Security Analysis of Atlas.mit.edu

Caroline Chin, Kelly Liu, Kevin Wang

Introduction

Atlas.mit.edu (Atlas) is a platform that connects MIT students, faculty, staff and other affiliates to relevant MIT applications. The goal is to serve as a one-stop-shop for all MIT-related affairs. In order to provide the information necessary to make Atlas user-friendly and convenient, the website contains a significant amount of personal information and direct links to users' accounts within different MIT modules.

Some of the data that Atlas holds include:

- **Financial information:** W-2 forms, tax information, bank account numbers etc.
- **Personal Information:** full social security number, home address, phone number
- **Links to:** User-authenticated reimbursement forms, time sheet entry

Due to the copious amount of data that Atlas stores about each user, a security vulnerability can allow a malicious user to pose a serious threat to private payment information and identity theft. To provide a sense of scale for the size of the population that Atlas serves, there are approximately 23,000 MIT affiliates (including students, faculty and staff). Among these are some of the world's most renowned scientists and engineers. Any data breach that exposes sensitive information about Atlas users can result in significant damage to both the individual and the MIT community as a whole. Ensuring the security of the platform is essential to protect the safety of MIT affiliates.

Due to the major risks associated with security vulnerabilities in the Atlas system, our goal is to conduct a comprehensive security analysis of the Atlas platform, and provide recommendations to mitigate vulnerabilities if any are found. This paper provides Atlas's security policy, the architectural overview of Atlas, a sampling of the attacks tried and the results that followed, as well as recommendations for how Atlas can improve its security.

Responsible Disclosure

This project seeks to provide an audit of the security of Atlas Release 8 (live on March 25th, 2016). The work conducted in this project was made possible due to the new MIT initiative called "The MIT Security Bug Bounty Program". The goal of this program is to "improve MIT's online security and foster a community for students to research and test the limits of cyber security in a responsible fashion". Upon the completion of this project, we will contact the developers of Atlas with our findings and recommendations before disclosing the security vulnerabilities with other members of the class. Publication of this report should be delayed until the beginning of the Fall semester in September 2016.

Security Policy

Atlas.mit.edu is a large and complex platform that contains a number of interconnected modules. For the purposes of this project, we focus mainly on the areas that contain sensitive personal data. As such, the modules, actions and principles described in the security policy contain only the relevant modules, and do not reflect user interactions with the system as a whole.

Modules

There exist three main modules with crucial private information in Atlas.

- **Personal Information**
    - This module refers to the panel displayed under the About Me → My Profile → Personal Information tab. This module contains a user's home address, date of birth, contact information, etc.
- **Money Matters**
    - This module refers to all pages that result from the links in the money matters section in the menu. These are: Paystubs, W-2s, Tax Withholding, Direct Deposit Preferences, Charitable Contributions
- **Reimbursements**
    - Technically part of the adminappsts page, this module contains all information regarding requesting, cloning, and tracking reimbursements.

Levels of Access

Below is a list of the different levels of access that may be present in Atlas.

- Personal Full Access
    - Viewing and Modifying privileges for the user's own data
- Personal Viewing Access
    - Viewing access only for the user's own data
- General Viewing Access
    - Viewing access only for all users' data

Principals

The set of permissible actions for each principal are outlined below. If a principal in question does not fall in one of the principals below, they have the permissions of unaffiliated individuals.

- **MIT Affiliates**: These are individuals who are affiliated to the MIT community and have access to a valid kerberos username and password.

  Permissible Actions:
  - Personal Full Access to Personal Information module
  - Personal Full Access to Reimbursements module. In this case, modifying privileges indicate that the user can create and edit RFPs.
  - Personal Viewing Access to Paystubs, W-2s, Tax Withholding, Charitable Contributions
  - Personal Full Access to Direct Deposit Preferences
  - No access to any module of other users

- **Administrators:** These are individuals that oversee the entire Atlas site. Unfortunately, information about administrator privileges is not publicly available, so we do not have a clear idea of an administrator's permissible actions.

- **Employers:** These are individuals that employ others and approve the timesheets of their employees.

  Permissible Actions:
  - Employers are provided with all permissible actions for MIT affiliates
  - Viewing access to timesheets and sick time reporting of their employees

- **Unaffiliated Individuals**: Individuals who do not have a working certificate or kerberos username and password

  Permissible Actions:
  - No access to any modules in the system beyond the Touchstone@MIT login page

Architecture Overview

Atlas consists of several different types of applications and frameworks, which are explained below. Understanding Atlas's architecture helped us identify aspects of Atlas that may have lower or higher security.

Touchstone: Primary Authentication

Atlas uses MIT Touchstone, a single sign-on web authentication service, to allow MIT affiliates to log on to the site. Touchstone uses either user certificates or kerberos/passwords to authenticate users and give them access to all MIT Touchstone related sites. From our research, we've found that Touchstone is highly secure and hard to breach[1]. First, Touchstone uses X.509 client certificates, which are difficult to duplicate or gain access to. The main way to reproduce a certificate is by having access to a person's identity information (such as a person's MIT ID number, kerberos username, and password), but that is also hard to do. Second, username and password verification passes information using TLS[2], which ensures that no third party can tamper with any message between the server and client. This means that a hacker can only try to brute force the right username password combination, which, given a limited amount of tries, is also hard to do.

Because of Atlas's strong authentication security, we decided to focus on attacks that could be done by someone that has access to a valid MIT certificate and can access Atlas using this certificate.

Struts2: Web Application Framework

Struts2 is an open source web application framework that uses an MVC model for making Java web applications. Struts2 makes rapid development of enterprise applications easier by offering modularity, abstraction, low coupling, and high cohesion[3]. In this framework, client requests are sent to Action Classes, which return the result code to the Controller, which then selects a view to render as the response.

We have chosen not to exploit security vulnerabilities in Struts2 because the latest version 2.3.28.3 does not contain any major known vulnerabilities. While there do exist critical vulnerabilities in past versions of Struts2[4], we decided not to look further into exploiting these because there is no public information regarding which version Atlas is currently using.

Other MIT Domains

Atlas also links to other domains, whose responses are displayed on the Atlas website. These domains include adminappsts and adminconnect. These domains take care of some of Atlas's features, such as the Charitable Contributions and RFP Tracking pages.

[1] https://ist.mit.edu/web/touchstone/faqs

[2] http://searchsecurity.techtarget.com/definition/Transport-Layer-Security-TLS

[3] http://www.jitendrazaa.com/blog/java/struts/what-is-struts-2-and-how-it-works/

[4] https://www.cvedetails.com/cve/CVE-2016-3081/

Previous Work

Over the course of the last 2 years, multiple patches and releases have been made to update Atlas. In the process, several potential security vulnerabilities have been incrementally addressed.

Release 3: Release 3 introduced two major improvements to security: Touchstone integration and security check for the Money Matters section. The addition of certificates and Touchstone, in lieu of presumably standard login, greatly enhances the login security of the Atlas portal. Requiring a security check (using the last four digits of the user's Social Security Number) creates a second layer of security in case the user's certificates are compromised. This security check helps guard some of the most sensitive information, including full Social Security Numbers and bank account/routing numbers.

Release 8: Cross Site Request Forgery (CSRF) was prevented with additional security enhancements to enforce more of the Same-Origin policy for requests. Likely enhancements include the addition of the referrer header, which cannot be spoofed in an CSRF attack.

Summary of Attacks

Request Replay: Replay attacks are defined as the copying of messages sent from client to server and the resending of those messages to the server[5]. If not protected against, the server can process the replayed messages as legitimate messages, which can cause anything from duplicated data to unauthorized access to sensitive information, such as banking information.

- **Attack Model:** Our attacker is someone who has access to another user's network requests to Atlas. They should not be able to replay these requests to receive the same personal information.
- **Strategy:** Assuming a user has logged in to Atlas through Touchstone and an adversary is listening to the user's messages to the server (via the Chrome Inspector Tools), we attempted replaying valid requests to the server. We performed this test under two conditions, (1) when the adversary was not logged in through Touchstone and (2) when the adversary had proper certificates and access to Atlas. For both tests, we included the necessary headers in the replayed requests, including cookies and referrer.
- **Result**: Neither version of the attack we attempted ran into any success. Both returned the response:

[5] https://msdn.microsoft.com/en-us/library/aa738652(v=vs.110).aspx

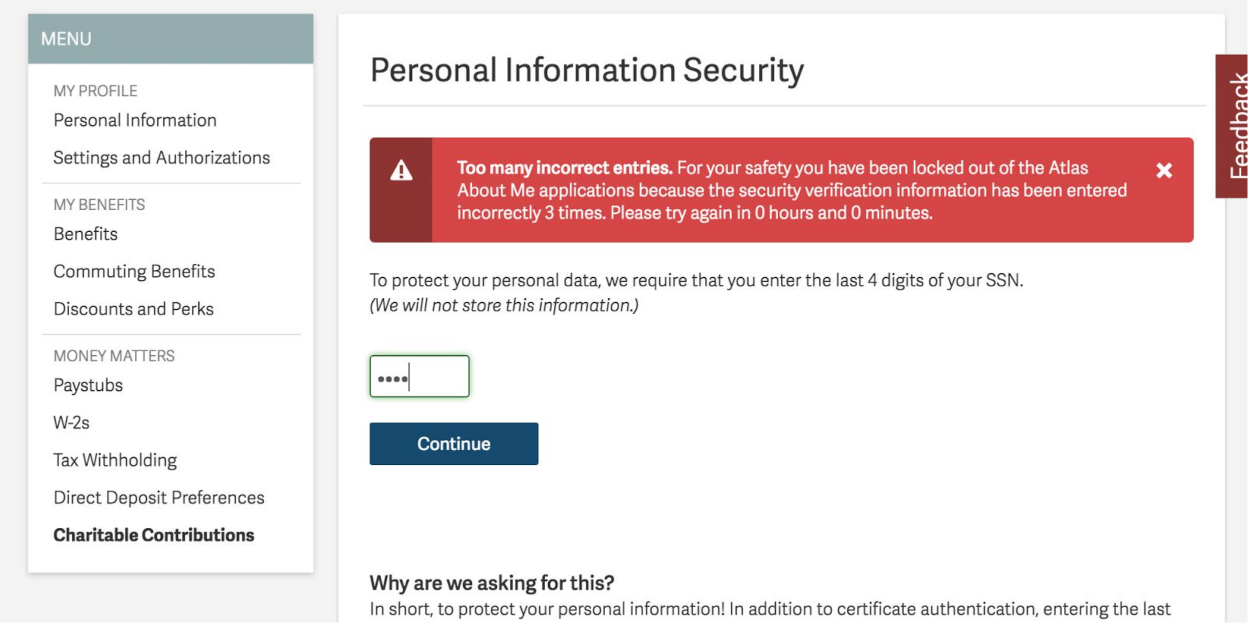
```
<html>
  <head>
    <title>Apache Tomcat/7.0.57 - Error report</title>
    ....
  </head>
  <body>
    <h1>HTTP Status 403 - </h1>
    ...
    <p>
      <b>description</b>
      <u>Access to the specified resource has been forbidden.</u>
    </p>
    <HR size="1" noshade="noshade">
    <h3>Apache Tomcat/7.0.57</h3>
  </body>
</html>
```

Evidently, somewhere along the line, the server performs some form of credential verification. Even with valid headers and cookies (including an EZproxy cookie), the Atlas server was able to differentiate between a legitimate, original attack and a replayed attack.

Brute forcing SSN: In Atlas, users must enter the last four digits of their social security number to access the Money Matters Module. When the entered numbers are verified, the user is then logged into all tabs in the Money Matters section. If an attacker could successfully guess the last four digits of the user's SSN, he/she could gain access to a user's banking information, tax withholding information, and W-2 forms.

- **Attack Model:** Our adversary is anyone that can gain access to a user's Atlas account(either by using that person's computer or getting the user to login to their Atlas account on a computer that the attacker has access to). The adversary has access to all Atlas functions that a user typically does after having logged in using a certificate.
- **Strategy:** The attacker navigates to the Money Matters tab under Settings and Authorizations. He/she can then try to brute force the SSN because there are only 10,000 combinations of the last four digits.
- **Results:** We soon found that Atlas only allows users to try up to three times total before locking them out of the Money Matters Module for four hours and sending an email to notify the account holder of the event. We did find that the Charitable Contributions tab exhibited strange behavior. The input form appeared each time the page was refreshed regardless of the number of tries, and the lockout time randomly changed, ranging from 0 minutes to 4 hours. One contributing factor for this behavior is that the methods used to validate the SSN for charitable contributions is different than the method used to validate SSN in other sections of Atlas. However, from our continued brute force efforts, we

concluded that these inconsistencies were only present on the frontend, but still performed the correct lockout functionality. Thus, because of the limited number of tries we had to test different numbers, this approach did not succeed.



Personal Information Security

Too many incorrect entries. For your safety you have been locked out of the Atlas About Me applications because the security verification information has been entered incorrectly 3 times. Please try again in 0 hours and 0 minutes.

To protect your personal data, we require that you enter the last 4 digits of your SSN.
(We will not store this information.)

Why are we asking for this?
In short, to protect your personal information! In addition to certificate authentication, entering the last

Figure 1. Screenshot of Charitable Contributions Page

SSN Sessions Vague Security Policy: To protect the security of a user's financial information in the Money Matters Module, Atlas performs a supposed session timeout on all Money Matters tabs after a predetermined amount of time. However, we noticed that the Money Matters Module sessions did not timeout after the browser was closed and instead seemed to be based off of a set amount of time.

- **Attack Model:** Our adversary is anyone that can get access to a user's computer or to a user's Atlas account. The adversary has full access to the user's Atlas page.
- **Strategy:** For this model, the attacker must be informed of the time at which a user logs into Atlas's Money Matters section. Then, after the user closes the tab and leaves the computer, the attacker can reopen the same page (with the valid credentials) and still have access to the user's private information.
- **Results:** We found that this approach would work, but only if 1) the attacker knows within the timeout period that a user has signed into the Money Matters section and 2) the attacker can access the user's Atlas page through valid verification. As a result, this approach is not easily scalable and could only really happen if the attacker was physically next to the user and could access the user's computer immediately after he/she has left the computer. Regardless, this security model does not provide users with maximum

expected security because Atlas does not timeout the SSN verification after a session ends, and this model should be changed.

SQL injection: SQL injection occurs when an attacker is able to submit SQL requests to the database through a user-input form. SQL injection vulnerabilities can occur when user-supplied data is not properly sanitized upon submission. As SQL injections are the second most common software vulnerability[6], it is important to ensure that Atlas does not expose any confidential information through this attack.

- **Attack Model**: Our adversary is an MIT affiliate who can authenticate themselves into the Atlas platform. They should be able to access information from the database only thorough developer-moderated methods. They should not be able to submit direct SQL queries into the database.
- **Strategy and Results:** A series of SQL injection attempts were made the people search form, with no success in exposing any contents of the database.

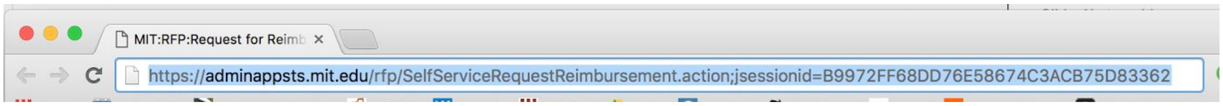
CORS for Cookies: CORS refers to a "Cross-Origin HTTP Request". A cross-origin request is made when the domain that a request is originating from is different from the target domain of the request[7]. Many browsers limit CORs requests to prevent malicious users from exploiting cookies or requests that should remain within the control of a specific domain. However, analysis of the Atlas source code reveals that cookies for Atlas are exposed to requests from any .mit.edu domain.

- **Attack Model:** Our attacker is an MIT affiliate who has developer access to an MIT domain and is able to view certain Atlas session cookies of other users. The attacker should not be able to view personal information about a user through these cookies, or use stolen cookies to authenticate themselves into a user's Atlas account.
- **Strategy:** A user authenticates themselves on Atlas. Before their Atlas session terminates, the user then navigates to a .mit.edu domian that is maintained by an attacker. The attacker now has access to the user's cookies from their Atlas session. The attacker then uses the cookie to authenticate themselves into the user's Atlas account.
- **Results:** While we were able to access a user's Atlas cookies from a different .mit.edu domain, we were not able to use the cookies to authenticate ourselves into Atlas, as we did not have the necessary Kerberos information/certificate. Additionally, no confidential user information was exposed in the cookie itself.

[6] http://www.veracode.com/security/sql-injection
[7] https://developer.mozilla.org/en-US/docs/Web/HTTP/Access_control_CORS

Cookie Exploitation in Reimbursements: Cookies are small pieces of texts, sent between web applications and clients that are used to authenticate users and record user preferences. This allows users to navigate to different sections of an application without logging in on every page.

- **Attack Model:** Our adversary is an MIT affiliate who can easily access user sessionids or URL requests. They should not be able to view a user's account by making a request to the same URL. They also should not be able to use the sessionid to authenticate themselves into a user's account.
- **Strategy:** When a user navigates to any page in the reimbursements module, their "session id" associated with their Atlas session is concatenated to the end of the URL (see figure 2 below). One feature of Atlas is that users do not have to sign in to their reimbursements page after signing into Atlas--the system directs a user directly to the reimbursement page associated with the user's account. An attacker can simply navigate to the URL and access the user's reimbursement information



https://adminappsts.mit.edu/rfp/SelfServiceRequestReimbursement.action;jsessionid=B9972FF68DD76E58674C3ACB75D83362

Figure 2. Screenshot of Reimbursement URL

- **Results:** Our attack was successful. We were able to submit the same URL containing the user's session id, authenticate ourselves using the our own kerberos information to view the reimbursements page of the original user. However, the sessionid only appears in the URL under specific circumstances, and we have not yet been able to replicate this behavior it consistently.

Next Steps

Our team performed a series of attacks on the Atlas platform and found the platform to be reasonably robust to our attacks. However, there are a number of more complex attacks that can potentially expose additional confidential user information. Some of the more promising possibilities are discussed in more detail in this section. Developers at Atlas are encouraged to ensure that the platform is secure to these attacks.

SSL Labs

Qualys SSL Labs is a service that runs a number of tests on a public domain to check for common security vulnerabilities. When querying atlas.mit.edu on the service, the resulting SSL

Report[8] rated Atlas with a “B”. Following up on the vulnerabilities exposed by the SSL Labs tests are outside the scope of this project and our understanding of network security, however exploring the issues raised in the report.

- Atlas accepts RC4 cipher but only with older protocol version[9]. There are known methods to attack the RC4 cipher, but currently they are expected to require significant amount of time and computational power. However, given recent advances in these attacks, researchers are expected to produce a more efficient algorithm to exploit RC4 in the near future. As such, SSL labs are recommending that servers discontinue the use of RC4 in TLS and use GCM suites instead.
- The server does not support Forward Secrecy with reference browsers. This is described in more detail in the Recommendations section, so will not be repeated here.

Recommendations

Sensitive Information in Plaintext:

Some requests to the server are returned with sensitive information in plaintext. For example, the ReadExpenseReimbursements.action request returns, if authenticated properly, the following HTML:

```
<div class="row panel-row">
         <div class="col-xs-6 col-sm-6">
        <label class="contentBoxLabel">Routing #:</label>
        <span id="reimbRoutingNumber-displayValue">
         012345678
        </span>
</div>
<div class="col-xs-6 col-sm-6">
         <label class="contentBoxLabel">Account #:</label>
        <span id="reimbAccountNo-displayValue">
         000123456789
         </span>
</div>
```

Unfortunately, any adversary listening to messages from the server will be able to directly read the account and routing numbers for all active users. Therefore, it would be prudent to encrypt this sensitive information when responding to client requests, using a simple public/private key scheme.

[8] https://www.ssllabs.com/ssltest/analyze.html?d=atlas.mit.edu
[9] https://blog.qualys.com/ssllabs/2013/03/19/rc4-in-tls-is-broken-now-what

Forward Secrecy: Atlas has yet to implement forward secrecy, which poses a relatively major security vulnerability. Defined as the guarantee that user data and session keys are secure even if the server's private key is compromised[10], forward secrecy prevents attackers from using compromised keys to decrypt previously recorded communications and sessions.

Atlas can easily be secured with forward secrecy by implementing two schemes:

1. Using random public keys for each session
2. Using non-deterministic encryption algorithms

An example of a non-deterministic encryption algorithm is Optimal Asymmetric Encryption Padding (OAEP) which, used with a one-way encryption scheme such as RSA, is IND-CPA secure. With this scheme, Atlas can secure all sessions against future compromisations of the system.

Cookie Exploitation in Reimbursements: The URL should not expose any information when a user is redirecting users to other MIT applications. This is particularly true in the reimbursements page, where the URL exposed a user's sessionid in plain text, allowing attackers to reuse the URL and access a user's financial information.

Charitable Contributions: The unexpected behavior in the timeout for SSN authentication in the charitable contributions page should be fixed to provide consistency for the user. This is largely due to the fact that the code for Charitable Contributions was written as part of adminappsts. To make the authentication verification consistent among all Money Matters tabs, we recommend using the same security method in Charitable Contributions as that used in the other sections.

Security Check Timeout: After successfully passing the Money Matters security check, the authorization persists for a certain amount of time in minutes, which provides convenience to the user. However, the authorization continues to persist even after the user's session with Atlas has ended and continues until that time limit is reached. Since the authorization is server-side, the authorization remains granted even if the user logs back in on a different machine, as long as it is within the time limit. As a result, an adversary with access to the user's certificates can piggyback off the legitimate user's security check and gain open access to the sensitive information in the Money Matters section without having to re-up the security check.

[10] https://scotthelme.co.uk/perfect-forward-secrecy/

The solution for this vulnerability is simple - reset the security check upon the user ending the Atlas session (when the user navigates away or closes the browser). This should override any built-in time limit on the authorization but the time limit should still exist to protect against prolonged sessions.

Works Cited

"Atlas Release Notes." *The Knowledge BASE*. MIT IS&T, 28 Mar. 2016. Web. 10 May 2016.

"Common Vulnerabilities and Exposures." *CVE*. N.p., n.d. Web. 10 May 2016.

Helme, Scott. "Perfect Forward Secrecy - An Introduction." *Scott Helme*. N.p., 10 May 2014. Web. 10 May 2016.

"HTTP Access Control (CORS)." *Mozilla Developer Network*. Mozilla, n.d. Web. 10 May 2016.

"Optimal Asymmetric Encryption Padding." *Wikipedia*. Wikimedia Foundation, n.d. Web. 10 May 2016.

"RC4 in TLS Is Broken: Now What?" *Qualys Blog*. N.p., 19 Mar. 2013. Web. 10 May 2016.

"Replay Attacks." *MSDN*. Microsoft, n.d. Web. 10 May 2016.

"SSL Report: Atlas.mit.edu." *Qualys SSL Labs*. N.p., n.d. Web. 10 May 2016.

"Touchstone at MIT: FAQ." *Information Systems & Technology*. MIT, n.d. Web. 10 May 2016.

"What Is Struts 2 and How It Works." *All about Web and Cloud*. N.p., 24 Mar. 2011. Web. 10 May 2016.

"What Is Transport Layer Security." *SearchSecurity*. N.p., n.d. Web. 10 May 2016.