P2410™/D11
Draft Biometric Open Protocol Standard

Standards Development Board
of the
IEEE Communications Society

Technical Committee on RFID (CRFID)
of the
IEEE Technical Activities Board

Approved <Date Approved>

IEEE-SA Standards Board

Copyright © 2014 by The Institute of Electrical and Electronics Engineers, Inc.
Three Park Avenue
New York, New York 10016-5997, USA
All rights reserved.

This document is an unapproved draft of a proposed IEEE Standard. As such, this document is subject to change. USE AT YOUR OWN RISK! IEEE copyright statements SHALL NOT BE REMOVED from draft or approved IEEE standards, or modified in any way. Because this is an unapproved draft, this document shall not be utilized for any conformance/compliance purposes. Permission is hereby granted for officers from each IEEE Standards Working Group or Committee to reproduce the draft document developed by that Working Group for purposes of international standardization consideration. IEEE Standards Department shall be informed of the submission for consideration prior to any reproduction for international standardization consideration (stds.ipr@ieee.org). Prior to adoption of this document, in whole or in part, by another standards development organization, permission shall first be obtained from the IEEE Standards Department (stds.ipr@ieee.org). When requesting permission, IEEE Standards Department will require a copy of the standard development organization's document highlighting the use of IEEE content. Other entities seeking permission to reproduce this document, in whole or in part, shall also obtain permission from the IEEE Standards Department.

IEEE Standards Department
445 Hoes Lane
Piscataway, NJ 08854, USA
Abstract: Biometric Open Protocol Standard (BOPS) provides Identity Assertion, Role Gathering, Multi-Level Access Control, Assurance, and Auditing. The BOPS implementation includes software running on a client device (Android, iPhone, etc.), a trusted BOPS Server, and an Intrusion Detection (IDS) system. The BOPS implementation allows pluggable components to replace existing components functionality accepting integration into current operating environments in a short period of time. The BOPS implementation provides continuous protection to the resources and assurance of the placement and viability of adjudication, and other key features. Accountability is the mechanism that proves a service level guarantee of security. The BOPS implementation allows the systems to meet security needs by using the API. The BOPS implementation need not know whether the underlying system is a Relational Database Management System (RDBMS) or a Search Engine. The BOPS implementation functionality offers a “point and cut” mechanism to add the appropriate security to the production systems as well as to the systems in development. The architecture is a language neutral allowing REST, JSON and Secure Socket Layers (SSL) or Transport Layer Security (TLS) to provide the communication interface. The architecture is built on the servlet specification, open SSLs, Java, JSON, REST and an open persistent store. Each and every tool adheres to the open standards allowing maximum interoperability.

Keywords: admin console, application, BOPS admin, BOPS cluster, BOPS server, BOPS IDS, client device IDS, Jena Rules, IDS cluster, IEEE 2410, liveness, original site admin, site admin, trusted adjudicated data, user, user device
Important Notices and Disclaimers Concerning IEEE Standards Documents

IEEE documents are made available for use subject to important notices and legal disclaimers. These notices and disclaimers, or a reference to this page, appear in all standards and may be found under the heading “Important Notice” or “Important Notices and Disclaimers Concerning IEEE Standards Documents.”

Notice and Disclaimer of Liability Concerning the Use of IEEE Standards Documents

IEEE Standards documents (standards, recommended practices, and guides), both full-use and trial-use, are developed within IEEE Societies and the Standards Coordinating Committees of the IEEE Standards Association (“IEEE-SA”) Standards Board. IEEE (“the Institute”) develops its standards through a consensus development process, approved by the American National Standards Institute (“ANSI”), which brings together volunteers representing varied viewpoints and interests to achieve the final product. Volunteers are not necessarily members of the Institute and participate without compensation from IEEE. While IEEE administers the process and establishes rules to promote fairness in the consensus development process, IEEE does not independently evaluate, test, or verify the accuracy of any of the information or the soundness of any judgments contained in its standards.

IEEE does not warrant or represent the accuracy or content of the material contained in its standards, and expressly disclaims all warranties (express, implied and statutory) not included in this or any other document relating to the standard, including, but not limited to, the warranties of: merchantability; fitness for a particular purpose; non-infringement; and quality, accuracy, effectiveness, currency, or completeness of material. In addition, IEEE disclaims any and all conditions relating to: results; and workmanlike effort. IEEE standards documents are supplied “AS IS” and “WITH ALL FAULTS.”

Use of an IEEE standard is wholly voluntary. The existence of an IEEE standard does not imply that there are no other ways to produce, test, measure, purchase, market, or provide other goods and services related to the scope of the IEEE standard. Furthermore, the viewpoint expressed at the time a standard is approved and issued is subject to change brought about through developments in the state of the art and comments received from users of the standard.

In publishing and making its standards available, IEEE is not suggesting or rendering professional or other services for, or on behalf of, any person or entity nor is IEEE undertaking to perform any duty owed by any other person or entity to another. Any person utilizing any IEEE Standards document, should rely upon his or her own independent judgment in the exercise of reasonable care in any given circumstances or, as appropriate, seek the advice of a competent professional in determining the appropriateness of a given IEEE standard.

IN NO EVENT SHALL IEEE BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO: PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE PUBLICATION, USE OF, OR RELIANCE UPON ANY STANDARD, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE AND REGARDLESS OF WHETHER SUCH DAMAGE WAS FORESEEABLE.

Translations

The IEEE consensus development process involves the review of documents in English only. In the event that an IEEE standard is translated, only the English version published by IEEE should be considered the approved IEEE standard.
Official statements

A statement, written or oral, that is not processed in accordance with the IEEE-SA Standards Board Operations Manual shall not be considered or inferred to be the official position of IEEE or any of its committees and shall not be considered to be, or be relied upon as, a formal position of IEEE. At lectures, symposia, seminars, or educational courses, an individual presenting information on IEEE standards shall make it clear that his or her views should be considered the personal views of that individual rather than the formal position of IEEE.

Comments on standards

Comments for revision of IEEE Standards documents are welcome from any interested party, regardless of membership affiliation with IEEE. However, IEEE does not provide consulting information or advice pertaining to IEEE Standards documents. Suggestions for changes in documents should be in the form of a proposed change of text, together with appropriate supporting comments. Since IEEE standards represent a consensus of concerned interests, it is important that any responses to comments and questions also receive the concurrence of a balance of interests. For this reason, IEEE and the members of its societies and Standards Coordinating Committees are not able to provide an instant response to comments or questions except in those cases where the matter has previously been addressed. For the same reason, IEEE does not respond to interpretation requests. Any person who would like to participate in revisions to an IEEE standard is welcome to join the relevant IEEE working group.

Comments on standards should be submitted to the following address:

Secretary, IEEE-SA Standards Board
445 Hoes Lane
Piscataway, NJ 08854 USA

Laws and regulations

Users of IEEE Standards documents should consult all applicable laws and regulations. Compliance with the provisions of any IEEE Standards document does not imply compliance to any applicable regulatory requirements. Implementers of the standard are responsible for observing or referring to the applicable regulatory requirements. IEEE does not, by the publication of its standards, intend to urge action that is not in compliance with applicable laws, and these documents may not be construed as doing so.

Copyrights

IEEE draft and approved standards are copyrighted by IEEE under U.S. and international copyright laws. They are made available by IEEE and are adopted for a wide variety of both public and private uses. These include both use, by reference, in laws and regulations, and use in private self-regulation, standardization, and the promotion of engineering practices and methods. By making these documents available for use and adoption by public authorities and private users, IEEE does not waive any rights in copyright to the documents.

Photocopies

Subject to payment of the appropriate fee, IEEE will grant users a limited, non-exclusive license to photocopy portions of any individual standard for company or organizational internal use or individual, non-commercial use only. To arrange for payment of licensing fees, please contact Copyright Clearance Center, Customer Service, 222 Rosewood Drive, Danvers, MA 01923 USA; +1 978 750 8400. Permission to photocopy portions of any individual standard for educational classroom use can also be obtained through the Copyright Clearance Center.
Updating of IEEE Standards documents

Users of IEEE Standards documents should be aware that these documents may be superseded at any time by the issuance of new editions or may be amended from time to time through the issuance of amendments, corrigenda, or errata. An official IEEE document at any point in time consists of the current edition of the document together with any amendments, corrigenda, or errata then in effect.

Every IEEE standard is subjected to review at least every ten years. When a document is more than ten years old and has not undergone a revision process, it is reasonable to conclude that its contents, although still of some value, do not wholly reflect the present state of the art. Users are cautioned to check to determine that they have the latest edition of any IEEE standard.

In order to determine whether a given document is the current edition and whether it has been amended through the issuance of amendments, corrigenda, or errata, visit the IEEE-SA Website at http://ieeexplore.ieee.org/xpl/standards.jsp or contact IEEE at the address listed previously. For more information about the IEEE-SA or IEEE’s standards development process, visit the IEEE-SA Website at http://standards.ieee.org.

Errata

Errata, if any, for all IEEE standards can be accessed on the IEEE-SA Website at the following URL: http://standards.ieee.org/findstds/errata/index.html. Users are encouraged to check this URL for errata periodically.

Patents

Attention is called to the possibility that implementation of this standard may require use of subject matter covered by patent rights. By publication of this standard, no position is taken by the IEEE with respect to the existence or validity of any patent rights in connection therewith. If a patent holder or patent applicant has filed a statement of assurance via an Accepted Letter of Assurance, then the statement is listed on the IEEE-SA Website at http://standards.ieee.org/about/sasb/patcom/patents.html. Letters of Assurance may indicate whether the Submitter is willing or unwilling to grant licenses under patent rights without compensation or under reasonable rates, with reasonable terms and conditions that are demonstrably free of any unfair discrimination to applicants desiring to obtain such licenses.

Essential Patent Claims may exist for which a Letter of Assurance has not been received. The IEEE is not responsible for identifying Essential Patent Claims for which a license may be required, for conducting inquiries into the legal validity or scope of Patents Claims, or determining whether any licensing terms or conditions provided in connection with submission of a Letter of Assurance, if any, or in any licensing agreements are reasonable or non-discriminatory. Users of this standard are expressly advised that determination of the validity of any patent rights, and the risk of infringement of such rights, is entirely their own responsibility. Further information may be obtained from the IEEE Standards Association.
Participants

At the time this draft standard was completed, the Biometrics Open Protocol Working Group had the following membership:

Scott Streit, Chair
Jack Callahan, Vice Chair
Will Lumpkins, Secretary
Brad Boyer, Technical Editor

Participant1
Participant2
Participant3
Participant4
Participant5
Participant6
Participant7
Participant8
Participant9

The following members of the <individual/entity> balloting committee voted on this standard. Balloters may have voted for approval, disapproval, or abstention.

Balloter1
Balloter2
Balloter3
Balloter4
Balloter5
Balloter6
Balloter7
Balloter8
Balloter9

When the IEEE-SA Standards Board approved this standard on <Date Approved>, it had the following membership:

John Kulick, Chair
Jon Walter Rosdahl, Vice Chair
Richard H. Hulett, Past Chair
Konstantinos Karachalios, Secretary

Also included are the following nonvoting IEEE-SA Standards Board liaisons:

Richard DeBlasio, DOE Representative
Michael Janezic, NIST Representative

IEEE-SA Publishing

Tricia Gerdon
Jonathan Goldberg

IEEE-SA Standards Technical Community

Copyright © 2014 IEEE. All rights reserved.
This is an unapproved IEEE Standards Draft, subject to change.
Introduction

Convenience drives consumers toward the biometrics based access management solutions, say studies from Ericsson, PayPal, IBM, and Microsoft.

According to the Ericsson’s study “Your body is the new password”, 52 percent of smartphone users want to use their fingerprints instead of the passwords, a further 61 percent want to use fingerprints to unlock their phones, and 48 percent want to use eye-recognition.

The study conducted by PayPal says that consumers approve biometrics for access management. In terms of readiness to switch from an old fashion password protection to the new technology, 53 percent of surveyed population would be comfortable replacing passwords with the fingerprints and 45 percent would choose a retinal scan, which is presumably an iris scan – the misplaced terminology points to the lack of a consumer education.

IBM Fellow and Speech CTO David Nahamoo states that over the next five years, your unique biological identity and biometric data – facial definitions, iris scans, voice files, even your DNA – will become the key to the safeguarding of your personal identity and information and will replace the current user ID and password system.

Microsoft Research funded a study that titled “The Quest to Replace Passwords: A Framework for Comparative Evaluation of Web Authentication Schemes”, the cornerstone conclusion of which indicates that the vast passwords replacement transition should conform to the following criteria: nothing to carry, efficient to use, and easy recovery from a loss. The Microsoft study goes as far as concluding such criteria could be achieved mostly in the biometric schemes.

Biometric technologies provide consumer with a long-awaited convenience to securely enter into the cyberspace on the frontend. The Biometric Open Standards Protocol (BOPS), developed by Hoyos Labs, protects digital assets and digital identities on the backend.

BOPS is a biometrics agnostic standard that opens an API for the registered developers. Entering as a game-changer, BOPS communication architecture enables 2-way Secure Socket Layer (SSL or Transport Layer Security (TLS)) connection over the encryption mechanism to the server, which employs Intrusion Detection System (IDS). The IDS is an external system responsible for blacklisting devices that are violating the replay violating the replay portion of this specification.
Contents

1. Overview .................................................................................................................. 1
   1.1 Scope ...................................................................................................................... 1
   1.2 Purpose ................................................................................................................... 2
   1.3 Intended audience ................................................................................................. 2

2. Normative references ................................................................................................. 2

3. Definitions, acronyms and abbreviations .................................................................. 2
   3.1 Definitions .............................................................................................................. 2
   3.2 Acronyms and abbreviations .................................................................................. 3

4. Conformance .............................................................................................................. 4

5. Security considerations ............................................................................................. 4
   5.1 Background ............................................................................................................ 4
   5.2 Identification assertion ......................................................................................... 4
   5.3 Role gathering ....................................................................................................... 4
   5.4 Access control ...................................................................................................... 5
   5.5 Audit and assurance .............................................................................................. 5

6. BOPS interoperability ............................................................................................... 6

7. BOPS overview, application, registration, and prevention of replay ......................... 6
   7.1 Overview ............................................................................................................... 6
   7.2 Application .......................................................................................................... 9
   7.3 Registration ......................................................................................................... 10
   7.4 Prevention of replay ........................................................................................... 12

8. BOPS API overview .................................................................................................. 13
   8.1 Identity assertion API ........................................................................................... 13

9. API ............................................................................................................................ 14
   9.1 Enterprise concepts ............................................................................................... 14
   9.2 Format of API cells .............................................................................................. 14
   9.3 Genesis ................................................................................................................ 14
   9.4 API—Genesis ...................................................................................................... 15
   9.5 API—QROpportunity ......................................................................................... 16
   9.6 Role API ............................................................................................................. 19
   9.7 Access control API ............................................................................................ 20
   9.8 Auditing .............................................................................................................. 21
   9.9 Administration ..................................................................................................... 22
   9.10 Reporting .......................................................................................................... 23

10. Client device requirements ....................................................................................... 23

11. Server-side intrusion detection system ..................................................................... 24
   11.1 API list blacklist ............................................................................................... 24
   11.2 API—Incident .................................................................................................... 25

Annex A (informative) Glossary ..................................................................................... 26
Draft Standard for Biometric Open Protocol

1. Overview

1.1 Scope

The Biometric Open Protocol Standard (BOPS) provides Identity Assertion, Role Gathering, Multi-Level Access Control, Assurance, and Auditing. The BOPS implementation includes software running on a client device (e.g., smartphone or mobile device), a trusted BOPS Server, and Intrusion Detection System. The BOPS implementation allows pluggable components to replace existing components functionality accepting integration into the current operating environments in a short period of time. The BOPS implementation adheres to the principle of continuous protection in adjudicating access to resources. Accountability is the mechanism that proves a service level guarantee of security. The BOPS implementation allows the systems to meet security needs by using the API (Application Programming Interface). The BOPS implementation need not know whether the underlying system is a Relational Database Management System (RDBMS) or a Search Engine. The BOPS implementation functionality offers a “point-and-cut” mechanism to add the appropriate security to the production systems as well as to the systems in development. It is assumed the Extended Network Time Protocol (XNTP) is used on all BOPS Server for timing coordination.
1.2 Purpose

This standard provides a biometric agnostic multi-level security protocol.

1.3 Intended audience

The intended audience of this document includes security evaluators, system underwriters, developers, and systems engineers. The Biometric Open Protocol Standard is a subject to changes and updates.

2. Normative references


3. Definitions, acronyms and abbreviations

3.1 Definitions

For the purposes of this document, the following terms and definitions apply. The IEEE Standards Dictionary Online should be consulted for terms not defined in this clause. 

---

account: A user account which was validated (against an external system or by email validation mechanism). It can have associate one or multiple Mobile Devices. The enrollment process ends by creating a Client Certificate for the device that shall be used for subsequent calls to authenticate against the Hoyos ID Platform.

Bell-LaPadula: The multilevel model that was proposed by Bell and LaPadula for enforcing access control in government and military applications. A subject can only access objects at certain levels determined by his security level.

JUnit: A testing framework for Java programming language.

RESTful: Refers to REST Representational State Transfer, which is an architectural style.

SHA1: Secure hash algorithm one, which was designed by the US NSA.

3.2 Acronyms and abbreviations

AOP Aspect Oriented Programming

API Application Programming Interface

App A mobile client application

BOPS Biometric Open Protocol Standards

CPU Central Processing Unit

CSRF Cross-site request forgery

IDS Intrusion Detection System

IP Internet Protocol

JSON JavaScript Object Notation

MAC Mandatory Access Control

RDBMS Relational Database Management System

SSL Secure Socket Lavers

TCSEC Trusted Computer System Evaluation Criteria

URL Uniform Resource Identifier

QR Code Quick Responses Code
Conformance

The Biometric Open Protocol Standard is comprised of the rules governing secure communication between a variety of client devices and the trusted server. This standard is based on the tested computer based implementation of the Trusted Computer System Evaluation Criteria (TCSEC).²

BOPS conforms to the Trusted Computer System Evaluation Criteria (TCSEC), which is the US Government Department of Defense (DoD) standard that sets basic requirements for assessing the effectiveness of computer security controls built into a computer system, created by the National Computer Security Center (NCSC), an arm of the National Security Agency (NSA) and is also frequently referred as Orange Book, section B1; to the Director of the Central Intelligence Directive 6/3 protection levels 3, 4, and 5 (PL3, PL4, PL5); and to the standards of Multiple Independent Levels of Security (MILS).

4. Security considerations

4.1 Background

Security considerations include the security policies in place and unambiguously defined levels of security. One of the BOPS main functions is to provide Authentication instead of Authorization in a way where the server does not retain the client information, but rather recognizes one client from another. The key components of security considerations include Identity Assertion, Role Gathering, Access Control, Auditing, and Assurance.

4.2 Identification Assertion

The BOPS implementation helps provide continuous protection to resources and assurance of the placement and viability of adjudication and other key features. Accountability is the mechanism that can help provide a service level guarantee of security.

The BOPS implementation identity assertion helps confirm named users are who they claim to be. The identity assertion implies reliance on human biometrics, however, the BOPS is an interoperable standard and can incorporate any identity asserter, or a number of asserters associated with a named user. The application of the IDS provides active monitoring to help prevent spoofing of the credentials set and blacklisting of a subject or device that makes malicious attempts.

4.3 Role gathering

Role gathering is focused on the data confidentiality and privileged access based on the rules enforced by known system. To determine whether a specific access mode is allowed, the privilege of a role is compared to the classification of the group to determine if the subject is authorized for a confidential access. The objects structure is defined by the access control. Role gathering occurs on the system’s level or through the client/server call. The BOPS implementation server stores role gathering information to associate a unique user with a unique device.

² Hoyos Lab (software HoyosID™), located at MOMA Building, 25 W 53rd St., 14th Floor, New York, NY 10019 originated and built BOPS to gauge the interoperability of biometric products.
4.4 Access Control

4.4.1 General

Access control asks whether a given subject (person, device or program) may read, write, execute, or delete a given object. The community further divides access control into Discretionary Access Control and a more granular form of access control called Mandatory Access Control (MAC).

4.4.2 Discretionary Access Control

The BOPS implementation supports access control between the named users and the named objects (e.g., files and programs). The adjudication mechanism is role based and allows users and administrators to specify and control sharing of those objects by named individuals and/or defined groups of individuals. The discretionary access control mechanism provides that objects are protected from unauthorized access. Discretionary access control provides protection at the group or individual level across singular or group of objection. The granularity ranges from individual to group.

4.4.3 Mandatory access control

The BOPS implementation shall enforce a mandatory access control policy over all subjects and storage objects under its control (e.g., processes, files, segments, devices). These subjects and objects are assigned sensitivity labels, which are a combination of hierarchical classification levels and non-hierarchical categories. The labels are used in the adjudication as the basis for mandatory access control decisions. The client software shall maintain labels or have the BOPS server maintain the data, which forces adherence to labeling of the subject and objects. The BOPS implementation server maintains a trusted store as a component of BOPS.

The following requirements hold all accesses between subjects and objects controlled by the BOPS: a subject can read an object only if the hierarchical classification in the subject’s security level is greater than or equal to the hierarchical classification in the object’s security level. The non-hierarchical categories in the subject’s security level include all the non-hierarchical categories in the object’s security level. A subject can write an object only if the hierarchical classification in the subject’s security level is less than or equal to the hierarchical classification in the object’s security level and all the non-hierarchical categories in the subject’s security level are included in the non-hierarchical categories in the object’s security level. Identification and authentication data should be used by BOPS to authenticate the user’s identity and to maximize that the security level and authorization of subjects external to the BOPS that may be created to act on behalf of the individual user are dominated by the clearance and authorization of that user.

4.5 Audit and assurance

4.5.1 Background

The worst possible case is a system is compromised and goes undetected. To prevent this case, the aforementioned specifications rightly require auditing and proof of the security model, which is called assurance.
4.5.2 Audit

The BOPS implementation supports all auditing requests at the Subject/Object level or at the group level. The BOPS implementation uses Aspect Oriented Programming (AOP) to maximize the likelihood that all calls are safely written to an audit trail. A RESTful web services and JavaScript Object Notation interface provides a mechanism to read the audit trail. Auditing may occur at the subject per action, the object per action or the group per action. For example, a group of users called “Accounting” may audit all writes to General Ledger. Or, the “Chief Financial Officer” may have audits for reads of the Income Statement.

4.5.3 System integrity

The JUnit tests exist for all boundary conditions of the BOPS. The suite of tests includes testing all boundary components and conditions of the system.

5. BOPS interoperability

The BOPS implementation allows the systems to meet security needs by using the API. The BOPS implementation does not need to know whether the underlying system is a Relational Database Management System (RDBMS) or a Search Engine. The BOPS implementation functionality offers a “point and cut” mechanism to add the appropriate security to the production systems as well as to the systems in development. The architecture is a language neutral allowing REST, JSON and Secure Socket Layers to provide the communication interface. The architecture is built on the servlet specification, open secure socket layers, Java, JSON, REST and a persistent store. Each and every tool adheres to open standards allowing maximum interoperability as shown in figure 1.

6. BOPS Overview, Application, Registration, and Prevention of Replay

6.1 Overview

The BOPS implementation provides Identity Assertion, Role Gathering, Multi-Level Access Control, Assurance and Auditing. The BOPS implementation includes software running on a client device (Android, iPhone, etc.), a trusted BOPS Server, and the Intrusion Detection System (IDS). The BOPS implementation allows pluggable components to replace existing components functionality accepting integration into current operating environments in a short period of time.

The client/device application loads a one-time 2-way SSL/TLS key for the initial communication to the server. This key is replaced, in function, by the subjects 2-way SSL/TLS key during the identityGenesis phase. Identity Genesis, is an initial step and fuses a set of biometrics with a given subject.

The client/server application interaction with BOPS could be described as a three steps process with two possible variations after the first step. For the purposes of this document, the BOPS Client/Server Application architecture is presented in three components: client application, the BOPS security server, and the server-side application (App Server on the following diagrams).

As shown below in Figure 2 through Figure 6, the most important part of the BOPS Client/Server Application connection is the server-side application is not running through the BOPS server, and the SSL/TLS connection terminates at the application server. The BOPS implementation deployment doesn’t require the application to trust the BOPS system with the unencrypted content.
During Genesis, the client makes a call to the BOPS server, authenticates the user, and client-side application software, then receives, allocated by the BOPS server, a certificate, which is specific to the client identity of a specific application.

During the next step (Figure 3), the client device/app calls the app server directly; the BOPS server is not on this path. The SSL/TLS connection between client and server parts of the application starts and terminates at these points. All content exchange is not visible outside of the application to the BOPS server or others not trusted within this application entity.
During the client session the app server calls the BOPS server to get identification details and confirms the certificate has not been revoked already.

In the variation flow the first step is the same: Genesis. After the first step is complete, the BOPS server contacts the application server component to notify that a new client has been registered and allocated.

This flow (Figure 5) differs from the previously described flow (Figure 3) in a matter of the identity details, and revocation check is procured in the client session (Figure 6).
At the third step, when the client calls the application directly, the application server calls the BOPS server to check if the certificate has not been revoked yet.

### 6.2 Application

For example, the entire BOPS suite could be used through the access control or simply added to identity assertion of already existing framework. The BOPS implementation enables trusted processing by performing the minimum actions in the production environment and in the most cases does not require the change of any application software.

2-way SSL/TLS, which is built on a top of 1-way SSL/TLS, provides communication starting at the client. The initial or Genesis communication establishes the origin of the client’s identity and passes the BOPS compliant 2-way certificate that the client uses for subsequent communication in conjunction with the session oriented Identity Assertion. It is important to note that the client application shall have a pre-loaded 2-way SSL/TLS key that allows subsequent identityGenesis.

The BOPS implementation compliant server receives 1-way SSL/TLS communication with 2-way SSL/TLS identity. Communication is conducted both 1-way SSL/TLS and 2-way SSL/TLS. The server uses a data store to take trusted identity and gather the roles for processing on behalf of the identity. Auditing maximizes the appropriate artifacts for continued verification and validation of the trusted access. The Assurance occurs through the simplification and documentation of the multi-level access control mechanism. The BOPS implementation requires an Administration Console, which is available after the registration process (See section 7.2 BOPS Registration), which allows dynamic modification of Users, Groups, and Roles.

BOPS shall be implemented with an active Intrusion Detection System that provides prevention of any form of brute-forcing or denial-of-service (distributed or single Denial of Service) attacks. The standard contains a custom rule that identifies and tracks the attempts to forge 2-way SSL/TLS certificates impersonation, a session replay, forged packets, and variety of other attempts to circumvent the BOPS server. See figure 7.
6.3 Registration

6.3.1 Background

The registration process initiates the BOPS adoption within an organization. BOPS may appear as a cluster, but is considered a business component. Before the BOPS admin sets up an environment, the organization shall register for an API key at the BOPS Server. The individual developers may apply for the API key as well.

At enrolment completion, the originalSiteAdmin may create additional site administrators. In the future, the enrollment information shall be associated with the API key of the organization. The API registration pertains two domains: the enrolled original site admin and the issued API key, which is based on the enrollment information, the organization, and use case. The registration process is complete when the application commencement is agreed. After, the BOPS admin creates originalSiteAdmin for an organization; the original site admin may create a site admin (**Figure 8**). The steps after the registration are described in the section 5 of this document.
6.3.2 Developers and the BOPS service

Prior to the development process that utilizes the BOPS service, a developer needs to be registered in the BOPS Admin Console. By providing the application name and using Axiom to identify the developer, the BOPS establishes a new account and creates the API key, which would be identified with the application name and associated with the application.

6.3.3 The end user and the BOPS service

The communication between the application and the BOPS server is established on the top of 2-way SSL/TLS. Genesis is a mechanism that establishes this connection. Genesis specifies how the users identify themselves to the BOPS server, so the server would generate a private key to set up the 2-way SSL/TLS communication between the application and BOPS server. Axiom is one of the mechanisms that BOPS uses to identify users.

6.3.4 The end user device and the BOPS service

The application is responsible for providing a unique ID that identifies the device of the end user. The application uses the device association API to notify the BOPS server about the link between the user and the user’s device.
6.4 Prevention of replay

6.4.1 The format of all requests

This example demonstrates communication between a client and a server.

An important note: All calls to the BOPS server have an API call with the notable exception of CreateApplication, which actually creates the API Key. The request is coming from a client device to:

Example: https://xyz.domain.com:8443/{BOPS_Instance_Server}/

The initial call receives a 2-way SSL/TLS certificate and creates a user. The user is created in a clustered persistent environment. The sums that prevent playback are assumed as one-way encrypted using SHA1. Switching SHA1 with any suitable algorithm does not change the algorithm. For the duration of this document, assume SHA1.

For val1=<n1>, n1 is a SHA1 sum of an integer between -59 and 59 added or subtracted from the current time in ISO-8601 format. The server is required to use the xntp protocol for timing. For val2=<n2>, n2 is a SHA1 sum of an integer between -59 and 59 and is greater than the plaintext value of n1. The values for usernames and passwords are client dependent and used to reach the current identity asserter. Similar mechanisms exist for Security Assertion Markup Language, Lightweight Directory Protocol, Active Directory in conjunction with a variety of mechanism for Asserting Identity and Role Gathering.

The initial call would be to:

https://xyz.domain.com:8443/{BOPS_Instance_Server}/?val1=<n1>&val2=<n2>&siteId=<client>&username=<username>&password=<password>&newPassword=<newpassword>

Let’s examine the consequence of our Genesis Request:

<table>
<thead>
<tr>
<th>username</th>
<th>email</th>
<th>val1</th>
<th>val2</th>
<th>sessionTimeout</th>
<th>roles</th>
<th>siteld</th>
</tr>
</thead>
<tbody>
<tr>
<td>scott</td>
<td><a href="mailto:scott@scottstreit.com">scott@scottstreit.com</a></td>
<td>5</td>
<td>40</td>
<td>3600</td>
<td>Admin</td>
<td>businessCustomer</td>
</tr>
</tbody>
</table>

The user *scott* has an email *scott@scottstreit.com*. The first replay value in a plain text is 5 and the second is 40. The *sessionTimeout* exists at the *sessionId*, *siteld* pairing. For an administrator of the business customer website the *sessionTimeout* exists for one hour. The session timeout is defined on a per client basis.

In the greater detail the example works as follows, with the current time as 2013-12-22T17:46:03.647Z. move back to the five-minute interval and get

2013-12-22T17:45:03.647Z with an SHA1 sum of 28b70156eaaf38767f80c876b948285d7f570e4a

Example:
https://xyz.domain.com:8443/{BOPS_Instance_Server}/genesisval1=fa8e14cf7f80f885084ee0e3cb256182
bb6a4e40&val2=fa8e14cf7f80f885084ee0e3cb256182bb6a4e40&newPassword=gasol

The values associated with *val1* is fa8e14cf7f80f885084ee0e3cb256182bb6a4e40 is a 5 offset and for *val2*=fa8e14cf7f80f885084ee0e3cb256182bb6a4e40 which happens to be the same for 40. The *newPassword* is the password for the 2-way SSL/TLS key, which is never going to be stored on the BOPS
server. To execute this operation the BOPS Server shall have the SHA1 sum for all integers between -59 and 59 to decipher the sums.

6.4.2 Subsequent API calls

For instance, at 2:18pm Zulu time user scott uses a client device (Android phone) to create a session. The call contains deviceId for a session, as well as the following parameters:

---

val1=<SHA1sum of current time rolled back to the nearest 5 minute interval>&

---

val2=<SHA1sum of current time rolled forward to the nearest 20 minute interval>&command=<SHA1sum of an a low level operating system such as fopen>&version=<version of command>&val3=<SHA1sum for the command file>

To prevent the replay of a previous session or a replacement the key kernel object files, the BOPS server shall contain SHA1 sums for commands names and the files on a version-by-version basis. Using the BOPS protocol in a conjunction with the BOPS intrusion detection system prevents the replay attack. The Intrusion Detection System (IDS) updates the list of the blacklisted objects, as threats and attacks, on the further attack recognition level.

7. BOPS API overview

(All API names are in the RESTful JavaScript Object Notation (JSON) format.)

7.1 Identity Assertion API

7.1.1 Developer API Key

Individual developers need to apply for an API Key for their applications that shall use BOPS. This shall be done in the Developer Center. Once individual developers have their own API Key, all the APIs call their applications make, shall need to insert this API Key as one of parameters. The Identity Assertion Platform shall verify the API Key at that API level.

7.1.2 Application Identification

Application Identification creates an application for use by a development team. The definition of the Application Creation process is provided below.

Once application commencement is agreed, the overall BOPS Admin creates a user with the special role of originalSiteAdmin. Once the original site admin exists, the first action of the person with the originalSiteAdmin role associates their biometrics with identity. Subsequently, all actions have genesis and API.

Additionally, the originalSiteAdmin role may create users of a siteAdmin role. The siteAdmin role is used for an additional administration work.

Some terms are required to further understand the API.
8. API

8.1 Enterprise concepts

In the system can be registered one or multiple integrations. That is named Members data. Special data registered in this chunk of data is LoginDefinition field where it is described the credentials format for a certain integration.

When a user register a profile (MemberProfile), basically the system authenticates the user profile / credentials against the external system and associates the external user profile with his Account.

Once the user is enrolled and has a profile for a certain enterprise integration registered, (s)he can authenticate against the external system. That means, the user can attach to a session, which was initiated (Session Opportunity), authenticates on mobile against registered biometric data (saved during enrolment process) and send the result to the backend.

8.2 Format of API cells

Each call receives the input parameters in JSON format and produce result in JSON format too.

Each response contains the Error information (error JSON field)

```json
ErrorResponse {
  errorDescription (string, optional),
  errorCode (integer, optional)
}
```

eerrorCode = 0 means success. Otherwise, an error occurred and description of the error lays in errorDescription field.

8.3 Genesis

8.3.1 Overview

The start of the process is Genesis aka Enrollment. This fuses an initial identity with an individual. To do this, the device uses a 1 time 2-Way Secure Socket Layers connection. The device offers to the BOPS Server, any and all information uniquely identifying the device and this information is sent to a separate and distinct BOPS server, which only responds to Genesis requests. At this point, the BOPS Server responds with a variety of information.

The server responds with a 2-Way SSL/TLS Key containing identity, a password for encryption and decryption, and a set of values, which prevent replay. This is the standard 2-way SSL/TLS communication with the password requested by the device and the very same password used for passivated encryption. This
is called key the BOPS key. The transport layer of encryption uses 571 bit Elliptic Curve Encryption. The passivated data on the client device is encrypted also using 571 bit Elliptic Curve Encryption. This passivated encrypted data is all data stored on the device for initial or subsequent use.

As far as storage, on the client (device) is an encrypted version of the biometrics and the 2-Way SSL/TLS key which offers the possible identity on the device. The matching and liveness tell us if the identity matches the genesis identity. The device has no key generation artifacts and all information is encrypted when stored (passivated) on the device.

On the server, the device information that relates to a particular device. The key generation is from the server and never stored on the server. It is encrypted with the password only known by the client and also never stored on the servers. The server maintains identity information and no keys or other artifacts are stored.

### 8.3.2 Post-genesis communication

Post-genesis communication uses replay values, encrypted with a checksum and sent to the server. The BOPS implementation Server sends any failed attempts to the Intrusion Detection System, which is viewed in the following way. The key, as part of 2-way SSL/TLS, tells the device the possible identity. The matching, liveness and replay algorithms authenticate by checking and fusing the identity with Genesis.

### 8.3.3 Is the device blacklisted?

The BOPS implementation Server asks the Intrusion Detection System (IDS) if a device is blacklisted or not. If the device is blacklisted, all communication ceases. Every failed replay prevention value1 and replay prevention value2 result in machine learning logic on the IDS.

The intrusion detection system uses machine learning through Semantic Web in determining how broad of an area to blacklist. Blacklisting may occur at the device level, the Internal Protocol address level, the subnet level or beyond. This multi-tiered approach prevents trial and error attacks and restricts bad intention attempts. Semantic Web is defined as version of technology that implements the Web Ontology Language.

The Enterprise solution offers a mechanism to enroll a profile using the validation against the external system. This can be done at backend integration level. After the genesis call happens, BOPS shall respond to mobile application with client certificate which identifies unique that device. All the subsequent calls shall use the client certificate to authenticate against BOPS server.

### 8.4 API—Genesis

The genesis service is the initial setup for a user. It can work with the already stored initial identities or use an external axiom services for the initial identity.

**/RegisterAccount**

```plaintext
Request {
    name (string): Name of device,
    info (string, optional): Details information of the device,
}```
8.5 API—QROpportunity

8.5.1 Overview

One of the application authentication flows is using Quick Responses codes. On the business partner login page, it shall display a QR Code image that shall contain a Session Opportunity Identifier Mobile Application should start an authentication wizard, which shall scan the QR Code, register the session to signal that it is attached to the session and authenticate with his/her biometrics. After biometric authentication, it should send to Backend the result of authentication. See figure 9.

8.5.2/QROpportunity

Request {
  name (string): Name of device,
  info (string, optional): Details information of the device,
  val1 (string),
  val2 (string)
}

Response {
  qrImage (string): Base64 encoded QR Code based on a random number,
Biometric authentication should send to Backend the result of authentication.

```plaintext
error (ErrorResponse, optional)

ErrorResponse {
  errorCode (integer, optional),
  errorDescription (string, optional)
}

Biometric authentication should send to Backend the result of authentication.

Scan Activity

Register Session

Biometric Authentication

Send Authentication Result to Backend

Figure 9—Mobile application authentication flow

8.5.3 /enterprise/RegisterSession Opportunity

8.5.3.1 Input

Request {
  memberExternalId (string): External Identifier of enterprise integration (e.g. ‘garanti’),
}

8.5.3.2 Output

Response {
  sessionQrImage (string): Base64 encoded image with session informations,
  sessionId (string): Session Internal Identifier,
  error (ErrorResponse, optional)
}

ErrorResponse {
  errorCode (integer, optional),
  errorDescription (string, optional)
}
8.5.4 /enterprise/GetSessionStatus

8.5.4.1 Input

GetSessionStatusRequest {
  sessionId (string, optional)
}

8.5.4.2 Output

SessionResponse {
  status (string): Status of session, OPPORTUNITY, CREATED, AUTHENTICATED, FAILED, TIMEOUT, COMPLETED, CANCELED
  sessionId (string): Session Internal Id,
  data (Map[string, Object]): Extra values attached to session instance,
  error (ErrorResponse, optional)
}

ErrorResponse {
  errorCode (integer, optional),
  errorDescription (string, optional)
}

8.5.5 /enterprise/RegisterSession

8.5.5.1 Input

SessionRequest {
  sessionId (string): Session Opportunity Identifier
}

8.5.5.2 Output

RegisterSessionResponse {
  error (ErrorResponse, optional)
}

ErrorResponse {
  errorCode (integer, optional),
  errorDescription (string, optional)
}
8.5.6 /enterprise/AuthenticationResponse

8.5.6.1 Input

Request {
    sessionId (string, optional),
    result (integer, optional)
}

8.5.6.2 Output

Response {
    error (ErrorResponse, optional)
}

ErrorResponse {
    errorCode (integer, optional),
    errorDescription (string, optional)
}

8.6 Role API

8.6.1 Overview

The Role Gathering is retrieved from an authoritative Role Source, i.e., Active Directory, LDAP or relational database Big Data server, or conducted through an additional API call on the BOPS server to find the list of the roles. Roles are gathered and stored in the BOPS server.

<table>
<thead>
<tr>
<th>Name</th>
<th>Definition</th>
<th>Input parameters</th>
<th>Output parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>loadRoleGenesis</td>
<td>Given a userId, deviceId and the systems, go to the well-defined role-gathering source and replace the roles in BOPS. This also cancels all open sessions. All sessions shall be reconstructed after this API call. The duration of each session is a security policy decision. So, how long each session lasts and how long of inactivity prior to the creation of a new session (Time To Live), the device scanning result may be sent to the BOPS server to continue the session validation.</td>
<td>Input userId, deviceId</td>
<td>The roles are loaded into server memory no output.</td>
</tr>
</tbody>
</table>

8.6.2 Dynamic image code session construction at a glance

The web page for a dynamic image returns a sessionId. A sub-API call returns a Multi-purpose Internet Mail Extensions-encoded image that has the sessionId in the dynamic image. The next API call returns a URL of the image and the sessionId in JSON text format. At the conclusion of the session construction all Roles (labels) are associated with the User.
<table>
<thead>
<tr>
<th>Name</th>
<th>Definition</th>
<th>Input parameters</th>
<th>Output parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>sessionConstruction</td>
<td>This API is used to start a session that shall create a sessionId to identify this session. Besides the sessionId, the API shall also return a dynamic image with embedded code information.</td>
<td>siteId</td>
<td>Returns sessionId</td>
</tr>
<tr>
<td>sessionCreation</td>
<td>This API calls the backend after the Mobile Client Application (MCA) scans a QR code displayed on the embedded device.</td>
<td>siteId, sessionId</td>
<td>The server shall use the input parameters to create a new persistent store document. All the input parameters shall be saved to the persistent store document. Also in this persistent store document, a field “status = CREATED” shall be inserted too.</td>
</tr>
<tr>
<td>sessionStatus</td>
<td>This is an API to check the current session status, which is associated with the given sessionId.</td>
<td>sessionId</td>
<td>The following result codes are returned: sessionNotReady, validationInProgress, userAuthenticated, userRejected, sessionTerminated, sessionExpired, sessionLogoff, userLogoff.</td>
</tr>
<tr>
<td>sessionData</td>
<td>An API call after QROpportunity from the embedded device to the backend.</td>
<td>sessionId, siteId</td>
<td>The server shall use input parameters to retrieve the data. If the persistent store document is found, then the API call returns “status = CREATED”.</td>
</tr>
<tr>
<td>sessionTermination</td>
<td>This API is called from the embedded device to the backend after a complete transaction.</td>
<td>sessionId, siteId</td>
<td>The server marks “status=ENDED” and keeps all session data.</td>
</tr>
</tbody>
</table>

An input device scans the dynamic image and validates the scanned sessionId with BOPS, which triggers the triple association of user, device, and session. The BOPS implementation client software validates the biometric. The biometric status is sent to the BOPS server. The biometric data itself is never sent to the BOPS server for the privacy concerns.

The next steps: the device scans biometric and sessionId is created. The session status sessionId returns sessionNotReady, validationInProgress, userAuthenticated, userRejected, sessionTerminated, sessionExpired, sessionLogoff, and userLogoff.

Session termination brings a logoff notification. Once received, the session is closed for a future activity as defined by the sessionTermination in sessionId. All sessionId creation failures shall be handled by Intrusion Detection System (IDS), which can take the appropriate actions to terminate a sessionId creation. This may be blocking IP addresses, blacklisting domains, blacklisting users, or other means. The blacklisting has a hierarchy of a restricting access to the system based on the complex machine learning rules.

8.7 Access control API

Given sessionId, the data label and the access are allowed.

The set of data in JSON format (JSONArray of JSONObject) is a securityLabel field. The security label field is matched against the roles associated with the user through the session. If the data (JSONObject) is a subset of the roles, then the data is returned. Otherwise, the partial data of JSONObject is not returned.
As the API redirects the call, the returned data becomes restricted. At the redirect API call; intercept a getJSON call.

The access control algorithm is applicable for each User at the session construction time flattened the hierarchies. So, if the user is a Manager implies that the Manager label is both a Manager and a User, then:

If Bob is a Manager, the labels for Bob are Manager, User

If a Piece of data is Manager, the hierarchy is not flattened.

For adjudication, if the data is a subset of the users roles (groups), the adjudication allows the user to see it:

No read up, no write. Bell-LaPadula model.

At a given point in time, the user works at the non-hierarchical security level. So, irrespective of the number of flattened labels, the user works at one label at the time, when it comes to writes. Thus, if Bob is working as a manager, he may only write data as a manager. If he is working as a user, then he may only write data as a user. This prevents the security policy from violation by “write down.”

<table>
<thead>
<tr>
<th>Name</th>
<th>Definition</th>
<th>Input</th>
<th>Output</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>adjudicateAction</td>
<td>Adjudicate an action</td>
<td>An identity from 2-way ssl SSL/TLS certificate; a comma separated set of the labels.</td>
<td>Actions allowed: read, write, update, delete.</td>
<td>If BOPS doesn’t store data.</td>
</tr>
<tr>
<td>addData</td>
<td>addData to the BOPS Store.</td>
<td>An identity from 2-way ssl SSL/TLS certificate; data stored in the tag, value pairs; a comma separated set of the labels for each piece of data.</td>
<td>Success or failure.</td>
<td>If BOPS stores multi-level secure data.</td>
</tr>
<tr>
<td>deleteData</td>
<td>removeData from the BOPS store.</td>
<td>An identity from 2-way ssl SSL/TLS certificate; the tags to remove.</td>
<td>Success, if data is removed; Failure, if there is a security exception or insufficient privileges</td>
<td>If BOPS stores multi-level secure data.</td>
</tr>
<tr>
<td>readData</td>
<td>readData from the BOPS store</td>
<td>An identity from 2-way ssl SSL/TLS certificate; name, value pairs to the read function.</td>
<td>The data is in a JSON format that the user may see based on security labels.</td>
<td>If BOPS stores multi-level secure data.</td>
</tr>
</tbody>
</table>

8.8 Auditing

All steps of the Identity Assertion, Session Creation, and Adjudication have an audit capability. The capability may be set for any user, groups of users or roles across any action (read/write) on any set of data. The audit is gathered RESTfully and then stored on a BOPS Server.
API for audit request

<table>
<thead>
<tr>
<th>Name</th>
<th>Definition</th>
<th>Input</th>
<th>Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>startAudit</td>
<td>2-way SSL/TLS for identity, optionally a group; action as read, write, update, delete, optionally a data object. If data object is not specified, then all data shall be audited for the user.</td>
<td>A group or a user, action to audit (read, write, update, delete) or optionally a piece of data to apply the audit.</td>
<td>Not available</td>
</tr>
<tr>
<td>stopAudit</td>
<td>2-way SSL/TLS for identity, optionally a group; action as read, write, update, delete, optionally a data object. If data object is not specified, then auditing of all data for the user shall be stopped.</td>
<td>2-way SSL/TLS for identity, a group or a user, action to audit (read, write, update, delete) or optionally a piece of data to apply the audit.</td>
<td>Not available</td>
</tr>
<tr>
<td>auditRecord</td>
<td>2-way SSL/TLS for identity, action (read, write, update, delete), source of data. This writes an audit record.</td>
<td>2-way SSL/TLS for identity, a piece of data to audit (tag, value in JSON format), and the action, which is being audited.</td>
<td>Not available</td>
</tr>
</tbody>
</table>

API for read audit logs

<table>
<thead>
<tr>
<th>Name</th>
<th>Definition</th>
<th>Input</th>
<th>Output</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>readAudit</td>
<td>2-way SSL/TLS for identity, start date in ISO8601 format; end date in ISO8601 format. If there is an administrator role, then the audit record is returned in JSON format.</td>
<td>2-way SSL/TLS for identity, to show user an audit record; datetime for start; datetime for end; data records to report (allowing “wild cards”).</td>
<td>The audit records in JSON format.</td>
<td>Shall have an administrator privilege to perform audit.</td>
</tr>
</tbody>
</table>

8.9 Administration

The mapping of Users to Groups and Groups to Roles and Attributes to Groups is provided by an API call. All calls require a 2-way SSL/TLS communication layer and should be conducted by the administrator role.

Example: UPDATE_URI=https://xyz.domain.com:8443/{BOPS_Instance_Name}/JSONUpdate

Example: UPDATE_URI=https://xyz.domain.com:8443/{BOPS_Instance_Name}/JSONUpdate
<table>
<thead>
<tr>
<th>Input parameters</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>the users name</td>
</tr>
<tr>
<td>id</td>
<td>the unique identifier for a user</td>
</tr>
<tr>
<td>login</td>
<td>the user login</td>
</tr>
<tr>
<td>password</td>
<td>the password used for a role gathering source</td>
</tr>
<tr>
<td>category</td>
<td>always “User” the persistence engine</td>
</tr>
<tr>
<td>email</td>
<td>the primary email for the User</td>
</tr>
<tr>
<td>groups</td>
<td>a comma separated list of group ids for which the user is a member</td>
</tr>
<tr>
<td>siteId</td>
<td>the siteId (organization) of the user</td>
</tr>
</tbody>
</table>

**To merge or update a user**

| name             | group name |
| id               | the unique id of the group |
| description      | description of the group in text format with spaces allowed |
| category         | always “Group” |
| attributes       | a comma separated list of attributes that are associated with every User in the Group |
| roles            | a comma separated list of roles in non-hierarchical format. All hierarchies are flattened |
| users            | a comma separated list of users ids that are members of this group. |
| siteId           | the siteId (organization) of the group |

**To add or update a group**

| name             | role name |
| id               | the unique id of the role |
| description      | description of the role in text format with spaces allowed |
| category         | always “Role” |
| siteId           | the siteId (organization) of the role |

**8.10 Reporting**

The administration level report is available in the auditing API.

**9. Client device requirements**

For an identity assertion, there are several requirements to the client devices. These requirements include:

a) The client devices shall not perform key generation. The key generation happens during the Genesis process and is stored on the device.

b) The client device shall implement the aforementioned value1 and value 2 replay prevention algorithms.

c) All communication via the client devices and the server shall be via 2-Way Secure Socket Layers.

d) All passivated client device data shall be encrypted. There shall be no unencrypted artifacts on the client devices.

e) All data transfer between the client device and the server shall be encrypted at the transport layer.

f) The client software shall blacklist itself if the user fails to authenticate more than X consecutive times. The client software shall have a client intrusion detection system capable of seeing the patterns of trial and error for blacklisting itself. All applications designed for use with the Hoyos BOPS specifications, shall include some form of the intrusion detection, whereby the software can detect spoofing attempts and restrict the access to the backend system. This would be defined as X.
amount of tries, which then cause the client application to stop working for \( Y \) period of time or indefinitely until the device can be properly assured that it is safe and valid.

g) Liveness requirement. It is required that all applications, which are intended to comply with the BOPS specification, include some form of liveness detection or an ability to maximize that the enrolled or authenticated user is an actual person and not an image or other representation of the user. For a face recognition system, it could be blink detection, for instance. The liveness requirement should be in place for anti-spoofing prevention and blocking of the false access into the system. The choice of a liveness is up to the enterprise organization that should determine which use case fits the best its particular needs.

h) The false negatives shall be below 1.2%.

i) The false positives shall be below .05%.

j) Biometric matches shall occur within 5 seconds.

k) Liveness checking shall be configurable on the server and at three levels. Level 1 shall take less than 8 seconds. The false positives in this mode shall be below 0.05 percent. Level 2 liveness checking shall take less than 12 seconds and the false positives in this mode shall be below 0.03 percent.

10. Server-side intrusion detection system

10.1 API list blacklist

For the case of a replay attack, for instance, an incident may be added to the Intrusion Detection System (IDS). If an incident is promoted to an attack, then the response shall have blacklist as “true”, otherwise, the device is valid.

/listAttacks

Input

ht
Request {
  id=(string, mandatory): The actual device with the incident.
}

Output

Response {
  blacklist(true|false)
  error (ErrorResponse, optional)
}
ErrorResponse {
  errorCode (integer, optional),
  errorDescription (string, optional)
}
10.2 API—Incident

For the case of a replay attack, for instance, an incident may be added to the Intrusion Detection System (IDS). The IDS uses a variety of techniques to increase, or decrease the area of an incident find if an area, domain, deviceId is under attack.

/checkSecurity

Input

Request {
  ip=(string,optional): Source IP address of device or network gateway.
  &mac=(string, option): MAC address of device or network gateway.
  devId=(string, mandatory): The actual device with the incident.
  dom=(string, option): Domain of device or IP address. Used to find larger area for attack.
  fField=(string, option): From field or the field that was not an expected value. Could be value1 or value2 or anything else
  aVal=(string, option): Accepted value. The value received which was not correct.
  dVal=(string, option): Desired value. The value that should have been received.
}

Output

Response {
  error (ErrorResponse, optional)
}

ErrorResponse {
  errorCode (integer, optional),
  errorDescription (string, optional)
}
Annex A

(informative)

Glossary

admin console: An online portal that facilitates the registration and enrollment with BOPS.

application: A unique software/system, which is created using the BOPS Application Programming Interface (API) key.

BOPS admin: A BOPS administrator, who sets up an environment and creates an Original Site Admin based on the enrollment information during the registration.

BOPS cluster: A set of loosely or tightly connected computers, devices that communicate using BOPS.

BOPS server: An instance of the server, such as in the client/server paradigm, which supports BOPS architecture.

BOPS IDS: An instance of the Intrusion Detection System on the private cluster that supports BOPS architecture.

client device IDS: An instance of the Intrusion Detection System running locally on a user device.

Jena rules: Syntax and a system of machine learning rules for inference.

IDS cluster: A set of loosely or tightly connected Intrusion Detection Systems that supports BOPS.

Liveness: An aspect of algorithm that defines an animated object in Computer Vision.

original site admin: An administrator created by the BOPS administrator with the privilege to create other administrators within the same organization. The Original Site Administrator should be able to assert his/her unique identity according to the client requirements (See section 5.1.2(i) Genesis API/Client Requirements Note).

site admin: An Application administrator who is created by The Original Site Administrator.

trusted adjudicated data: The data, which is stored in BOPS with Multi-level Secure adjudication in the BOPS server.

user: A unique user, whose identity is being asserted by BOPS that may have several devices.

user device: A single device that has biometric-driven client software.
Annex B

Bibliography

