible for determining whether user is enabled to access a particular resource based on pre-configured access policy e.g. only users who belong to Human Resources department are allowed to edit personnel records. The authorization module although not used to authenticate the user is shown this architecture for completeness.
- **User Store:** User store is responsible for storing all end-user profile information. This will include information such as a unique user identifier (username); profile information such as address, first name, last name, etc.; application specific attributes; and finally it may also store some authentication information such as password, token information, etc. User store is typically an LDAP directory or in some cases it could be a database.
- **Policy Store:** The policy store is responsible for storing all policies. There are two deployment models: the first model uses a common policy store as shown in the diagram above; in the second model some modules (such as validation or authorization) may have their own policy store.

- **Audit Store:** The audit store is a central repository for all audit and sometimes operational events. As with the policy store, an enterprise may choose to deploy a central audit store that collects the audit events across all modules or may optionally allow some of the modules to have their own audit stores.
- **Authentication and Identity Sharing:** Increasingly, organizations are required to share user authentication and identity with other applications. OATH also feels that enterprises may not want to accept on the liability associated with sharing identities but may still want to share a 2nd factor for authentication. This module is responsible for implementing the specific technology primitives that enables sharing of authentication and/or identities.

5. OATH Reference Architecture

5.1 Client Framework

The OATH reference architecture provides a flexible client framework to allow for a range of authentication methods, tokens and protocols to be supported when deploying strong authentication across an enterprise or service provider infrastructure. The client framework allows for standards-based integration of multiple forms of strong authentication implemented using either existing or new authentication token technologies, and communicated using standard authentication protocols. This section presents the high level architecture for the OATH client framework, the salient features of this framework, and then identifies the existing standards/technologies and OATH focus areas for four key elements of the client framework:

- Authentication Methods
- Authentication Tokens
- Token Interface
- Authentication Protocols

5.1.1 High Level Architecture

The OATH client framework architecture is shown in Figure 5.1-1. Each of the framework components and interfaces are defined below.

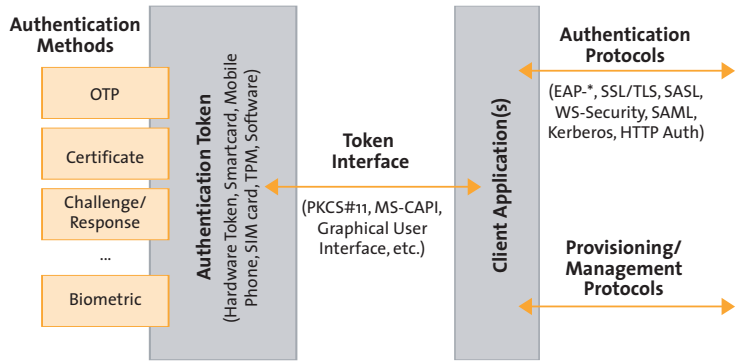


Figure 2: OATH Client Framework Architecture

- **Authentication Method:** a function for authenticating users or devices, including One-Time Password (OTP) algorithms, symmetric key challenge-response, public key certificates, and other methods.
- **Authentication Token:** a hardware or software container implementing one or more authentication methods, performing the security critical client-side authentication operations and the secure storage of the authentication credential(s).

- **Token Interface:** the interface by which an authentication token communicates authentication data and credentials between the Client Application and the Authentication Token. This interface can be either an API when the token is connected to the client application, or a GUI display in the case of standalone token implementations (supported only for some authentication methods such as OTP).
- **Authentication Protocol:** over-the-wire protocol used to exchange authentication data between the Client Application and Server Application. Authentication Protocols support one or more Authentication Methods, with many existing industry standard protocols currently supported today.
- **Provisioning/Management Protocol:** protocol used to provision authentication credentials and authentication token software (in the case of soft tokens), as well as ongoing lifecycle management operations. The OATH Reference Architecture does not address user/identity provisioning, which is covered by other standards organizations. Provisioning and management is addressed separately in Section 5.3.
- **Client Application:** the software client that uses strong authentication methods to access an application function either locally, or remotely over a network connection (such as web access, VPN client, email client), utilizing an authentication protocol to exchange authentication data with remote server applications as required. The Client Application includes a Provisioning Agent component to support client provisioning and management, as defined in Section 5.3. The Client Application supports one or more Token Interfaces, and typically runs on a PC or mobile device owned by a consumer or enterprise user.

5.1.2 Salient Features

OATH Client Framework includes the following key features, to allow for flexible strong authentication deployments ranging in cost, security, user convenience, functionality and performance.

- Standalone token device and embedded token/integrated device implementations
- Connected and unconnected modes of token operation
- Standard authentication algorithms to enable interoperable vendor implementations of Authentication Tokens and Validation Servers (e.g., IETF HOTP)
- Multi-Key tokens (e.g., multiple instances of OTP credentials on one device, each having unique token IDs associated with different service providers)
- Multi-function authentication tokens (multiple authentication methods supported on a single device, such as OTP and PKI certificate-based authentication)
- Multi-factor authentication (supporting two and three factor authentication as part of a single session/negotiation)
- Support for internet-based client applications and mobile/wireless client applications for strong authentication anywhere from any device

- Multiple Token Interface APIs: PKCS #11, MS-CAPI, others in future
- Multiple client applications (e.g., implementing different authentication protocols) sharing a common authentication token on the client
- Trusted proxy server implementation of the client Authentication Token function to support low end mass market client devices (e.g., server-side OTP generation with SMS transmission to mobile phones)

5.1.3 Authentication Methods

At a high level Authentication Methods supported are function-oriented and generally based on data matching. Authentication methods describe how to authenticate individual users and/or devices.

Each authentication method and its relationships with authentication protocols is described below.

5.1.3.1 Existing Standards and Technologies

Password authentication method is the oldest and still most commonly used method for user authentication. Password authentication is no longer considered adequately secure because users can share the same password and because replay attacks are common. Password authentication is not recommended by OATH, but in order to provide a more complete picture of methods it is described here. Password authentication method is based on data matching. The user is identified with a user ID and the authentication password is matched with corresponding data stored for the corresponding user.

OTP (One Time Passwords) authentication (commonly used today) can be divided in two types; synchronous (based on a transformation of a common shared secret and a moving value that is synchronous on both the server side and the client side. This method is what usually is referred to as OTP) and challenge-response (in which a server generates a challenge value that will be transformed by the client based on a secret shared between the client and the server).

Challenge/ Response authentication is usually based on a shared secret transformation using symmetric key hashing techniques. The server side sends the client a Challenge (e.g. 123456). The client uses the challenge as the data input and the shared secret as the key in the transformation. The resulting code is called the Response (e.g. 1a52c9) and is sent back to the server.

Certificate-based authentication uses public key encryption techniques, supported by a public key infrastructure (PKI) for key and certificate management. Digital certificates are issued by a certificate authority and bind the user's identity to their public key. In a typical certificate-based authentication protocol. the client uses a private key to sign a challenge from the server, and the server verifies the signature using the client's certificate along with other PKI-based information provided by the certificate authority..

Biometric authentication is based on a physiological characteristic of a user, such as a fingerprint, iris image or facial image. Biometric authentication represents the "what you are" component of multi-factor authentication. Biometric authentication is based on data matching of the biometric characteristic of the user.

5.1.3.2 OATH Focus Areas

OATH has endorsed a new OTP algorithm standard called HMAC-based OTP [HOTP], based on the HMAC SHA-1 algorithm. It is an event-based OTP algorithm, in which a counter value is used in the OTP calculation and incremented on the client and server after each use. The algorithm has been submitted to the IETF for standardization as an Informational RFC. Areas of future work include possible extensions to the current HOTP algorithm, such as:

- Time-based OTP algorithm variant
- Counter-based re-synchronization method for clients that can send the count value to the server along with the OTP value
- Composite shared secrets (e.g., based on user PIN or other deterministic data for computing the shared secret)
- Addition of a data field for computing OTP values

Additionally, OATH will also look to promote standardization of other low cost authentication technologies, specifically targeted towards consumer usage scenarios. Some of the areas that OATH is investigating include scratch-cards and methods derived from battleship or bingo cards.[2]

5.1.4 Authentication Tokens

The OATH client framework is designed to support existing hardware token implementations, as well as soft token implementations on PCs. mobile phones and PDAs, and new technologies that combine multiple authentication methods on a single token. Typical authentication system deployments will support multiple types of tokens to cover a broad range of user profiles, security requirements and application scenarios. By using standard token interfaces and standard authentication algorithms such as IETF HOTP [HOTP], a range of authentication tokens can be supported in a uniform way within the OATH reference architecture.

1 *Scratch cards are cards that may contain a series of authentication codes. The user needs to scratch a piece of paper to reveal a dynamic authentication code that can be used to access to the application in addition to a username and a password.*

2 *Battleship or Bingo cards - Each user will have a card containing a unique 2-dimensional grid of characters. During authentication step, the application will challenge the user to enter characters that are located at a dynamically generated set of co-ordinates on the card.*

5.1.4.1 Existing Token Types

There are many forms of existing authentication tokens deployed today, including:

- Standalone OTP generators
- Smart Cards
- USB Key FOBs
- Software tokens
- Trusted Platform Module (TPM) [TPM]

5.1.4.2 OATH Focus Areas

OATH's objective in the token arena is to foster innovation by encouraging embedding strong authentication technologies into devices you already carry such as mobile phones, and to provide flexibility and cost advantages through multi-function tokens. To allow for innovative token solutions, OATH intends to champion the development of standards that are easily implemented on a range of token types, as in the case of the HOTP algorithm. Specific areas of focus for tokens are:

- Soft tokens on mobile devices
- SIM-based authentication tokens
- Multi-Key Token – wallet features for support multiple services using the same physical token device for authentication
- Reference implementations of HOTP algorithm to accelerate adoption, such as Java smart card.

In addition OATH also intends to encourage standardization of certain aspects such as namespace for token serial numbers to increase interoperability across vendors.

5.1.5 Token Interface

The OATH reference architecture assumes industry standard interfaces between the Client Application and the Authentication Token to enable interoperability across different vendor tokens and client application implementations. The client framework also allows for integrated token/application implementations such as a soft token or SIM card on a mobile device for wireless remote access, or a standalone token device operating in either connected or unconnected mode. In connected mode, the token device interfaces to the client application via a standard cryptographic API such as PKCS #11 or MS-CAPI. In unconnected mode, the token requires a user interface to display authentication values such as one time passwords, and optionally a keypad to enter a PIN or password. Token Interfaces must support both single function and multi-function tokens (supporting more than one authentication method), such as OTP and certificate-based authentication on a USB token.

5.1.5.1 Existing Standards and Technologies

Token interface standards already exist for PKI-based authentication, specifically:

- PKCS #11 [PKCS#11]
- Microsoft CryptoAPI (MS-CAPI) [CAPI] for Windows

Other authentication methods rely on proprietary token interfaces at present.

5.1.5.2 OATH Focus Areas

The OATH reference architecture envisions both extensions to existing industry standard APIs and the emergence of new APIs for the Authentication Token interface.

- Extensions existing APIs to support OTP algorithms,
  - PKCS#11 Mechanisms for One-Time Password Tokens, (http://www.rsasecurity.com/rsalabs/node.asp?id=2818) is being extended to support the IETF HOTP mechanism; the new mechanism name is CKM_HOTP.
  - MS-CAPI will require extensions to support IETF HOTP
  - Other device-specific API extensions as HOTP gains adoption
- New Token Interface APIs foreseen
  - A platform-agnostic API for OTP software token interface standard, allowing vendor interoperability for soft tokens and client applications
  - A SIM interface standard for an application to request an OTP from the SIM.

5.1.6 Authentication Protocols

Over-the-wire authentication protocols are used to exchange authentication data between the client and server application. Each authentication protocol supports one or more authentication methods. The OATH reference architecture provides for the use of existing protocols, and envisions the use of extended protocols which support new authentication methods as they are defined.

5.1.6.1 Existing Standards and Technologies

Providing a comprehensive framework for authentication services requires that we first have an understanding of protocols and mechanisms widely in use in systems today. Listed below are the most common authentication protocols with short descriptions of their use and references to where to find out more details about them.

- **CHAP (Challenge-Handshake Authentication Protocol) [RFC1334] [RFC1994]:** is an authentication protocol used to log on a user to an Internet access provider. Widely used in early dial-up services.

- **EAP Extensible Authentication Protocol [RFC3748]:** Extensible Authentication Protocol is used between a dial-in client and server to determine what authentication protocol will be used. EAP is also widely used for other client server authentication services.
- **GSS-API:** Generic Security Service Application Program Interface [RFC1508] - Provides security services to callers in a generic fashion, supportable with a range of underlying mechanisms and technologies and hence allowing source-level portability of applications to different environments. The authentication specified in 1508 is very generic and further defined in other RFCs that build on GSS-API as the base.
- **HTTP Basic Access Authentication [RFC2616]:** Provides basic username/ password authentication as specified in HTTP 1.1. Commonly used in combination with SSL by which the password is encrypted over the network, but still possesses the fundamental weakness associated with simple reusable passwords.
- **Kerberos [RFC1510]:** Network Authentication protocol used in a distributed networking environment. Based on the principle that the user authenticate to a Ticket Authentication server. The Authentication server grants the user rights to authentication tickets on one or more Ticket Server(s) that hands out authentication tickets to any application that the user is have rights to use. Kerberos acts much like a single sign on solution between applications and trusted computers. Kerberos is as an example used in Microsoft Windows 2000 and above.
- **MS-CHAP v1 and v2 [RFC2433] [RFC2759]:** Microsoft's PPP CHAP dialects (MS-CHAP), which extend the user authentication functionality, provided on Windows networks to remote workstations. MS-CHAP is closely derived from the PPP Challenge Handshake Authentication Protocol.
- **PAP [RFC1334] [RFC1994:** Password Authentication Protocol is a two way handshake protocol designed for use with PPP. Authentication Protocol Password Authentication Protocol is a plain text password used on older SLIP systems. It is not secure since it sends the credentials in clear text.
- **Security Assertion Markup Language [SAML]:** is a standard describing security assertions that are encoded in XML. Also defines a set of Profiles for exchanging SAML assertions and transport Bindings.
- **SASL [RFC2222]:** The Simple Authentication and Security Layer is a method for adding authentication support to connection-based protocols.
- **S/Key [RFC1760] [RFC2289]:** An early one time password system based on hashing that secures against password replays.
- **SSL/TLS [RFC2246]:** the transport layer security protocol widely supported in standard internet browsers and web servers, supporting PKI certificate based authentication for both server and client.

- **WS Security OASIS Web Service Security [WSS]:** describes enhancements to SOAP messaging to provide quality of protection through message integrity, message confidentiality, and single message authentication. These mechanisms can be used to accommodate a wide variety of security models and encryption technologies. WS-Security also provides a general-purpose mechanism for associating security tokens with messages.

5.1.6.2 OATH Focus Areas

OATH intends to encourage extensions to existing authentication protocols to support new standard authentication methods, with IETF HOTP as the initial case, in the following areas:

- **EAP-OTP:** OATH members will work within the existing IETF EAP framework to develop a standard EAP mechanism for OTP authentication
- **Web Services:** OATH intends to promote the development a standard web services interface to support OTP, in context of the OASIS Web Services Security specification (WS-Security).
- **Sideband Signaling:** Additionally, OATH plans to investigate requirements and scenarios for use of sideband signalling in context of authentication protocols. The various authentication protocols of section 5.1.6.1 are, to varying degrees, subject to Identity Theft, Man-in-the-Middle and Trojan attacks. In these attacks the 'bad' guy attempts to impersonate a user through use of legitimate credentials at the remote computing resource. Sideband signaling may be used to mitigate these threats. Sideband signaling can be directly integrated into the authentication protocol. This is a particularly appealing approach in situations where wireless devices and wired devices are both participating in the authentication process thereby providing multiple IP communications paths to a remote computing resource. In addition to direct integration into the authentication protocols sideband signaling may also be used to select the authentication protocol to be used. Sideband signaling can further mitigate these attacks by requiring explicit out-of-band notification and/or confirmation of high value actions.

5.2 Validation Framework

In any large organization, there are several applications that need strong authentication. It is unlikely that the same authentication method will satisfy all application and user constituents. Today any organization needs to deploy multiple authentication solutions and possibly separate infrastructure to support these solutions.

The OATH Validation framework is an architecture that will enable vendors to write custom validation modules and enables enterprises to deploy multiple types of authenticators in the same infrastructure. Additionally, the validation framework will enable organizations to deploy multiple protocol handlers such as RADIUS, OCSP, WS-Security, etc.

The OATH validation framework will benefit the organization that deploys strong authentication by enabling use of multiple authentication methods in the same infrastructure, thereby enabling phased rollout of strong authentication solutions across a wider set of applications and user groups.

Today authentication vendors need to build a complete validation server. Implementations of the OATH Validation framework will remove the necessity for a vendor to rebuild a complete solution. It is possible to imagine that some vendors may specialize in validation servers that comply with this framework. Vendors that have existing identity management servers such as AAA servers may choose to implement this framework within their existing products. Other vendors may choose to provide hosted validation services that are based on this framework. This framework will enable vendors to develop pluggable modules that are specific to the authentication method that they are bringing to market. It will enable vendors to bring innovative solutions to market quicker.

5.2.1 High Level Architecture

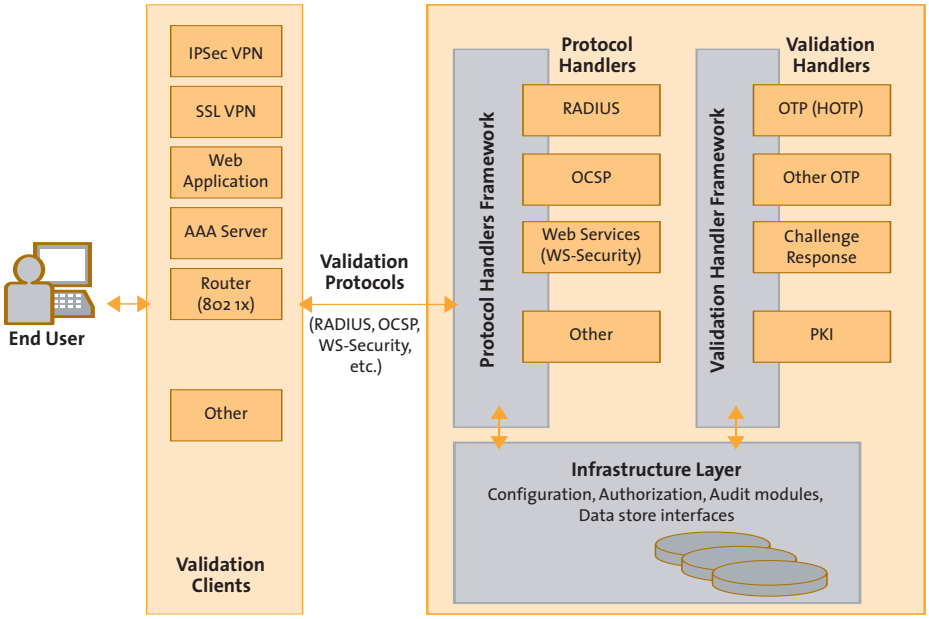


Figure 3: OATH Validation Framework

As shown in the figure above, the validation framework will consist of the following sub-modules:

- **Protocol Handler framework:** This framework will enable deployment of multiple protocol handlers. The framework will be responsible for I/O, threading, loading the handlers, etc. This framework will specify the necessary interfaces and configuration.
  - **Protocol handlers:** A protocol handler is the component that will implement support for a particular validation protocol such as RADIUS or OCSP or WS-Security.

- **Validation Handler framework:** This framework will enable deployment of multiple validation handlers. Among other things this framework will specify the necessary interfaces and configuration and will be responsible for loading the various validation handlers installed in the system.
  - **Validation handlers:** Each validation handler will support validation of a particular instance or flavour of an authentication method e.g. HOTP, RSA SecurID, etc.
- **Infrastructure Layer:** The infrastructure layer will provide some of the common functionality that can be used by the various other components in the validation system. These include the various protocol and validation handlers as well as the framework themselves.
  - **Configuration module:** This module will allow you to register and configure the various entities in the system such as validation clients, protocol handlers and validation handlers.
  - **Authorization module:** This module will allow you configure the access policies of which validation clients can have access to which set of validation handlers.
  - **Audit/Logging module:** This module will allow you log various audit and operational events from the various components in the validation framework.
  - **Data store interfaces:** The various interfaces in the system are listed below. These interfaces will enable the rest of system components to store various data agnostic of the specific data store implementation.
    - Configuration Store Interface
    - Token Store Interface
    - Audit/Log store Interface

5.2.2 Salient Features

The salient features of the OATH validation framework are:

- **Multiple authentication methods:** The validation framework will support validation of multiple authentication methods such as certificates, OTPs, challenge response, etc. simultaneously. Note that the framework will enable organizations to change the supported authentication methods by adding or removing validation handlers. Additionally, the validation framework will also enable an enterprise to deploy multiple flavours of a particular authentication method that can for example validate OTPs from different vendors or validate certificates from different certificate authorities (CAs). The validation framework will also have the ability to validate more than one authentication credential in the same transaction.

- **Multiple validation protocols:** The validation framework will enable validation clients to submit validation requests using a variety of protocols such as RADIUS, CRL, OCSP, Web Services, etc. simultaneously. The validation framework will enable organizations to change the supported protocols by adding or removing the deployed list of protocol handlers in the system.
- **Multiple validation clients:** The validation framework will enable multiple applications that need to validate authenticate validation credentials (validation clients) to send requests simultaneously. This framework will enable administrators to configure authorization policies around which validation clients can validate against which set of validation handlers.
- **Standardized Configuration:** Validation framework will support standard primitives to configure the various validation and protocol handlers. It should also enable a standard way to register and query the various handlers that are active in the system.
- **Framework for logging operational and audit events:** The validation framework will support common methods to enable the various handlers as well as the framework components to log operational and audit events.
- **Deployment agnostic:** The validation framework will enable validation do be deployed in-premise within an organization's infrastructure as well as a hosted service. Additionally, specific validation handlers may choose to make remote calls to distributed components to perform validation of credentials.
- **Data Store abstraction:** The validation framework will provide appropriate data store interfaces for various types of data such as token data, audit and log data, and configuration data. This will let an organization use that data store that best meets its requirements.

5.2.3 Existing Standards and Technologies

Existing Validation Protocols:

- **Lightweight Directory Access Protocol (LDAP)** is a protocol for accessing on-line directory services. LDAP defines a relatively simple protocol for updating and searching directories running over TCP/IP. Among other things, directories are used to centrally store information about end-users including usernames and passwords. Consequently, LDAP is often used by applications to validate the username and password for end-users
- **OCSP [RFC2560]:** Online Certificate Status Protocol is a method for determining the revocation status of an X.509 digital certificate using means other than CRLs. OCSP can provide more timely information regarding the revocation status of a certificate. OCSP's request/response nature leads to OCSP servers being termed as *OCSP responders.*

- **RADIUS [RFC2138]:** (Remote Authentication Dial-In User Service) is an Authentication, Authorization and Accounting (AAA) protocol for applications such as network access – used in both local and roaming situations..
- **TACACS(+) [RFC1492]** is a remote authentication protocol that is used to communicate with an authentication server commonly used in UNIX networks. TACACS allows a remote access server to communicate with an authentication server in order to determine if the user has access to the network..
- **XKMS [XKMS]:** XML Key Management Specification provides a secure method for registration and subsequent life-cycle management for public key information. The XML Key Management Specification (XKMS) comprises two parts -- the XML Key Information Service Specification (X-KISS) and the XML Key Registration Service Specification (X-KRSS).X-KISS may be used to validate the signature and the certificate associated with that signature.
- **WS-Security [WSS]:** is a protocol used for providing secure communications for Web Services. The protocol contains specifications on how Authentication and Confidentiality can be enforced on Web Services messaging. The WSS protocol encompasses the use of authentication methods (tokens) such as SAML and X.509.

Existing Authentication and Validation Interfaces:

- **PAM (Pluggable authentication modules) [PAM]:** is a generalized API for authentication-related services which allows a system administrator to add new authentication methods simply by installing new PAM modules, and to modify authentication policies by editing configuration files. PAM was first developed by Sun Microsystems, and is currently supported in Solaris, Linux, FreeBSD and NetBSD.
- **JAAS (Java Authentication and Authorization Service) [JAAS]** is an API that enables Java applications to access authentication and access control services without being tied to those services, It implements a Java technology version of the standard Pluggable Authentication Module (PAM) framework, and supports user-based authorization. This permits Java applications to remain independent from underlying authentication technologies. New or updated technologies can be plugged in without requiring modifications to the application itself. Several sample authentication modules are available that implement JNDI (Java Naming and Directory Interface), UNIX, Kerberos, and Windows NT authentication.

5.2.4 OATH Focus Areas

- OATH intends to promote the development of appropriate interfaces (for both protocol and validation handlers) that will enable vendors to write pluggable handlers as described above.
- Existing validation protocols that are described above may be used by applications to authenticate end-user's credentials. However, these protocols may not be adequate to support the various authentication methods (credentials) and the various deployment topologies (enterprise hosted, outsourced). OATH will also evaluate the requirement for standardized extensions to existing protocols (e.g. RADIUS), and will investigate whether a requirement exists for one or more additional validation protocols to meet strong authentication requirements.

5.3 Client Provisioning and Management

Devices come in many shapes and sizes, and their capabilities and functionality varies widely. It is challenging to implement a single protocol for provisioning and subsequent life cycle management of software and different types of security credentials across all Devices. The goal of the OATH Provisioning and Management Architecture is to specify a framework that can accommodate support for multiple standards-based provisioning protocols to enable provisioning of different type of credentials across all types of Devices. Additionally, for some platforms this framework will also enable provisioning of authentication software to the device.

Typically, Software can be provisioned to the Device using one of the two following methods:

1. Embedded at manufacturing or personalisation time of the Device e.g. SIM cards, hardware tokens
2. Loaded to the Device post-personalization over a network interface e.g. PDAs and cell phones

Similarly, the associated security credentials can be provisioned to the Device using in one of the following methods:

1. Embedded at manufacturing or personalisation time of the Device e.g. SIM cards, smartcards, hardware tokens
2. Loaded to the Device post-personalization over a network interface e.g. smartcards, PDAs and cell phones

Security credentials may also be provisioned to a Secure Repository and made accessible to a Trusted Proxy Server such as the SMS OTP Proxy server in order to generate OTP values on behalf of the Device.

The OATH provisioning architecture offers a generic and extensible framework for provisioning the Authentication Token software modules and/or associated security credentials onto Devices.

5.3.1 High Level Architecture

The figure below illustrates the OATH Client Provisioning and Management Architecture.

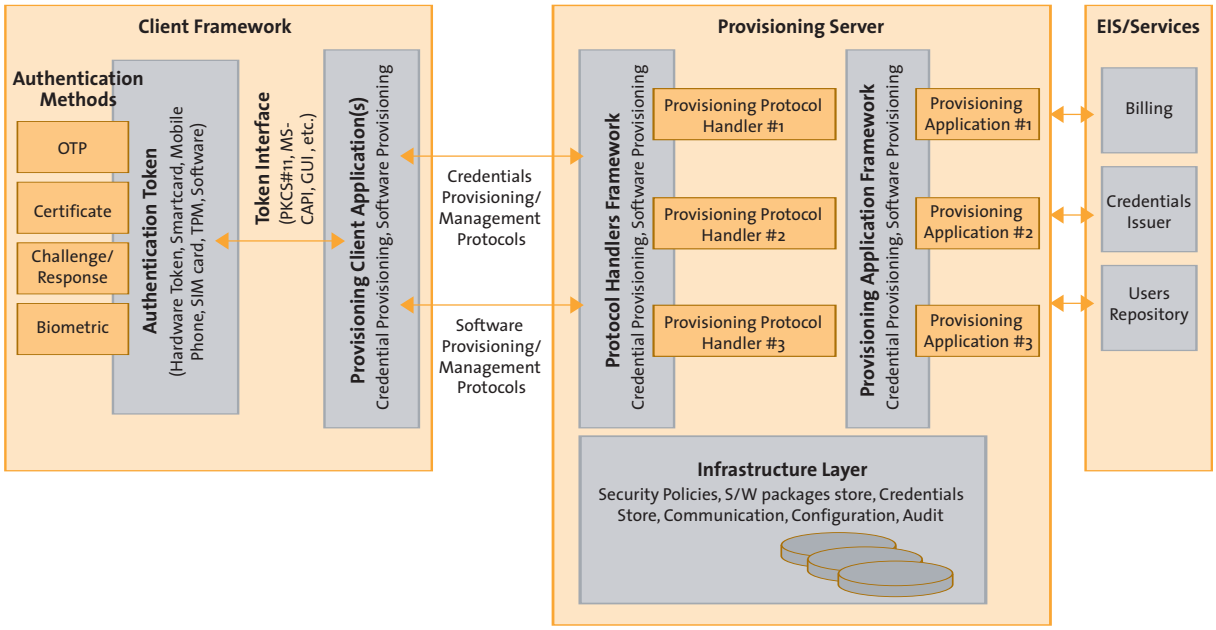


Figure 4: OATH Client Provisioning Architecture

As shown in the figure above, the provisioning architecture will consists of three (3) tiers:

1. The Provisioning Client Application
2. The Provisioning Server
3. Enterprise Information Systems or Services

5.3.1.1 The Provisioning Client Application

The Provisioning Client Application runs on the Device or a personal computer. The Provisioning Client Application is responsible for provisioning the Authentication Token software modules and/or associated security credentials on to the Device. The Provisioning client Application may implement one or multiple provisioning protocols and may support one or multiple types of credentials. Multiple Provisioning Client Applications may also co-exist on the same device.

5.3.1.2 The Provisioning Server

As shown in the figure above the Provisioning Server consists of:

1. **Protocol Handler Framework:** the Protocol Handler Framework is built on top of the Infrastructure Layer and supports the Provisioning Application Framework functions through a generic application-programming interface. Different Device platforms require different provisioning protocols. The framework will enable deployment of multiple provisioning protocols by specifying the necessary interfaces and configuration. The Protocol Handler Framework may host several Protocol Handlers. These may include Credentials specific Protocol Handlers such as XKMS and RSA CT-KIP or software specific Protocol Handlers such as J2ME DL and OMA DLOTA.
2. **The Provisioning Application Framework:** the *Provisioning Application Framework* defines generic interfaces and configuration for building and deploying Provisioning Applications. A Provisioning Application implements the business rules for processing provisioning requests for credentials and/or software. The Provisioning application may process provisioning requests internally or delegate the requests to external systems or services such as external Credential Issuers. Multiple provisioning applications can coexist within the Provisioning Application Framework. For example one application may be responsible for provisioning and managing the life cycle of security credentials such as HOTP secrets while another application is responsible for provisioning and managing updates of the Authentication Token software modules on the Devices.
3. **The Infrastructure Layer:** the Infrastructure layer forms the base of the provisioning system. It provides some of the common functionality and services such as client authentication, configuration, persistent storage and communication to the top level components of the system i.e. the Provisioning Application Framework or the Protocol Handler Framework. The Infrastructure Layer services may be offered by several modules including:
   - **Customer management module:** this module allows access and management of customer records and their provisioning and management services orders.
   - **Device Inventory module:** this module allows registering and configuring supported Devices in the system such as PDAs, cell phones, SIM cards.
   - **Configuration module:** this module allows registering and configuring the various entities in the system such as Protocol Handlers, Provisioning Applications and Interfaces to external EIS and services.
   - **Security Policies module:** this module allows to configure the access policies of which provisioning clients can have access to which set of provisioning handlers.
   - **Persistent store interfaces:** these interfaces will enable the system components to store and/or access various persistent data types such as configuration data, credentials, software packages, Security policies, business rules and audits.

5.3.1.3 Enterprise Information Systems / Services

The Provisioning Application may connect through the Infrastructure Layer to various external Enterprise Information systems or Services in order to fulfil the business rules. An external system may be for example:

- Credentials Issuer
- Billing / Payment system
- Enterprise User directory

5.3.2 Salient Features

The OATH provisioning architecture supports the following key features:

- **Credentials Provisioning:** the *Protocols Handler Framework* will enable implementation and deployment of secure protocols for credentials provisioning and subsequent life-cycle management (renewal, revocation, suspension and reactivation).
- **Software Provisioning:** the *Protocols Handler Framework* will enable implementation of secure protocols for software provisioning and subsequent life-cycle management (update, uninstall).
- **Multiple Credential Types:** the *Protocols Handler Framework* will allow for multiple credential provisioning protocols to co-exist and hence allow for provisioning of different types of credentials (shared keys, certificates, etc) to various device platforms.
- **Multiple Device types:** the provisioning architecture will enable support for a wide range of Devices types.
- **Optimized Provisioning Protocols:** the *Protocols Handler Framework* will allow supporting highly optimized and customized provisioning protocols to cope with device constraints such as network latency, bandwidth, memory and processing power.
- **Multiple Provisioning Client Applications:** The *Client Framework* will enable multiple provisioning client applications to co-exit on the Device and handle various types of credentials.
- **Standardized Configuration:** the Provisioning framework will support standard primitives to configure, register and query the various *Protocol Handlers* and *Provisioning Applications* in the system.
- **Framework for logging operational and audit events:** The provisioning architecture will support common methods to enable the various *Protocol Handlers* as well as the *Provisioning Applications* to log operational and audit events.
- **Data Store abstraction:** The provisioning architecture will provide appropriate data store interfaces for various types of data such as credential data, audit and log data, and configuration data. This will let an organization use that data store that best meets its requirements.

5.3.3 Existing standards and technologies

There are several methods for secure credential provisioning on connected devices. Existing methods and protocols are usually designed to support one type of credential. The objective of OATH provisioning framework is to accommodate support for credential issuance, provisioning and other lifecycle management functions for all types of credentials (i.e. symmetric keys, RSA key pairs, certificates) and across all types of Devices.

5.3.3.1 Existing Credential Provisioning Protocols

This section describes some of the credentials provisioning protocol and other related security protocols that can be leveraged by the OATH provisioning architecture.

- **XML Key Management Specification (XKMS):** XKMS [XKMS] provides a secure method for registration and subsequent life-cycle management for public key information. The XML Key Management Specification (XKMS) comprises two parts — the XML Key Information Service Specification (X-KISS) and the XML Key Registration Service Specification (X-KRSS). The X-KRSS specification defines a protocol for a web service that accepts registration of public key information. Once registered, the public key may be used in conjunction with other web services including X-KISS.
- **PKCS #10 / PKCS #7:** The PKCS #10 [RFC 2986] defines a standard syntax for a certificate requests. Certificate requests are sent to a Certification Authority, which transforms the request into an X.509 public-key certificate. The resulting certificate or certificate chain is usually returned in PKCS #7 format [RFC2315]. PKCS #10 and PKCS #7 are widely supported in public-key infrastructures and public-key enabled applications.
- **Simple Certificate Enrollment Protocol (SCEP):** SCEP [SCEP] is a relatively new protocol proposed by CISCO as an Internet Standard. Its most compelling feature is the possibility to automatically enroll certificates for large scale installations: each certificate requesting entity is authenticated by a password that is used only once, during the enrollment, but is never transmitted in clear. SCEP supports RSA public key algorithm only and leverages PKCS #7.
- **Simple Password-authenticated Exponential Key Exchange (SPEKE):** SPEKE [SPEKE] is a cryptographic system for zero-knowledge password proof key exchange. It provides authentication and key establishment over an insecure channel using only a small password, without risk of network attacks. SPEKE is a variant of Diffie-Hellman Encrypted Key Exchange (DH-EKE) [DHEKE].

- **RSA Crypto Token Key Initialization Protocols (CT-KIP) Proposal:** The RSA CT-KIP [CTKIP] proposal provides a secure method of initializing and configuring cryptographic tokens with secret keys without exposing generated secrets to any other entities than the server and the cryptographic token it-self. The protocol does not require private-key capabilities in the cryptographic tokens and does not mandate an established public-key infrastructure. The initialization session may be secured either using a pre-shared key between the client and the server or using the server's public-key.

5.3.3.2 Existing Software Provisioning Protocols

This section describes different methods and protocols that can be used for downloading the Authentication Token software modules to Devices.

- **Browser based Download over HTTP/S:** Software modules can be downloaded to most Devices using standard mobile internet browsers over HTTPS.
- **OMA Download Over-the-Air (DLOTA) Protocol:** The DLOTA [DLOTA] protocol is defined by OMA Forum. It allows discovery and download of content and applications to mobile Devices. The protocol can be leveraged to download and the Authentication Token software modules. The DLOTA security relies fully on the transport layer security i.e. it uses HTTP basic authentication.
- **Java MIDP OTA Provisioning:** The Java MIDP 2.0 Download [MIDPOTA] Allows discovery and delivery of Java MIDLets to Java Devices.
- **GSM 03.48 Applet Download:** GSM 03.04 [GSM0348] defines a secure protocol for over the air delivery and subsequent life cycle management of SIM applets to SIM cards.

5.3.4 OATH focus areas

- Implementations of the OATH Provisioning Architecture will allow vendors to implement existing standards-based provisioning protocols. The framework will also enable vendors and customers to deploy proprietary provisioning protocols for provisioning and managing specific credential and device.
- Existing protocols (such as XKMS) can be used to provision public key information as well shared keys. However, given the limitations low end devices such as SIM cards and some mobile phones such protocols may not be suitable. OATH will also evaluate the requirement for standardizing one or more provisioning protocols to target specific credential types.
- Different credential provisioning protocols have different requirements for handling user, authentication token and credential identification. OATH will explore the requirement for standardizing a common mechanism for managing these identities.

5.4 Common Data Model

The requirement for a common data model derives from two OATH principles

- Minimize impact on existing infrastructure and leverage existing infrastructure.
- Drive interoperability and best-of-breed solutions.

User Store schema extensions

Most enterprises have spent significant resources in the last few years to centralize their user stores either employee directories or customer databases. By standardizing the user store schema extensions required to support strong credentials, and working with the directory vendors to make this part of their default schema we will be minimizing the impact on the user directory. Additionally, this will also help standardize the provisioning and management of the credential attributes, allowing the enterprise to deploy a best-of-breed solution for the provisioning, management and life-cycle of the credentials that is integrated with the rest of the system.

Token meta-data

Standardized token meta-data will compliment the validation and provisioning framework described above. Token meta-data standardization will enable token vendors to ship token information including key material (shared secrets) in standardized formats that can be imported into validation/provisioning modules that may be implemented by other vendors.

5.4.1 Existing standards and technologies

- **inetOrgPerson [RFC2798]:** The inetOrgPerson object class is a general purpose object class that holds attributes about people. The attributes it holds were chosen to accommodate information requirements found in typical Internet and Intranet directory service deployments. The inetOrgPerson object class is designed to be used within directory services based on the LDAP [RFC2251] and the X.500 family of protocols, and it should be useful in other contexts as well. There is no requirement for directory services implementers to use the inetOrgPerson object class; it is simply presented as well-documented class that implementers can choose to use if they find it useful.

5.4.2 OATH Focus Areas

- OATH plans to encourage development of standardize schema extensions for common user stores such as LDAP directories.
- Also as specified above, OATH intends to encourage development of standardized portable token meta-data formats that will complement the provisioning and validation frameworks defined above.

6. Example Deployment Scenario

In this section we show an example that illustrates how an implementation of the OATH reference architecture might be deployed in the real-world. In this example, a fictitious bank – MyBank – has deployed 2-factor authentication for their retail banking scenario. The bank has a mixed user population with three types of users:

- The first category of users is mobile users who need to access their banking application from multiple locations. Also these set of users have next generation mobile phones that are capable of acting as a container for OTP credentials.
- The second category of users is users who mainly access their bank from a single location.
- Finally, a third category of users already have tokens from vendor B provisioned as part of a corporate banking application.

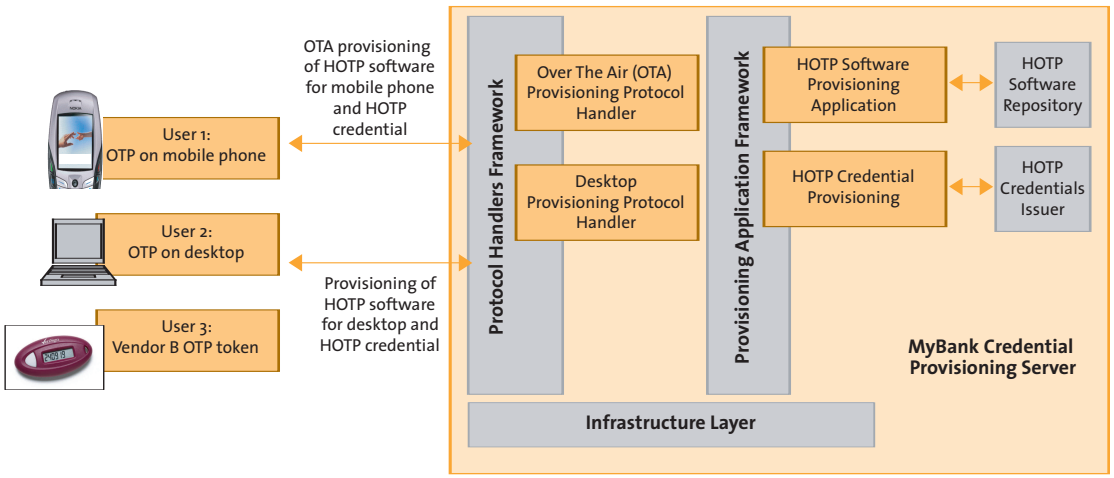


Figure 5: Example Provisioning Deployment Scenario

The figure above shows the provisioning infrastructure[3] deployed by MyBank. MyBank has deployed two provisioning protocol handlers-one can handle over-the-air (OTA) provisioning of authentication software and credentials to mobile devices. A second handler can handle provisioning of authentication software and credentials for desktop based PCs. MyBank has also deployed two provisioning applications – a software provisioning application for HOTP and a HOTP credential provisioning application. These applications talk to the HOTP software repository and the HOTP credential issuer respectively.

3 In order to reduce the complexity of the picture we have not shown the expanded client framework. You can assume the presence of the appropriate client framework pieces on both the mobile phone and the desktop.

As shown in the figure we have three users representing the user categories identified earlier. User 1 has a mobile phone, and is provisioned with the necessary HOTP software token and HOTP credential (shared secret)[4] using the OTA provisioning protocol. User 2 is similarly provisioned with the necessary HOTP software token for desktop and the HOTP credential using the desktop provisioning protocol. Finally, user 3 already has an OTP token from vendor B. At this point all three representative users can now access MyBank's retail banking application using strong authentication as shown.

The figure below shows the validation infrastructure deployed by MyBank. The validation deployment has two different protocol handlers that can accept RADIUS and WS-Security validation requests. Also, MyBank has two validation handlers to validate the two different flavors of OTP credentials (HOTP and vendor B's OTP tokens) that have been deployed to the user population. This particular deployment also has a password validation handler that the application may decide to leverage.

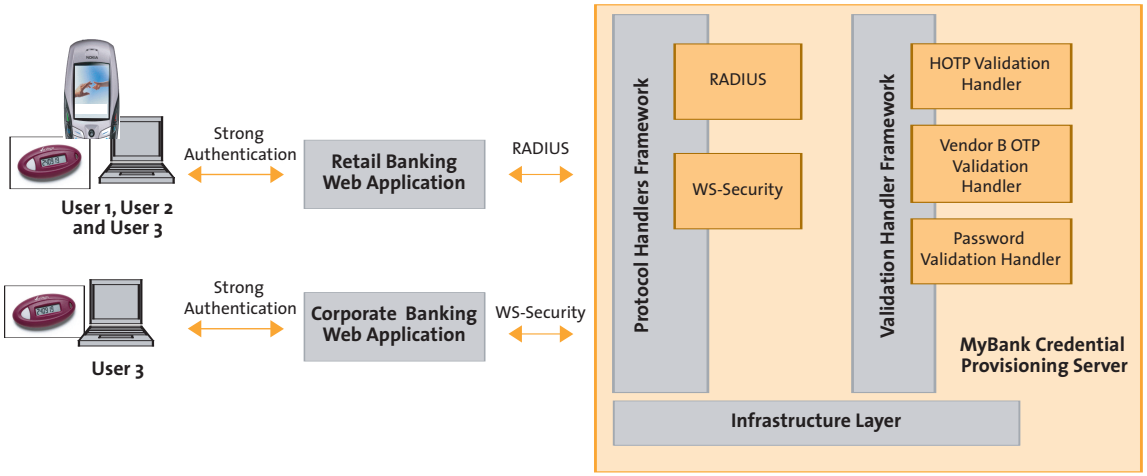


Figure 6: Example Validation Deployment Scenario

There are two applications that make use of this validation infrastructure – a retail banking web application that communicates to the validation server using RADIUS as a validation protocol, and a corporate banking web application that uses WS-Security as the validation protocol.

As shown in the figure, all three representative users can now access the retail }banking application using username, password and the OTP value generated by their respective OTP tokens viz. HOTP soft token on phone, HOTP soft token on desktop or the standalone token from vendor B. The validation server will route the request to the appropriate handler(s) to validate the user credentials.

4 Note that the HOTP software token may have multi-key capabilities. In this case, the user can be provisioned with more than one instance of HOTP credential – one from MyBank and another one from say the brokerage provider or the ISP, making the user's phone or desktop a credential wallet.

User 3 can additionally continue to log into the corporate banking application using the OTP token. The validation request will be received by the WS-Security protocol handler and routed to the Vendor B Validation Handler to be validated.

As the example shows, OATH reference architecture has enabled MyBank to consolidate the various credential provisioning, management and validation systems into a single infrastructure that can service its diverse user populations and application requirements.

7. Summary of OATH Focus Areas

The reference architecture described above, addressed four main areas: Client Framework, Validation Framework, Client Provisioning/Management, and Common Data Model. The focus areas that we identified above are summarized below:

Client Framework

- **Authentication Methods:** encourage standardization of an industry standard OTP algorithm to enable client-server interoperability for two-factor authentication, with continued work to define extensions to this algorithm. Additionally, OATH will also look to promote standard, lost-cost authentication methods for consumer usage scenarios.
- **Authentication Tokens:** foster innovation in tokens by embedding authentication technologies into devices you already carry such as mobile phones, and providing flexibility and cost advantages through multi-function tokens and multi-key tokens. OATH is also considering the requirement for standards governing certain aspects such as namespace for the token serial number to increase interoperability across vendors.
- **Token Interface:** promote extensions to existing APIs to support OTP algorithms such as HOTP and define new token interface APIs, including a standard software token OTP API, and a SIM OTP API.
- **Authentication Protocols:** a) EAP-OTP: encourage the development, within the existing IETF EAP framework, of a standard EAP mechanism for OTP authentication; b) Web services: encourage the development of a standard web services interface to support OTP, in context of the OASIS Web Services Security specification. Additionally, OATH plans to research the requirement for use of sideband signalling through separate IP channels (e.g., wired and wireless) to address more advanced account hijacking threats associated with ID theft and phishing/pharming.

Validation Framework

- OATH will promote the development of appropriate interfaces (for both protocol and validation handlers) that will enable vendors to write pluggable validation handlers.
- OATH will also evaluate the requirement for either specifying standardized extensions to existing validation protocols (e.g. RADIUS) or for standardizing one or more additional validation protocols to target specific credential types.

Client Provisioning and Management Framework

- OATH intends to promote development of a framework that would enable existing standards-based provisioning protocols, and also enable vendors and customers to deploy proprietary provisioning protocols for provisioning and managing specific types of credentials and devices.
- OATH will also evaluate the requirement for standardizing one or more provisioning protocols to target specific credential types or deployment scenarios.

Common Data Model

- OATH plans to promote the definition of standard user store extensions and OTP Token meta-data to support open authentication.

8. Feedback

The Initiative for Open AuTHentication (OATH) would like to receive input, suggestions and other feedback ("Feedback") on this work from a wide variety of industry participants to improve its quality over time.

Feedback on this document should be directed to oath_technical@v2.listbox.com.

If you are interested in getting more information about OATH or joining OATH, please contact info@openauthentication.org or visit http://www.openauthentication.org.

9. References

[CAPI] Microsoft Corporation, “Microsoft Crypto API”, http://msdn.microsoft.com/library/default.asp?url=/library/en-us/seccrypto/security/cryptography__cryptoapi__and_capicom.asp

[CTKIP] RSA Laboratories, CT-KIP: Cryptographic Token Key Initialization Protocol version , DRAFT2, April 14th, 2005, URL : ftp://ftp.rsasecurity.com/pub/otps/ct-kip/ct-kip-v1-0d2.pdf

[DHEKE] D. Jablon. Strong password-only authenticated key exchange. ACM Computer Communication Review, ACM SIGCOMM, 26(5):5-20, 1996

[DLOTA] “Generic Content Download Over The Air Specification Version 1.0”, Open Mobile Alliance, OMA-Download-OTA-v1_0, URL:http://www.openmobilealliance.org/

[GSM0348]: Digital cellular telecommunications system (Phase 2+); “Security Mechanisms for the SIM application toolkit”; (GSM 03.48 version 8.2.0 Release 1999)

[HOTP] D M’Raihi et. al, “HOTP : An HMAC-based One Time Password Algorithm”, http://www.ietf.org/internet-drafts/draft-mraihi-oath-hmac-otp-03.txt

[JAAS] “Java Authentication and Authorization Service”, http://java.sun.com/products/jaas/

[MIDPOTA] “JSR-000118 Mobile Information Device Profile 2.0”, Java Community Process, URL: http://jcp.org/aboutJava/communityprocess/final/jsr118/

[PAM] “Pluggable Authentication Module”, http://www.freebsd.org/doc/en_US.ISO8859-1/articles/pam/

[PKCS11] RSA Laboratories, “Cryptographic Token Interface Standard”,http://www.rsasecurity.com/rsalabs/pkcs/pkcs-11/index.html

[RFC2315]: B. Kaliski, “Cryptographic Message Syntax Version 1.5”, IETF, RFC2315, March 1998, http://www.faqs.org/rfcs/rfc2315.html

[RFC2616] “Hypertext Transfer Protocol – HTTP/1.1”. Network Working group, RFC2616. June 1999, http://www.ietf.org/rfc/rfc2616.txt

[RFC2986] M. Nystrom, B. Kaliski; “PKCS #10: Certification Request Syntax Specification version 1.7”, IETF RFC 2986; November 2000; URL:http://www.faqs.org/rfcs/rfc2986.html

[SAML] OASIS, “Security Assertions Markup Language”, http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=security

[SCEP] INTERNET DRAFT, X. Liu, C. Madson, D. McGrew, A. Nourse; “Cisco Systems' Simple Certificate Enrollment Protocol(SCEP) revised 11 Feb 2005; URL: http://www.ietf.org/internet-drafts/draft-nourse-scep-11.txt

[SPEKE]: Jablon, D., “Strong Password-Only Authenticated Key Exchange”, September 1996.

[TPM]: Trusted Computing Group, “Trusted Platform Module”, https://www.trustedcomputinggroup.org/downloads/specifications/

[WSS] OASIS, “Web Services Security”, http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-soap-message-security-1.0.pdf

[XKMS] W3C, “XML Key Management Specification (XKMS)”, 30 March 2001, http://www.w3.org/TR/xkms/

10. Terminology and Conventions

10.1 Abbreviations

AAA	Authentication, Authorization and Auditing
CA	Certificate Authority
CAPI	Microsoft CryptoAPI
CRL	Certificate Revocation List
CRM	Customer Relationship Management
CT-KIP	RSA Crypto Token Key Initialization Protocols
DLOTA	Download Over-the-Air (DLOTA) Protocol
DRM	Digital Rights Management
EAP	Extensible Authentication Protocol
EIS	Enterprise Information Systems
ERP	Enterprise Rights Management
GSM	Global System for Mobile Communications
GSS-API	Generic Security Service Application Program Interface
GUI	Graphical User Interface
HTTP	Hyper Text Transfer Protocol
HOTP	HMAC based OTP
IETF	Internet Engineering Task Force
I/O	Input/Output
IPSEC	Internet Protocol Security
ISP	Internet Service Provider
JAAS	Java Authentication and Authorization Service
LDAP	Lightweight Directory Access Protocol
MIDP	Mobile Information Device Profile
OATH	Initiative for Open AuTHentication
OCSP	Online Certificate Status Protocol
OMA	Open Mobile Alliance
OTA	Over the air
OTP	One Time Password
PAM	Pluggable Authentication Module
PAP	Password Authentication Protocol
PDA	Personal Data Assistant
PKCS	Public Key Cryptography Standards

PKI	Public Key Infrastructure
RADIUS	Remote Authentication Dial In User Service (RFC 2138)
RFC	IETF Request for Comments
SAML	Security Assertions Markup Language
SASL	Simple Authentication and Security Layer
SCEP	Simple Certificate Enrollment Protocol
SI	System Integrator
SIM	Subscriber Identity Module
SME	Small and Medium Enterprise
SMS	Short Message Service
SPEKE	Simple Password-authenticated Exponential Key Exchange
SSL	Secure Sockets Layer
TACACS	Terminal Access Controller Access Control System
TPM	Trusted Platform Module
TLS	Transport Layer Security
X-KISS	XML Key Information Service Specification
XKMS	XML Key Management Specification
X-KRSS	XML Key Registration Service Specification
USB	Universal Serial Bus
VoIP	Voice over Internet Protocol
VPN	Virtual Private Network

Contributors To This Paper

Eric LeSaint	ActivCard
Michael Williams	ActivCard
Ohad Ranen	Aladdin Knowledge Systems
Vladimir Becker	Aladdin Knowledge Systems
Eric Borcherding	Assa Abloy ITG
Philippe Guillaud	AudioSmartCard
Ernesto Frutos	Authenex
R. Boroughs	Aventail
Amol U. Deshmukh	Axalto
James A. McLaughlin	Axalto
Jeff Bohren	BMC Software
Rami Elron	BMC Software
Steve Anderson	BMC Software
Fred McClain	Boojum Mobile
Henri Quiniou	Citrix Systems
Hagai Bar-El	Discretix
Ophir Shalitin	Discretix
Salah Machani	Diversinet
Stu Vaeth	Diversinet
Eric Plet	Gemplus
George Robert Blakley	I.B.M.
Nathan Owen	I.B.M.
Frederic Engel	Livo Technologies
Logan Henriquez	LJM Systems
Kurt Lennartsson	RedCannon Security Inc.
H. Hassan	SafeNet
Laszlo Elteto	SafeNet
John Stewart	Signify
Liam Crilly	Signify
Ben Hodson	Venafi
Paul Turner	Venafi
Alex Deacon	VeriSign
David M'Raihi	VeriSign
Lawrence Ramontianu	VeriSign
Nico Popp	VeriSign
Siddharth Bajaj	VeriSign

initiative for open authentication