SECURING ONLINE ACCOUNTS VIA NEW HANDSHAKE PROTOCOL AND GRANULAR ACCESS CONTROL

Mehrdad Nourai and Haim Levkowitz

Computer Science Department,
University of Massachusetts Lowell, Lowell, MA, USA

ABSTRACT

When we need to make informed financial decisions, we seek out tools to assist us with managing and aggregating our finances. Traditionally, money management software packages have been available for personal computers; however, they were expensive and often had steep learning curve. With a paradigm shift to cloud-computing users are looking toward the web for an easier and low-cost solution. As a result, third-party companies have been formed to fill this gap. However, users have to share their login credentials with the third-party, and if that information gets compromised, an attacker can access and perform transactions on their account.

We present a novel, holistic model with a new handshake protocol and access control, which authenticates and forms a sandbox around a third-party access. When utilizing these novel techniques, users' original login credentials can remain private, and no one would be able to perform transactions on the users' account.

KEYWORDS

Security, Network Protocols, SSL Cryptography, PKI

1. INTRODUCTION

Today, all of our financial accounts are accessible online, and often they tend to be with different institutions. When one needs to figure out the overall picture of their finances (e.g., net worth or track what is happening with their money), one would need to start by logging into each of their accounts individually. This involves remembering login credentials for each account. Ideally, for good security practices, each account should have unique login credentials, however as a convenience, users' may use one set of credentials for most (if not all) of their accounts. Once the users log into their account, to get a big picture, they would need to download their account information in the proper format and import it to a locally installed financial software package (e.g., Intuit Quicken). Although using these tools are an improvement over tracking your financial life by hand, this process can be tedious, time-consuming, and may become overwhelming for some users that are not familiar with the world of finances. There are usability issues and inconveniences with the locally installed budgeting applications. For instance, the software localized to one computer and needs to be installed, maintained and patched to avoid security vulnerabilities. Also, they tend to be expensive, and their interfaces are a bit complicated to use and often change over time with each iteration or edition of the software. This model has a steep learning curve and although it may have been sufficient or the only form of financial aggregation software available years ago, it is no longer the case. Thus, it is not surprising that users are migrating to online tools for managing their personal finances.

The idea behind the third-party company is to provide a set of budgeting features that were previously offered by the locally installed financial software, with the added advantage of the third-party software doing all the work for free or for a small fee. For third-party companies to provide these services, they would need to login to users' online accounts to read and collect information. The third-party can utilize desired algorithms on the users' account information to critically examine transactions, net worth, and other finances, then create and present an aggregate report in a textual or graphical manner. This is a preferred method among users, who may have used locally installed software that they had to purchase, keep updated, and perform backups on a regular basis.

Although the online budget aggregate tool is an excellent and affordable tool, users have security concerns and are vulnerable to attack when they sign-up to use these types of services. The vulnerability starts by giving private accounts' login credentials to third-party companies. If an attacker manages to compromise third-party provider's computer, they have got users' entire financial lives in their hands. This is because, the current design of online accounts is in a way that when someone logs into a bank account, everything that the owner of the account can do there, they can do, too. That is, it is not just that they could look at one's bank accounts, or credit card information, they can also transact on it, too. The main idea of this paper is to showcase a novel and holistic login design with techniques for securing online accounts, by leveraging a whole new separate set of login credentials with lower permission. We explain precisely the proposed techniques which are needed to protect online accounts from potential fraudulent activity when users utilize services offered by third-party companies. The main contributions of this paper are:

- To introduce a new handshake protocol to be used by a third-party to authenticate access to users' online accounts (discussed in Section 4.2).
- To introduce a new granular access control layer for fine-grained access capability to users' online accounts (discussed in Section 7.4).

2. EXISTING PRACTICES AND INFRASTRUCTURE SHORTCOMINGS

We now describe what is currently being used in practice and its shortcomings.

2.1. Current practices

For financial institutions to provide a secure online mechanism for their customers to access their accounts, financial institutions utilize HTTPS (HTTP over SSL) technology that can be used via a web browser to access their account. The current process is as follows; one opens a web browser on the user's computer, types in "https://" followed by the website address of their financial institution. This communication between the client and server is secured using encryption and handshaking via SSL protocol. When the page is displayed, it may contain an area within the page to obtain the customer's login credentials or may have a sign-in link to open a login page. The mechanism used for inputting the account credentials utilizes HTML FORM INPUT tag via POST method. Customers get a couple of textboxes, one for the username and one for the password and a sign-in button. Once the user inputs the necessary fields and clicks the sign-in button, the process of user authentication gets started.

The financial institutions' server converts customers' passwords into a custom hash using the algorithms they first used to create the hash (i.e., when the user first set up the account's password), and checks for a match. If it matches, the customer is allowed to access the account. If it does not match, the server may allow a few more tries before locking the account. In addition to this process, financial intuitions often ask for more verification if their customer is signing in from a new device. This extra step involves one or more security questions that the user provided answers to when they first set up their account's credentials. The extra step is an added security measure to ensure that even with the hacked password, the attacker would have one more challenge before gaining access to the account. With the current practices, there are variations to these steps. Hence, not all financial institutions follow a standard mechanism for authenticating their users' access. For instance, some may display a predefined image for identification of the legitimate versus fake website that was made to look like their financial intuition's website. Others may ask for the username on the home page but not password at first, until the user clicks the sign-in, continue, or next button. There is also the second authentication using a code that is delivered either via email, phone, or a mobile app. In general terms, the extra steps may consist of some other information that the owner of the account knows other than their password which financial institutions can ensure it is indeed the owner of the account that is attempting to accessing the account. Therefore, a hacker has to break down at least two barriers to gain access to the account. While this process works well in practice, developers designed it for the humans' capabilities, and not for machines. Thus, financial institutions had to make their interface easy enough for human use, as well as appealing to the masses that use online services. That is, the login credentials used in current practices are a compromise between security and user's convenience.

2.2. Current Infrastructure Shortcomings

The infrastructure of online accounts lacks the mechanisms to support a different form of account authentication with restrictive access. As a result, users are giving their personal access credentials to third-party companies to utilize the financial services they provide. With ever-increasing cyber-security threats, coupled with the existing industry practices, users may make themselves vulnerable to cyber criminals. The current practices have created a challenging and engaging problem that needs to be solved to keep the users safe from potential cyber-attacks.
The following is a list of potential problems with the current infrastructure:

- Users are making themselves vulnerable by giving their banking credentials in the form of username/password plus security questions and answers to third-party companies.

- The users' credentials are designed to be private and not shared with others.

- Once users' credentials are given to a third-party company, they can be stored on their server, which may be exposed to an attacker.

- Users may use the same username/password for other accounts or other places, and as a result, if an attacker steals their credentials, they can access more than just that account.

- Current bank accounts are full-access accounts, and as a result, once a third-party company has access to these accounts, they can perform transactions on that account.

- Financial institutions are not the only company that always allow full-access once users' credentials are entered. Hence, other online accounts that users share with others may be at risk of being vulnerable to an attacker.

3. NETWORKING INFRASTRUCTURE

In this section, we will discuss the foundation of the networking infrastructure that our new protocol will utilize to deliver the needed security.

3.1. Secure Channel

Secure channels between two parties are needed when the transmitted information is sensitive and private while traveling over an insecure medium (e.g., the Internet). The current practices referred to as SSL, which is for securing a channel for private communications will be discussed next.

3.2. Secure Sockets Layer

The current secure connection technology used on the Internet for securing communications between a client and a server is called SSL (Secure Sockets Layer), and its predecessor TLS (Transport Layer Security). Although TLS is the next generation of SSL, the term SSL is prevalent and therefore we will use it throughout this document.

SSL protocol was originally developed by Netscape Communications in 1994 to address security issues of communicating via the Internet [1]. The protocol was a revolutionary technology for securing the Internet traffic that carried personal or sensitive information. The SSL technology is over two decades old and has evolved over time to be more secure. When new SSL vulnerabilities are discovered, computers become faster, and security demand of institutions grows, SSL will continue to evolve over time to address the needs of users. The workings of SSL depend on trust models provided by Certificate Authorities (CA) and public key infrastructure which is based on cryptography. Its underlying mechanisms protects the transmitted information integrity and security by providing authentication of the server to the client, and optionally provides authentication of the client to the server. Although SSL has several security features built-in, it is not immune to cyber-attacks (discussed in Section 8.2). Our new protocol will leverage SSL technology and its ability to secure the communications between client and server without any changes to this layer.

3.3. Transport Layer

The Transmission Control Protocol (TCP) and User Datagram Protocol (UDP) are located in this layer. TCP is a connection-oriented protocol and has three-way handshakes to establish the connection between the client (Initiator) and server (Receiver). TCP is the most reliable and prevalent protocol on the Internet because it guarantees packet delivery, ordering, and congestion control. UDP is a connection-less protocol that does not guarantee packet delivery and ordering which is typically used for streaming applications. The TCP along with IP layer, make up the TCP/IP protocol, which we will use as our transport layer protocol. No major changes are needed in this layer, however, with a minor exception that TCP flags might be set to flush out the packets.

3.4. Secure versus Insecure Network Ports

The TCP and UDP transport layer protocols have 65535 ports each. The Internet Assigned Numbers Authority (IANA) assigns the ports to be used for specific protocols and general use [2]. Although some ports are used for secure protocol, there is no such thing as “secure” or “insecure” port numbers. The traffic that flows through network ports can be encrypted or in plain text. Hence it is up to the underlying protocol to use them as “secure” or “insecure” ports.

Nevertheless, there are benefits in standardizing assignments of default ports for “secure” or “insecure” traffic. This may reduce confusion or errors of using certain common ports for secure communications during the setting up of the firewall rules.

4. APPLICATION LAYER PROTOCOLS

In this section, we will discuss the current HTTPS and its issues, and then present our new HTTPAS protocol which addresses security concerns of current practices when used with third-party companies.

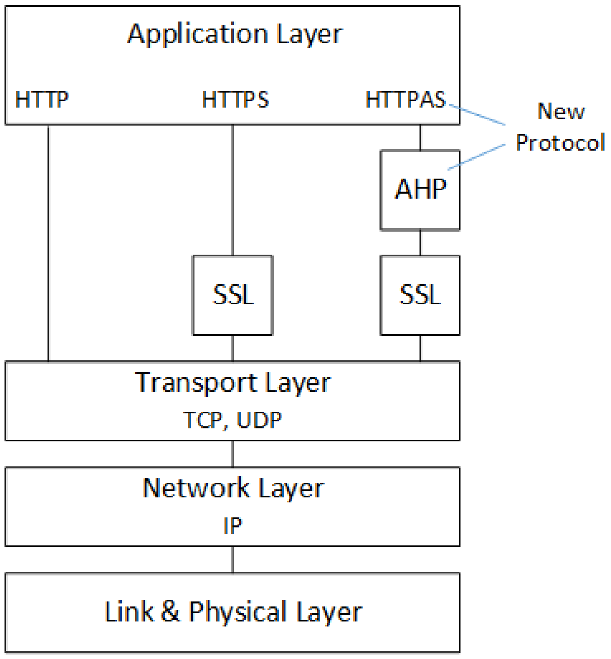


Figure 1. The TCP/IP stack with the new HTTPAS and the new authentication handshake protocol

4.1. HTTPS Protocol

The Internet is an open and insecure medium for communication, hence, when a connection is made between a client machine and a server, all the information transferred over the Internet is traveling over an insecure wire. This information can be intercepted and be seen by others. Therefore, sensitive information (e.g., username/password, bank account information, medical records) must be protected while the information is in transit. The Hypertext Transfer Protocol (HTTP) that is mostly used for accessing information via the web is done in plain text which makes it vulnerable to an attacker. To protect data going between a client and a web server, a protocol called HTTPS is used. HTTPS consists of HTTP over SSL protocol, which encrypts and secures the communication to protect it while in transit [3]. HTTPS works flawlessly and behind the scene, and it does that without user intervention unless something goes wrong with the connection, which then informs the user and allows the user to decide what to do next. When users open a website using HTTPS with their browser, and if it supports the HTTPS protocol, a green lock icon is shown as part of the link in the browser. Once the lock is clicked, information about the security and connection are displayed. For instance, details can show the connection information as TLS 1.2 AES 256 GCM RSA (2048), which is the version and cryptography specifications of the connection. HTTPS was geared toward human usage for securing their sensitive online web activity and interactions. No changes are needed to this protocol. Instead, we will introduce a novel protocol for computer-to-computer communications which we will discuss next.

4.2. Introducing the New HTTPAS Protocol

We have designed and created an architecture for a novel application layer protocol called HTTPAS, which stands for HTTP with new Authentication Handshake Protocol (AHP) over SSL. This new protocol utilizes SSL to secure a communication channel between a third-party's computer and a financial institution's web server and then uses the new AHP for the two computers negotiate and authenticate secure access to users' accounts. The motivation for a new protocol is flexibility, extra security, and custom enhancements that the current HTTPS does not offer. The HTTPS protocol was designed to be a general multipurpose protocol for providing secure communication channel on the web. This protocol is often used for human-to-computer communications which require more conveniences for a human user. Therefore, security versus convenience became a compromising factor.

The new HTTPAS protocol closes this gap by offering a set of features that is well-suited for computer-to-computer communications. This protocol can also be adapted for human-to-computer communication, however, due to extra security, it would require more efforts on the human side. Our new approach increases the security to another dimension by utilizing a public key infrastructure (PKI) framework. As a result, we can eliminate the need for third-party companies to ask for and use a user's username/password plus other login credentials while offering extra and better security not found in current practices. We will explain components of this new protocol (discussed in Section 5) in greater details later in the paper.

The diagram in Figure 1 shows HTTPAS within the realm of the TCP/IP network stack. The new protocol consists of new Authentication Handshake Protocol (AHP) (discussed in detail in Section 5), which authorizes a client computer and authenticates access to users' accounts. It uses SSL as its foundation for a secure communication channel. Using existing SSL protocol will reduce or minimize the risk of introducing new vulnerabilities.

The following are the benefits of using HTTPAS for third-party access:

- The solution we are envisioning would result in better security practices which address concerns of existing users and potential new users. The existing users get the better security. The new users that were hesitant to sign-up with a third-party due to security issues of such services, can now be sure that the third-party cannot perform any transactions on their accounts. This can potentially increase the number of users' base utilizing third-party services which benefit all parties involved, i.e., users, banks, and third-party companies as a whole.

- Users' don't have to give their banking credentials which can be in the form of username/password plus security questions and answers to a third-party site, i.e., their credentials which were meant to be private, can stay private and not shared. Instead, the third-party will utilize the new handshake protocol and access control for accessing users' accounts.

- Often users have the same username/password for more than one online account. As a result, once an attacker steals their credentials from a third-party's server, an attacker can get access to those accounts in other places, which can become a major security issue for the users.

- The solution is not limited to banking websites, hence, it can be adapted for other sites, such as email or any online accounts, in general, that use usernames/passwords for their authentication. A third-party can access these accounts on a read-only basis.

- If a third-party's server is compromised by an attacker and access information is stolen, the attacker cannot perform transactions on the account. In addition, the bank and/or the owner of the account can easily revoke the access and generate a new access protocol, which can be safer and more convenient than with current practices.

4.3. Port Numbers

To place the separation of traffic between the current and the new protocol, as well as, minimize or eliminate any changes to the existing protocols, HTTPAS uses a different TCP/IP port number than HTTPS. The current application layer TCP/IP protocol port assignments for existing application protocols have been designated by IANA, which are as follows: port 80 for HTTP and port 443 for HTTPS. The new HTTPAS protocol does not currently have a designated port assignment. Hence, in the interim, we will use default port 10443 which according to the IANA online registry search tool, has not officially been assigned to any protocol.

5. NEW AUTHENTICATION HANDSHAKE PROTOCOL

The new Authentication Handshake Protocol (AHP) utilizes public key infrastructure (PKI) framework, TCP/IP as its transport layer and SSL as its transport layer security for its underlying mechanisms. AHP is the main component and workhorse behind the new HTTPAS protocol and is responsible for authorizing and authenticating the access to users' accounts. In this section, we will describe the precise details of the new protocol.

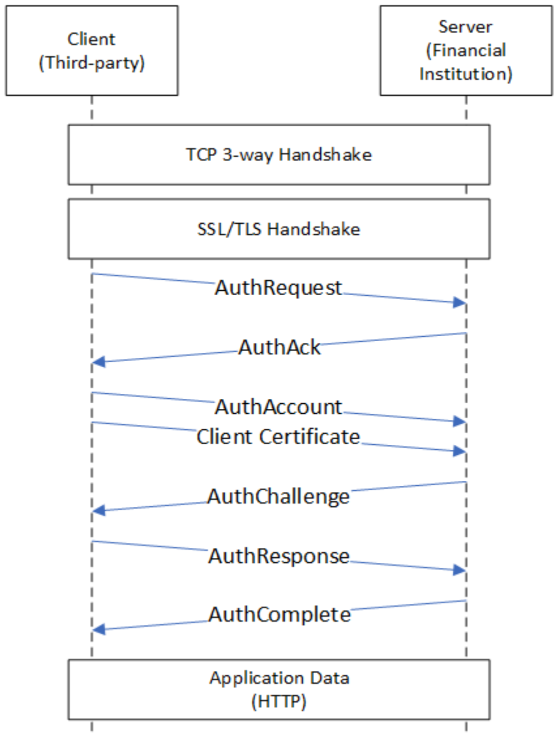


Figure 2. Sequence Diagram

5.1. Cipher Specification

We will use the secure communication channel that SSL provides as the foundation of a secure connection. SSL negotiates Cipher Spec between client and server, therefore, for performance reasons we will leverage the negotiated Cipher Spec to minimize unnecessary handshaking between client and server.

5.2. Sequence of Events

AHP is responsible for the following sequence of Events:

- Negotiate AHP version to use for the handshake
- Obtain user account id that the client is requesting access for
- Obtain client's computer certificate and public key
- Verify client's computer certificate with certificate authorities
- Verify that the user account has been authorized to be access from the client computer
- Authenticate the access utilizing challenge/response and pass-phrase
- Grant or deny access to user's account

5.3. Sequence Diagram

The diagram in Figure 2 shows the new handshake protocol between the client and the server. Note that the TCP and SSL handshake protocols have been drawn as a rectangle on the sequence diagram to show their placement within the protocol time line. A detailed explanation of those protocols is outside the scope of this paper. The new handshake consists of several components as follows:

- AuthRequest
- AuthAck
- AuthAccount
- AuthChallenge
- AuthResponse
- AuthComplete

We now describe each of the new handshake's components in a greater detail.

5.3.1. AuthRequest

The first step of the new AHP handshake protocol is AuthRequest, which occurs right after completion of SSL handshakes protocol securing the channel for communication. The client will start by sending a request for authentication by providing a list of AHP protocol version numbers it supports in a string (listed in order of preferred version numbers). The string must contain a comma separated displayable characters, with two zeros as terminating character. The purpose of exchanging version numbers between client and server is that in case either of the parties does not support some newer version numbers, the client and server can synchronize on which AHP protocol version they can use for the successful handshake.

5.3.2. AuthAck

The server will check to make sure the version numbers are acceptable and respond to the client with a version number string of its own (similar string format as client version number string). The client adjusts the protocol to the server required version number. Note that, for security reasons, only the server can decide on what AHP protocol version number to use. If the server does not support or agree on the client's desired version number, the connection is terminated.

5.3.3. AuthAccount

The client provides user account id information along with its CA certified digital certificate and public key. Once the server receives the necessary information, it checks the client's certificate with certificate authorities for authenticity and extracts domain information. Now the server must check to ensure that the user account id exists and the client has been authorized to access the account. The connection will be terminated if any of these steps fails to complete successfully. Note that although SSL provides an optional client verification, we will perform the client verification as a mandatory step here. This extra step ensures that even if the user account's key were compromised, the key would not be accepted from an unauthorized client.

5.3.4. AuthChallenge

The server creates a challenge string made up of an arbitrary length of random numbers, and then encrypts it with the user id's public key and sends the challenge string to the client. The random number can prevent a forward attack at which the challenge string cannot be reused for future authentication.

5.3.5. AuthResponse

The client must first decrypt the user account's private key utilizing a pass-phrase, then decrypt the challenge string and send it to the server.

5.3.6. AuthComplete

The server checks the decrypted string, and if there is a match, it returns AHP_SUCCESS. Otherwise, it returns an AHP_FAILED code. If authentication was successful, the HTTP request/response commands could be sent, otherwise the connection will be closed and no other data would be accepted.

6. TRUST MODEL

To identify a server or a client machine that we want to communicate with, a trusted source is needed to verify the identity of either of the party. We will discuss that next.

6.1. Certificate Authority

Certificate Authority (CA) is an entity (e.g., VeriSign, GeoTrust, Comodo, DigiCert) that is trusted and which issues digital certificates to be used for authentication of computer systems on the Internet. CA validates the originator of the request for a digital certificate before issuing their certificate. The concept is similar to trusting government institutions that issue a driver's license which is often used for identification.

6.2. Certificate

Certificates are based on X.509 standard [4] and are digital documents which are generated and issued by a CA. Certificates are made utilizing the public-key cryptography (PKI) which enables parties communicating over the Internet/network to validate the identity for establishing a secure connection [5]. Digital certificates can be purchased from CAs for an annual fee or other purchase plans offered by a particular CA. Certificates can also be locally generated (i.e., self-signed certificate) which are typically used for development and testing purposes. We require that third-party companies purchase digital certificates from their desired CA and deploy them on their computers to be used with our handshake protocol.

6.3. Certificate Revocation Lists

To identify the validity and status of a digital certificate (e.g., revoked or expired), CA keeps a record of those certificates in a list called the Certificate Revocation Lists (CRL) [6]. Our handshake protocol will check the CRL to ensure that the digital certificate is valid and has not been revoked. This is a critical step in the authentication process to ensure that the client or server meets the identity requirements of the CA.

7. ACCESS CONTROL

In general terms, access control includes the systems protection mechanisms for granting/denying access to the available resources. This ensures that persons or services that are requesting access to a resource have been authorized prior to use of the resource. The resource can have (but not limited to) the following protection attributes: read, write/modify, and delete. The protection mechanism needs to be designed for each type of system and may vary widely from system to system. [7]

7.1. Principles of Least Privilege

The best approach to giving authorization to a resource is to use the concept of giving the Least Privilege, i.e., giving what is needed to use a resource and restricting everything else [8]. Utilizing this principle can limit what can be done to a resource, which can minimize data leaks or misuse.

7.2. Access Models

Access control security can be designed for infrastructure protection via one or more of the following security models [9] (listed in alphabetical order):

- Access Matrix Model
- Bell-LaPadula Confidentiality Model

- Clark-Wilson Integrity Model
- Discretionary Access Control (DAC)
- Mandatory Access Control (MAC)
- Role-Based Access Control (RBAC)

These security models provide an array of protection and access control for a variety of real-world applications, such as operating systems, database, and web systems. This paper is interested in the Role-Based Access Control (RBAC) model since it provides the necessary granular access for securing online accounts. The RBAC has been used for protecting a variety of domains for government and commercial infrastructure [10]. We will utilize the RBAC to define new roles and types of access for particular web objects to provide added security for online accounts.

7.3. Current Access Mechanism

With the current account access mechanism, when a third-party company logs into users' online accounts at their financial institutions, they get the same privileges as the account's owner, i.e., full privilege access to users' accounts. This is the by-product of using the same authentication mechanism for logging into a user account, which makes it difficult to distinguish access made by a third-party company versus the account's owner. As a result, financial institutions may not be able to offer any account access control to their users, unless they can differentiate between accesses made to the same account. We believe that no one other than the account's owner should have full privileges to users' accounts, even when they were logged in successfully to that particular account. Therefore, the third-party access must be limited and sandboxed to only allow display of account information or authentication of the account, without any other privileges which account owner usually gets.

7.4. Granular Access Control via RBAC Model

With the new online account security model, we are envisioning in this paper, it makes it feasible to offer granular access control for the online accounts. Since it would be possible to distinguish third-party accesses from the account's owner access (e.g., based on which protocol used). Utilizing a granular access control with the role-based scheme of the RBAC model enables fine-grained access to sandbox third-party companies. When a third-party uses the new protocol to access the users' accounts, the source login by definition becomes distinguishable, and sandbox becomes feasible. That is, financial institutions can detect the alternative form of account access, when it is initiated using the new protocol versus current protocol, and then allow the access according to the initiator's role.

We now define the architecture of the access control structure as access pair (Object, Attribute) with Role in greater detail.

7.4.1. Roles

We define the following three roles to address the security of the accounts and enable each role to perform their authorized access:

- ROLE_ADMIN - This role performs account administration tasks. It offers full privileges that can allow an account manager at a financial institution to make changes to users' accounts.

- ROLE_OWNER - This role gives privileges to the owner of the account, which includes reading the contents of the account and performing transactions on the account.

- ROLE_THIRDPARTY - This role is used by third-party companies that are authorized by users to read the contents of their accounts. No transactions are allowed on the account via this role.

7.4.2. Object

The Object is the entity that needs to have granular access protection which can apply to the whole page, individual accounts, any components within of the page, to name a few. For example, objects can be a Checking Account, a Savings Account, or a Certificate of Deposit.

7.4.3. Attribute

The Attribute is a word length which consists of fine-grained privileges allowed for a particular object. We define four Attribute flags in a hexadecimal nibble format and present it as binary numbers as shown in Table 1:

Table 1. List of Attribute flags and their values.

Attribute flag	**Value (binary)**
ATTRB_READ	1000
ATTRB_MODIFY	0100
ATTRB_TRANSACT	0010
ATTRB_CUSTOM	0001

The binary numbers can be used with the logical OR operator to build a hexadecimal nibble representing an access configuration for a particular role. Then each role will have its dedicated hexadecimal nibble for specifying the privileges for that role. Note that, for the majority of objects, the first three privileges should cover most of the permission cases. However, a "custom" attribute has been added in case there are particular circumstances that an extra permission is needed.

7.4.4. Role Mask

The least significant hexadecimal nibble of an Attribute word is assigned to ROLE_ADMIN, then moving toward the most significant bit, the next nibble is assigned to ROLE_OWNER, and the next is assigned to ROLE_THIRDPARTY. We define role mask for the Attribute word in hexadecimal number as shown in Table 2:

Table 2. List of Role masks and their values.

Role mask	**Value** (hexadecimal)
ROLE_ADMIN_MASK	F
ROLE_OWNER_MASK	F0
ROLE_THIRDPARTY_MASK	F00

If new roles are needed, it can be directly added to the Attribute word after the last role moving toward the most significant digit. For example, access pair (Checking Account, 0x8EC) means, third-party can read, but cannot perform transactions or modify the Checking account in any shape or form; owner of the account can read, modify, and transact on the account; administrator (e.g., manager) can read and modify, but cannot perform any transactions on the Checking account.

8. SHORTCOMINGS

In this section, we will discuss security and performance shortcomings of our new techniques that we have discussed in this paper.

8.1. Performance Shortcomings

The new protocol HTTPAS is more secure when used for computer-to-computer communications. However, this new security enhancement comes at the cost of performance. The shortcomings of performance are mainly due to two major security steps such as verifying client's certificate and using public key infrastructure which uses asymmetric cryptography. Asymmetric encryption/decryption utilizing larger bits to avoid brute-force or other forms of attacks will run slower than a simpler method with lower security cryptography.

8.2. Security Shortcomings

In today's modern connected world, with many types of devices connecting us all together, security has never been more important. During our research and writing of this paper, cyber-attacks on two major companies were made public. Yahoo email attack was made public with data stolen from one billion accounts [11] and Yahoo data being sold on the "dark web" [12]. Dyn (domain name service provider) was attacked via Distributed-Denial-Of-Service (DDOS) causing interruption of services to the popular websites [13].

We now discuss security issues and vulnerability that can affect the security and integrity of secure communications over the Internet/network in greater detail.

8.2.1. SSL Protocol Vulnerability

Although SSL design and architecture has built-in mechanisms for preventing attacks (e.g., Eavesdropping and Man-In-The-Middle attack prevention features), it is not immune to attackers [14]. The vulnerability may come from flaws in SSL protocol, SSL libraries, hacked digital certificates, and cipher negotiations to name a few. These flaws are often discussed at security conferences such as Black Hat. Our protocol will be vulnerable to attackers if these infrastructures that we utilize as a foundation for our protocol become vulnerable.

8.2.2. CA Entities Vulnerability

The CAs are the foundations for trust model of digital certifications. If a vulnerability exists in CAs infrastructures and attackers can exploit it to their advantage, it will break the trust models. This may have such an adverse effect on the reputation of the CA which may result in cease of the operation of the CA. Nevertheless, if CA or certificates are hacked, it will make our protocol vulnerable to an attacker. An example of an attack on CA is, a CA named DigiNotar which was hacked by an attacker in 2011. The attack on DigiNotar had a catastrophic effect, and as a result, they were not able to recover from the attack, and filed for bankruptcy [15], [16].

8.2.3. Cryptography Vulnerability

Encryption algorithms have been proven to be mathematically sound, however, as computer systems are getting more powerful over time, the number of bits used in calculations of the encryption mechanism must also increase. For example, RFC7935 states that the RSA key pairs for use in public key infrastructure (which is a framework for what is called one-way function) must be using at least 2048-bit [17]. Therefore, the strength of encryption algorithms for a one-way function is based on the computing power and time required to decrypt the message.

8.3. Initial Burden for all Parties Involved

All parties involved (e.g., financial institutions, third-party companies, end-users) would need to make changes to their current processes, practices, and habits. For instance, financial institutions would need to modify their online accounts and systems to use the new protocol and offer new account granular access control; third-party companies must make changes to their current informational retrieval processes, as well as follow the specification of the new protocol for user authentication; the end-users would need to request alternate access credentials from their financial institutions for their accounts, provide the vendor information of the third-party, as well as change their full-access login credentials if they have already exposed that to others.

9. RELATED WORK

In this section, we will discuss related work in the area of authenticating users' access to online services.

9.1. OAuth

A related work to our research is the OAuth authorization framework. OAuth allows authentication of a user without exposing their account credentials to a third-party provider over HTTP protocol [18]. The main concept behind OAuth is that, when a user is trying to login to a third-party website, they can use their accounts' username/password from one of the major companies (e.g., Facebook, Twitter, LinkedIn) to authenticate themselves with a third-party company. With this method, users no longer have to reveal their login credentials with third-party companies. Currently, the deployments of OAuth framework have had limited exposure. This model has not been widely accepted as the alternative to hiding and protecting account credentials from third-party companies. As a result, OAuth currently suffers from penetration and adoption challenges, as well as privacy and security concerns.

9.2. Application Programming Interface

Another related work is where companies provide mechanisms for information retrieval and manipulation from their systems. This type of access is performed via application permission model [19] provided by Application Programming Interface (API) (e.g., Twitter API) [20]. This method is used to read, write, perform information retrieval and data mining to access and harvest data from companies (e.g., Twitter, Google) that support those APIs. The API may use OAuth technology as its underlying authorization mechanism to authenticate the request. Due to the nature and exposure of information provided with this model, it has its limit where sensitive information, security, and privacy are concerned.

10. CONCLUSIONS

The ubiquity of the Internet has increased the potential exploitation of security weaknesses and vulnerabilities of our financial institutions. Users are sharing their full access accounts' credentials with third-party companies and need to be wary of sharing this sensitive information with others, especially when those credentials can be used by others to execute transactions on their account. However, users' growing need of financial aggregation services from third-party providers, coupled with the lack of an alternate mechanism for account authentication and lack of restricted access control, can make users' vulnerable to attackers if their access credentials get compromised.

In this research paper, we have introduced and prescribed a novel and holistic model with a new protocol for online account architecture to be used by third-party companies. This model works on the notion of two security components: 1) Authentication mechanism utilizing new handshake protocol which enables verification of users' credentials that is more secure than a username/password combination; 2) User access sandboxing technique via granular access control that protects the account against unwanted transactions. Utilizing new architecture, design, and novel techniques we have presented in this paper, users no longer need to give out their full access accounts' credentials to third-parties. Instead, users can give limited alternate access login credentials for third-party use, and if attackers compromise their computer and access information is stolen, attackers cannot perform transactions on the account. In the case of a security breach, the alternate login credentials can be revoked and reissued with no impact on the existing full access account credentials. Furthermore, our novel and holistic techniques are universal and can be adapted for other domains (e.g., medical records, airline ticket system, online stores, emails) with little or no modifications to our architecture we have presented in this paper.

REFERENCES

[1] I. Jinwoo Hwang, "The Secure Sockets Layer and Transport Layer Security," Jun. 2012. [Online]. Available: http://www.ibm.com/developerworks/library/ws-ssl-security/index.html

[2] "Service Name and Transport Protocol Port Number Registry," 00017. [Online]. Available: http://www.iana.org/assignments/service-names-port-numbers/service-names-port-numbers.xhtml

[3] E. Rescorla, "HTTP Over TLS," Internet Requests for Comments, RFC Editor, RFC 2818, May 2000. [Online]. Available: http://www.rfc-editor.org/rfc/rfc2818.txt

[4] "ITU-T The Directory: Public-key and attribute certificate frameworks." [Online]. Available: http://www.itu.int/itu-t/recommendations/rec.aspx?rec=13031

[5] S. Chokhani, W. Ford, R. Sabett, C. Merrill, and S. Wu, "Internet X.509 Public Key Infrastructure Certificate Policy and Certification Practices Framework," Internet Requests for Comments, RFC Editor, RFC 3647, November 2003.

[6] D. Cooper, S. Santesson, S. Farrell, S. Boeyen, R. Housley, and W. Polk, "Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) profile," Internet Requests for Comments, RFC Editor, RFC 5280, May 2008, http://www.rfc-editor.org/rfc/rfc5280.txt. [Online]. Available: http://www.rfc-editor.org/rfc/rfc5280.txt

[7] A. Silberschatz, P. B. Galvin, and G. Gagne, Operating system concepts, 9th. Addison-Wesley Reading, 2013.

[8] G. S. Graham and P. J. Denning, "Protection: Principles and practice," in Proceedings of the May 16-18, 1972, Spring Joint Computer Conference, ser. AFIPS '72 (Spring). New York, NY, USA: ACM, 1972, pp. 417–429. [Online]. Available: http://doi.acm.org/10.1145/1478873.1478928

[9] M. G. Solomon and M. Chapple, Information security illuminated. Jones & Bartlett Publishers, 2009.

[10] D. F. Ferraiolo, R. Sandhu, S. Gavrila, D. R. Kuhn, and R. Chandramouli, "Proposed NIST Standard for Role-based Access Control," ACM Trans. Inf. Syst. Secur., vol. 4, no. 3, pp. 224–274, Aug. 2001. [Online]. Available: http://doi.acm.org/10.1145/501978.501980

[11] S. Fiegerman, “Yahoo says data stolen from 1 billion accounts,” Dec. 2016. [Online]. Available: http://money.cnn.com/2016/12/14/technology/yahoo-breach-billion-users/index.html

[12] S. Larson, “Hackers are selling Yahoo data on the dark web,” Dec. 2016. [Online]. Available: http://money.cnn.com/2016/12/16/technology/yahoo-for-sale-data-dark-web/index.html

[13] “Hacked home devices caused massive Internet outage.” [Online]. Available: http://www.usatoday.com/story/tech/2016/10/21/cyber-attack-takes-down-east-coast-netflix-spotify -twitter/92507806/

[14] “Logjam: the latest TLS vulnerability explained,” May 2015. [Online]. Available: http://blog.cloudflare.com/logjam-the-latest-tls-vulnerability-explained/

[15] J. Prins and B. U. Cybercrime, “DigiNotar certificate authority breach Operation Black Tulip,” 2011.

[16] K. Zetter, “DigiNotar files for bankruptcy in wake of devastating hack,” Wired magazine, September 2011.

[17] G. Huston and G. Michaelson, “The profile for algorithms and key sizes for use in the resource public key infrastructure,” Internet Requests for Comments, RFC Editor, RFC 7935, August 2016.

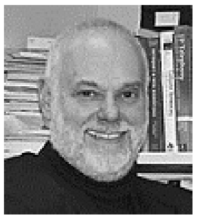
[18] “OAuth Community Site.” [Online]. Available: https://oauth.net/

[19] “Application Permission Model - Twitter Developers.” [Online]. Available: https://dev.twitter.com/oauth/overview/application-permission-model

[20] “Twitter Developer Documentation - Twitter Developers.” [Online]. Available: https://dev.twitter.com/docs

AUTHORS

Mehrdad M. Nourai received a B.S. degree in Electrical Engineering from Northeastern University, Boston Massachusetts in 1982, and an M.S. degree in Computer Science from Boston University, Boston Massachusetts in 2000. He is currently a Ph.D. degree candidate in Computer Science at theUniversity of Massachusetts Lowell, Lowell Massachusetts. He has over three decades of professional experience in computer industry and academics. His industry experience consists of architect, design, and development of software for all kinds of computer systems including embedded systems and systems with standard, proprietary, and real-time operating systems. He has been teaching computer science courses as an adjunct faculty for the MET Computer Science Department at Boston University since 2000. In addition, he has been teaching courses as an adjunct faculty for the Computer Science Department at Salem State University since 2008. His research and areas of interests include Computer Networks and Security, Data Communications, Human-Computer-Interaction, Database, and Mobile Apps Development.



Haim Levkowitz is the Chair of the Computer Science Department at the University of Massachusetts Lowell, in Lowell, MA, USA, where he has been a Faculty member since 1989. He was a twice-recipient of a US Fulbright Scholar Award to Brazil (August – December 2012 and August 2004 – January 2005). He was a Visiting Professor at ICMC — Instituto de Ciencias Matematicas e de Computacao (The Institute of Mathematics and Computer Sciences)—at the University of Sao Paul, Sao Carlos – SP, Brazil (August 2004 - August 2005; August 2012 to August 2013). He co-founded and was Co-Director of the Institute for Visualization and Perception Research (through 2012), and is now Director of the Human-Information Interaction Research Group. He is a world-renowned authority on visualization, perception, color, and their application in data mining and information retrieval. He is the author of “Color Theory and Modeling for Computer Graphics, Visualization, and Multimedia Applications” (Springer 1997) and co-editor of “Perceptual Issues in Visualization” (Springer 1995), as well as many papers on these subjects. He is also co-author/co-editor of "Writing Scientific Papers in English Successfully: Your Complete Roadmap," (E. Schuster, H. Levkowitz, and O.N. Oliveira Jr., eds., Paperback: ISBN: 978-8588533974; Kindle: ISBN: 8588533979, available now on Amazon.com:

http://www.amazon.com/Writing- Scientific-Papers-English- Successfully/dp/8588533979).

He has more than 44 years of experience teaching and lecturing, and has taught many tutorials and short courses, in addition to regular academic courses. In addition to his academic career, Professor Levkowitz has had an active entrepreneurial career as Founder or Co-Founder, Chief Technology Officer, Scientific and Strategic Advisor, Director, and venture investor at a number of high-tech startups.