WAS Web Application Report

QUALYSGUARD® ENTERPRISE SUITE

Web Application Report 11/03/2013

Each targeted web application is listed with the total number of detected vulnerabilities and sensitive content.

Andrew McDonnell
quays_am97

Qualys AJM

null, California null
United States of America

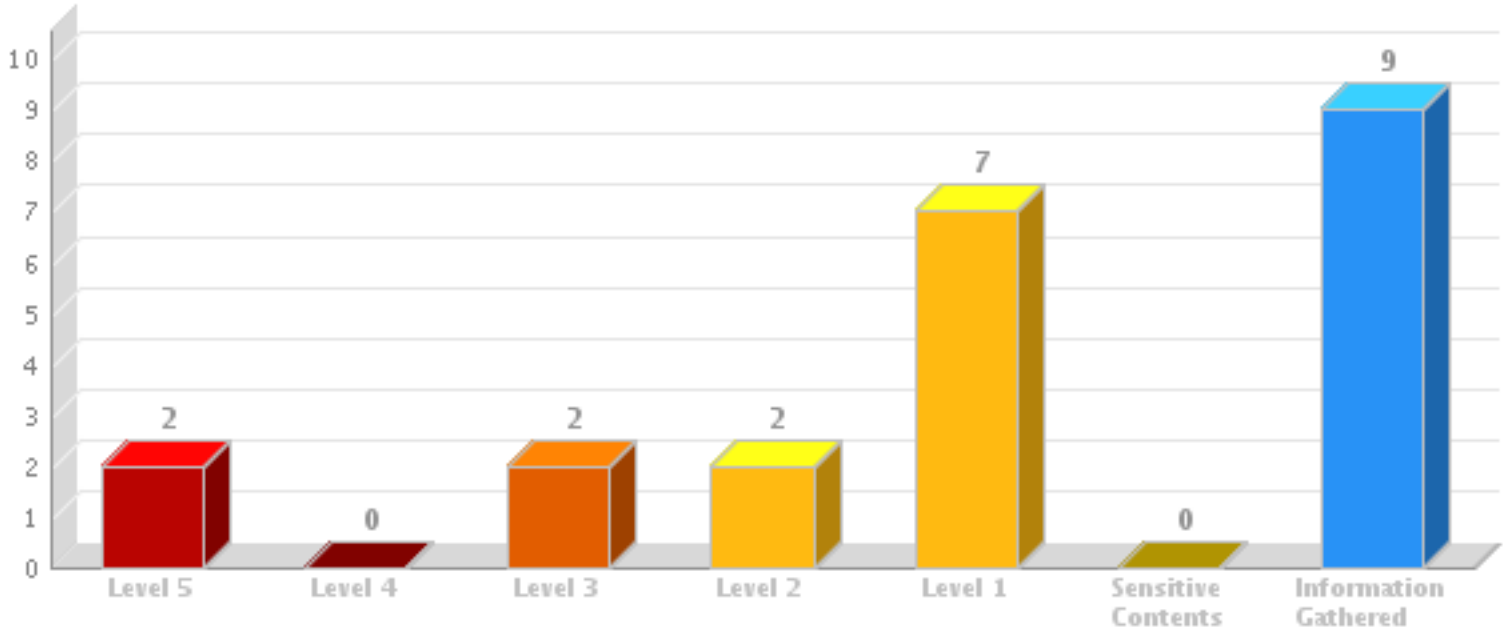
Target and Filters

Web Applications (1)	Google Gruyere
Status	New, Active, Re-Opened

Summary

Security Risk	Web Applications	Vulnerabilities	Sensitive Contents	Information Gathered
HIGH	1	13	0	9

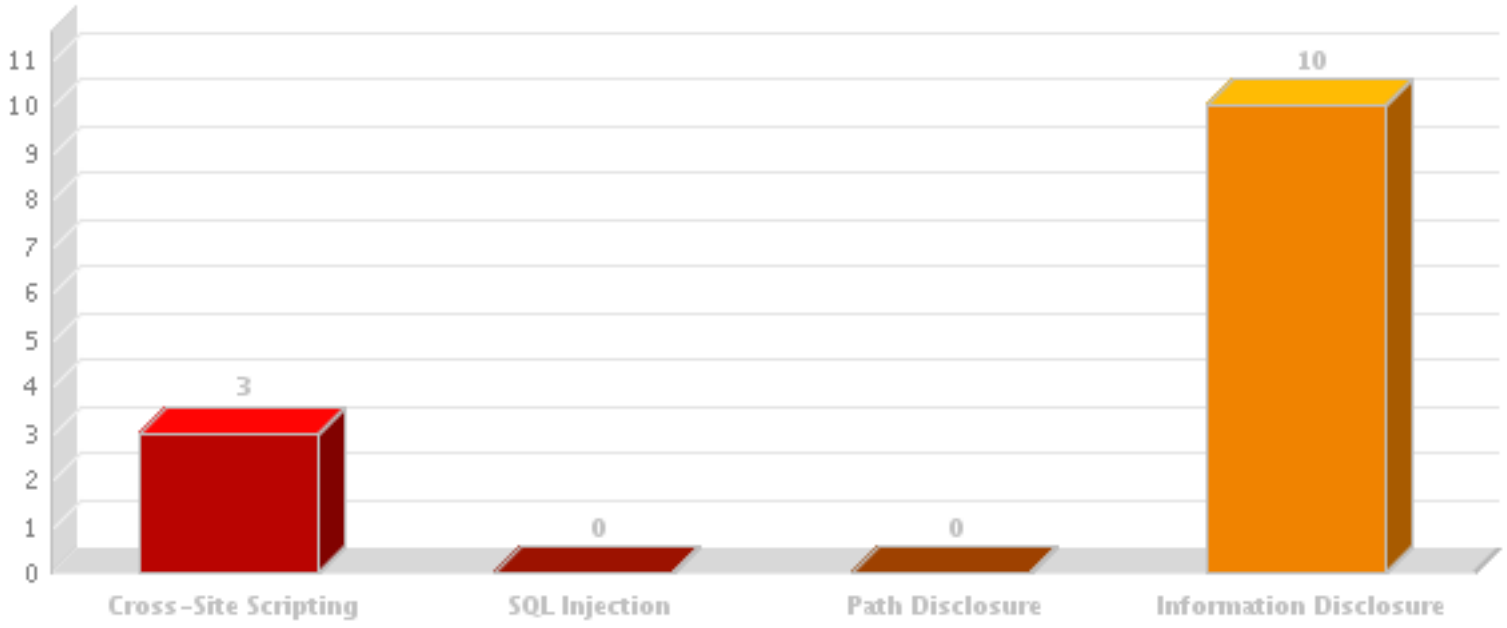
Findings by Severity



WAS Web Application Report

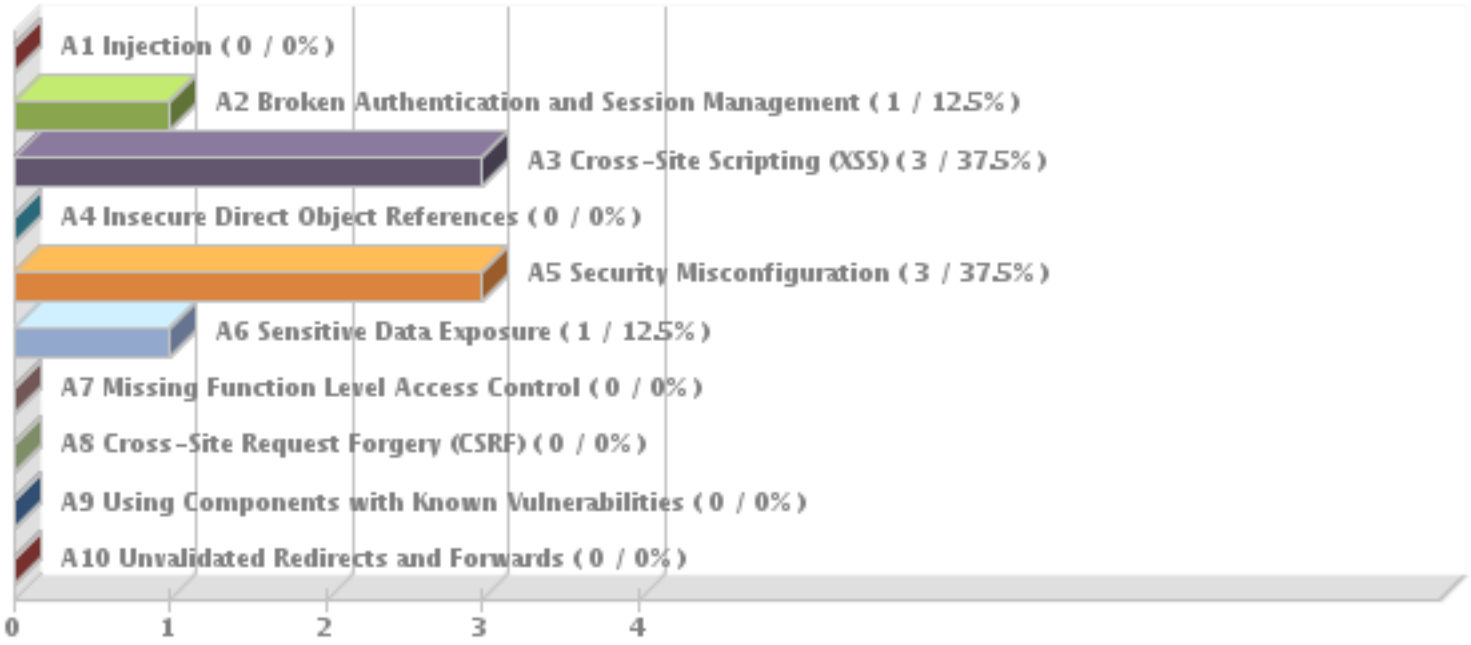
Vulnerabilities by Status

Vulnerabilities by Group



WAS Web Application Report

OWASP Top 10 2013 Detections



Web Application	Level 5	Level 4	Level 3	Level 2	Level 1	Sensitive Contents	Information Gathered
Google Gruyere	2	0	2	2	7	0	9

WAS Web Application Report

Results(22)

Vulnerability (13)

Cross-Site Scripting (3)

150001 Reflected Cross-Site Scripting (XSS) Vulnerabilities (2)

150001 Reflected Cross-Site Scripting (XSS) Vulnerabilities

Google Gruyere **New**

URL: http://google-gruyere.appspot.com/346700707728/feed.gtl?uid=%22'%3E%3Cqss%20a%3DX158242148Y1Z%3E

Finding #	1679538	**First Time Detected**	11/03/2013 23:32:23 GMT
Group	Cross-Site Scripting	**Last Time Detected**	11/03/2013 23:32:23 GMT
CWE	CWE-79	**Last Time Tested**	11/03/2013 23:32:23 GMT
OWASP	A3 Cross-Site Scripting (XSS) A2 Cross-Site Scripting (XSS)	**Times Detected**	1
WASP	WASC-8 Cross-Site Scripting		

Details

Threat

XSS vulnerabilities occur when the Web application echoes user-supplied data in an HTML response sent to the Web browser. For example, a Web application might include the user's name as part of a welcome message or display a home address when confirming a shipping destination. If the user-supplied data contain characters that are interpreted as part of an HTML element instead of literal text, then an attacker can modify the HTML that is received by the victim's Web browser.

The XSS payload is echoed in HTML document returned by the request. An XSS payload may consist of HTML, JavaScript or other content that will be rendered by the browser. In order to exploit this vulnerability, a malicious user would need to trick a victim into visiting the URL with the XSS payload.

Impact

XSS exploits pose a significant threat to a Web application, its users and user data. XSS exploits target the users of a Web application rather than the Web application itself. An exploit can lead to theft of the user's credentials and personal or financial information. Complex exploits and attack scenarios are possible via XSS because it enables an attacker to execute dynamic code. Consequently, any capability or feature available to the Web browser (for example HTML, JavaScript, Flash and Java applets) can be used to as a part of a compromise.

Solution

Filter all data collected from the client including user-supplied content and browser content such as Referrer and User-Agent headers.

Any data collected from the client and displayed in a Web page should be HTML-encoded to ensure the content is rendered as text instead of an HTML element or JavaScript.

Detection Information

Parameter	It has been detected by exploiting the parameter **uid** The payloads section will display a list of tests that show how the param could have been exploited to collect the information
Authentication	In order to detect this vulnerability, no authentication has been required.
Access Path	Here is the path followed by the scanner to reach the exploitable URL:

http://google-gruyere.appspot.com/346700707728/
http://google-gruyere.appspot.com/346700707728/snippets.gtl?uid=brie

Payloads

WAS Web Application Report

#1 Request

Payload uid=%22'%3E%3Cqss%20a%3DX158242452Y1Z%3E

Request GET http://google-gruyere.appspot.com/346700707728/feed.gtl?**uid=%22'%3E%3Cqss%20a%3DX158242452Y1Z%3E**

#1 Referer: http://google-gruyere.appspot.com/346700707728/

Click this link to try to reproduce the vulnerability using above payload.Note that clicking this link may not lead to visible results, either because the vulnerability requires context to be previously set (authentication, cookies...) or because the exploitation of the vulnerability does not lead to any visible proof.

#1 Response

```
_feed((

[
"'"><qss a=X158242452Y1Z>"
]


))
```

* The reflected string on the response webpage indicates that the vulnerability test was successful

150001 Reflected Cross-Site Scripting (XSS) Vulnerabilities

Google Gruyere **New**

URL: http://google-gruyere.appspot.com/346700707728/snippets.gtl?uid='%20onEvent%3DX158198304Y1Z%20

Finding #	1679543	**First Time Detected**	11/03/2013 23:32:23 GMT
Group	Cross-Site Scripting	**Last Time Detected**	11/03/2013 23:32:23 GMT
CWE	CWE-79	**Last Time Tested**	11/03/2013 23:32:23 GMT
OWASP	A3 Cross-Site Scripting (XSS) A2 Cross-Site Scripting (XSS)	**Times Detected**	1
WASP	WASC-8 Cross-Site Scripting		

Details

Threat

XSS vulnerabilities occur when the Web application echoes user-supplied data in an HTML response sent to the Web browser. For example, a Web application might include the user's name as part of a welcome message or display a home address when confirming a shipping destination. If the user-supplied data contain characters that are interpreted as part of an HTML element instead of literal text, then an attacker can modify the HTML that is received by the victim's Web browser.

The XSS payload is echoed in HTML document returned by the request. An XSS payload may consist of HTML, JavaScript or other content that will be rendered by the browser. In order to exploit this vulnerability, a malicious user would need to trick a victim into visiting the URL with the XSS payload.

Impact

XSS exploits pose a significant threat to a Web application, its users and user data. XSS exploits target the users of a Web application rather than the Web application itself. An exploit can lead to theft of the user's credentials and personal or financial information. Complex exploits and attack scenarios are possible via XSS because it enables an attacker to execute dynamic code. Consequently, any capability or feature available to the Web browser (for example HTML, JavaScript, Flash and Java applets) can be used to as a part of a compromise.

Solution

Filter all data collected from the client including user-supplied content and browser content such as Referrer and User-Agent headers.

Any data collected from the client and displayed in a Web page should be HTML-encoded to ensure the content is rendered as text instead of an HTML element or JavaScript.

WAS Web Application Report

Detection Information

Parameter	It has been detected by exploiting the parameter **uid** The payloads section will display a list of tests that show how the param could have been exploited to collect the information
Authentication	In order to detect this vulnerability, no authentication has been required.
Access Path	Here is the path followed by the scanner to reach the exploitable URL:

http://google-gruyere.appspot.com/346700707728/

Payloads

#1 Request

Payload	uid='%20onEvent%3DX158198304Y1Z%20
Request	GET http://google-gruyere.appspot.com/346700707728/snippets.gtl?**uid='%20onEvent%3DX158198304Y1Z%20**

#1 Referer: http://google-gruyere.appspot.com/346700707728/

Click this link to try to reproduce the vulnerability using above payload.Note that clicking this link may not lead to visible results, either because the vulnerability requires context to be previously set (authentication, cookies...) or because the exploitation of the vulnerability does not lead to any visible proof.

#1 Response

```
n in</a>
 | <a href='/346700707728/newaccount.gtl'>Sign up</a>

 </span>
</div>

<div>
<h2 class='has-refresh' id="user_name">

 ' onEvent=X158198304Y1Z



</h2>
<div class='refresh'><a class='button'
 onclick="_refreshSnippets("346700707728", "' onEvent=X158198304Y1Z ")'
 href='#'>Refresh</a></div>
<div class='content'>



 ' onEvent=X158198304Y1Z
 is not an author.




</div>
</body>
</html>
```

* The reflected string on the response webpage indicates that the vulnerability test was successful

150084 Unencoded characters (1)

150084 Unencoded characters

Google Gruyere **New**

URL: http://google-gruyere.appspot.com/346700707728/snippets.gtl?uid=%3C%0a%0dscript%20a%3D4%3Eqss%3D7%3C%0a%0d%2Fscript%3E

Finding #	1679542	**First Time Detected**	11/03/2013 23:32:23 GMT
Group	Cross-Site Scripting	**Last Time Detected**	11/03/2013 23:32:23 GMT
CWE	CWE-79	**Last Time Tested**	11/03/2013 23:32:23 GMT
OWASP	A3 Cross-Site Scripting (XSS) A2 Cross-Site Scripting (XSS)	**Times Detected**	1
WASP	WASC-20 Improper Input Handling		

WAS Web Application Report

Details

Threat

The web application reflects potentially dangerous characters such as single quotes, double quotes, and angle brackets. These characters are commonly used for HTML injection attacks such as cross-site scripting (XSS).

Impact

No exploit was determined for these reflected characters. The input parameter should be manually analyzed to verify that no other characters can be injected that would lead to an HTML injection (XSS) vulnerability.

Solution

Review the reflected characters to ensure that they are properly handled as defined by the web application's coding practice. Typical solutions are to apply HTML encoding or percent encoding to the characters depending on where they are placed in the HTML. For example, a double quote might be encoded as " when displayed in a text node, but as %22 when placed in the value of an href attribute.

Detection Information

Parameter	It has been detected by exploiting the parameter **uid** The payloads section will display a list of tests that show how the param could have been exploited to collect the information
Authentication	In order to detect this vulnerability, no authentication has been required.
Access Path	Here is the path followed by the scanner to reach the exploitable URL:

http://google-gruyere.appspot.com/346700707728/

Payloads

WAS Web Application Report

#1 Request

Payload uid=%22'%3E%3C%3CSCRIPT%20a%3D2%3Eqss%3D7%3B%2F%2F%3C%3C%2FSCRIPT%3E

Request GET http://google-gruyere.appspot.com/346700707728/snippets.gtl?**uid=%22'%3E%3C%3CSCRIPT%20a%3D2%3Eqss%3D7%3B%2F%2F%3C%3C%2FSCRIPT%3E**

#1 Referer: http://google-gruyere.appspot.com/346700707728/

Click this link to try to reproduce the vulnerability using above payload.Note that clicking this link may not lead to visible results, either because the vulnerability requires context to be previously set (authentication, cookies...) or because the exploitation of the vulnerability does not lead to any visible proof.

#1 Response

comment: A significant portion of the XSS test payload appeared in the web page, but the page's DOM was not modified as expected for a successful exploit. This result should be manually verified to determine its accuracy.

```
 </span>
 <span id='menu-right'>


 <a href='/346700707728/login'>Sign in</a>
 | <a href='/346700707728/newaccount.gtl'>Sign up</a>

 </span>
</div>

<div>
<h2 class='has-refresh' id="user_name">

 '"><<SCRIPT a=2>qss=7;//<</SCRIPT>



</h2>
<div class='refresh'><a class='button'
 onclick='_refreshSnippets("346700707728", '"><<SCRIPT a=2>qss=7;//<</SCRIPT>")'
 href='#'>Refresh</a></div>
<div class='content'>



 '"><<SCRIPT a=2>qss=7;//<</SCRIPT>
 is
```

* The reflected string on the response webpage indicates that the vulnerability test was successful

Information Disclosure (10)

150053 Login Form Is Not Submitted Via HTTPS (1)

150053 Login Form Is Not Submitted Via HTTPS — *Google Gruyere* **New**

URL: http://google-gruyere.appspot.com/346700707728/login

Finding #	1679546	**First Time Detected**	11/03/2013 23:32:23 GMT
Group	Information Disclosure	**Last Time Detected**	11/03/2013 23:32:23 GMT
CWE	-	**Last Time Tested**	11/03/2013 23:32:23 GMT
OWASP	A2 Broken Authentication and Session Management A6 Sensitive Data Exposure	**Times Detected**	1
WASP	-		

Details

Threat

The login form's default action contains a link that is not submitted via HTTPS (HTTP over SSL).

Impact

Sensitive data such as authentication credentials should be encrypted when transmitted over the network. Otherwise they are exposed to sniffing attacks.

Solution

Change the login form's action to submit via HTTPS.

Detection Information

Parameter	No param has been required for detecting the information.
Authentication	In order to detect this vulnerability, no authentication has been required.

Payloads

#1 Request

Payload	N/A
Request	GET http://google-gruyere.appspot.com/346700707728/login

Click this link to try to reproduce the vulnerability using above payload.Note that clicking this link may not lead to visible results, either because the vulnerability requires context to be previously set (authentication, cookies...) or because the exploitation of the vulnerability does not lead to any visible proof.

#1 Response

Login Form Is Not Submitted Via HTTPS

150085 Slow HTTP POST vulnerability (1)

150085 Slow HTTP POST vulnerability

Google Gruyere **New**

URL: http://google-gruyere.appspot.com/346700707728/

Finding #	1679540	**First Time Detected**	11/03/2013 23:32:23 GMT
Group	Information Disclosure	**Last Time Detected**	11/03/2013 23:32:23 GMT
CWE	-	**Last Time Tested**	11/03/2013 23:32:23 GMT
OWASP	A5 Security Misconfiguration A6 Security Misconfiguration	**Times Detected**	1
WASP	-		

Details

Threat

The web application is possibly vulnerable to a "slow HTTP POST" Denial of Service (DoS) attack. This is an application-level DoS that consumes server resources by maintaining open connections for an extended period of time by slowly sending traffic to the server. If the server maintains too many connections open at once, then it may not be able to respond to new, legitimate connections. Unlike bandwidth-consumption DoS attacks, the "slow" attack does not require a large amount of traffic to be sent to the server -- only that the client is able to maintain open connections for several minutes at a time. The attack holds server connections open by sending properly crafted HTTP POST headers that contain a Content-Length header with a large value to inform the web server how much of data to expect. After the HTTP POST headers are fully sent, the HTTP POST message body is sent at slow speeds to prolong the completion of the connection and lock up server resources. By waiting for the complete request body, the server is helping clients with slow or intermittent connections to complete requests, but is also exposing itself to abuse. More information can be found at the in this presentation.

Impact

All other services remain intact but the web server itself becomes inaccessible.

Solution

Solution would be server-specific, but general recommendations are: - to limit the size of the acceptable request to each form requirements - establish minimal acceptable speed rate - establish absolute request timeout for connection with POST request Server-specific details can be found here. A tool that demonstrates this vulnerability in a more intrusive manner is available here.

Detection Information

Parameter	No param has been required for detecting the information.
Authentication	In order to detect this vulnerability, no authentication has been required.

Payloads

#1 Request

Payload	N/A
Request	POST http://google-gruyere.appspot.com/346700707728/

Click this link to try to reproduce the vulnerability using above payload.Note that clicking this link may not lead to visible results, either because the vulnerability requires context to be previously set (authentication, cookies...) or because the exploitation of the vulnerability does not lead to any visible proof.

#1 Response

Vulnerable to slow HTTP POST attack
Connection with partial POST body remained open for: 311121 milliseconds

150112 Sensitive form field has not disabled autocomplete (2)

150112 Sensitive form field has not disabled autocomplete

Google Gruyere **New**

URL: http://google-gruyere.appspot.com/346700707728/login

Finding #	1679545	**First Time Detected**	11/03/2013 23:32:23 GMT
Group	Information Disclosure	**Last Time Detected**	11/03/2013 23:32:23 GMT
CWE	-	**Last Time Tested**	11/03/2013 23:32:23 GMT
OWASP	A5 Security Misconfiguration	**Times Detected**	1
WASP	-		

Details

Threat

An HTML form that collects sensitive information (such as a password field) does not prevent the browser from prompting the user to save the populated values for late reuse. Stored credentials should not be available to anyone but their owner.

Impact

If the browser is used in a shared computing environment where more than one person may use the browser, then "autocomplete" values may be submitted by an unauthorized user. For example, if a browser saves the login name and password for a form, then anyone with access to the browser may submit the form and authenticate to the site without having to know the victim's password.

Solution

Add the following attribute to the form or input element: autocomplete="off" This attribute prevents the browser from prompting the user to save the populated form values for later reuse.

Detection Information

Parameter	No param has been required for detecting the information.
Authentication	In order to detect this vulnerability, no authentication has been required.

Payloads

#1 Request

Payload	N/A
Request	GET http://google-gruyere.appspot.com/346700707728/login

Click this link to try to reproduce the vulnerability using above payload.Note that clicking this link may not lead to visible results, either because the vulnerability requires context to be previously set (authentication, cookies...) or because the exploitation of the vulnerability does not lead to any visible proof.

#1 Response

Form field does not set autocomplete="off".

150112 Sensitive form field has not disabled autocomplete

Google Gruyere **New**

URL: http://google-gruyere.appspot.com/346700707728/saveprofile

Finding #	1679535	**First Time Detected**	11/03/2013 23:32:23 GMT
Group	Information Disclosure	**Last Time Detected**	11/03/2013 23:32:23 GMT
CWE	-	**Last Time Tested**	11/03/2013 23:32:23 GMT
OWASP	A5 Security Misconfiguration	**Times Detected**	1
WASP	-		

Details

Threat

An HTML form that collects sensitive information (such as a password field) does not prevent the browser from prompting the user to save the populated values for late reuse. Stored credentials should not be available to anyone but their owner.

Impact

If the browser is used in a shared computing environment where more than one person may use the browser, then "autocomplete" values may be submitted by an unauthorized user. For example, if a browser saves the login name and password for a form, then anyone with access to the browser may submit the form and authenticate to the site without having to know the victim's password.

Solution

Add the following attribute to the form or input element: autocomplete="off" This attribute prevents the browser from prompting the user to save the populated form values for later reuse.

Detection Information

Parameter	No param has been required for detecting the information.
Authentication	In order to detect this vulnerability, no authentication has been required.

Payloads

#1 Request

Payload	N/A
Request	GET http://google-gruyere.appspot.com/346700707728/saveprofile

Click this link to try to reproduce the vulnerability using above payload.Note that clicking this link may not lead to visible results, either because the vulnerability requires context to be previously set (authentication, cookies...) or because the exploitation of the vulnerability does not lead to any visible proof.

#1 Response

Form field does not set autocomplete="off".

150081 Possible Clickjacking Vulnerability (6)

WAS Web Application Report

150081 Possible Clickjacking Vulnerability

Google Gruyere **New**

URL: http://google-gruyere.appspot.com/346700707728/

Finding #	1679541	**First Time Detected**	11/03/2013 23:32:23 GMT
Group	Information Disclosure	**Last Time Detected**	11/03/2013 23:32:23 GMT
CWE	-	**Last Time Tested**	11/03/2013 23:32:23 GMT
OWASP	-	**Times Detected**	1
WASP	-		

Details

Threat

An attack can trick the user into clicking on the link by framing the original page and showing a layer on top of it with dummy buttons.

Impact

Attacks like Cross-Site Request Forgery (CSRF) can be performed using Clickjacking techniques that frame a target site's content.

Solution

Two of the most popular preventions are: X-Frame-Options: This header works with modern browsers and can be used to prevent framing of the page. Note that is must be an HTTP header, the setting is ignored if it is created as an "http-equiv" meta element within the page. Framekiller: JavaScript code that prevents the malicious user from framing the page.

Detection Information

Parameter	No param has been required for detecting the information.
Authentication	In order to detect this vulnerability, no authentication has been required.

Payloads

#1 Request

Payload	N/A
Request	GET http://google-gruyere.appspot.com/346700707728/

Click this link to try to reproduce the vulnerability using above payload.Note that clicking this link may not lead to visible results, either because the vulnerability requires context to be previously set (authentication, cookies...) or because the exploitation of the vulnerability does not lead to any visible proof.

#1 Response

The response for this request did not have an "X-FRAME-OPTIONS" header present.

150081 Possible Clickjacking Vulnerability

Google Gruyere **New**

URL: http://google-gruyere.appspot.com/346700707728/feed.gtl

Finding #	1679536	**First Time Detected**	11/03/2013 23:32:23 GMT
Group	Information Disclosure	**Last Time Detected**	11/03/2013 23:32:23 GMT
CWE	-	**Last Time Tested**	11/03/2013 23:32:23 GMT
OWASP	-	**Times Detected**	1
WASP	-		

Details

Threat

An attack can trick the user into clicking on the link by framing the original page and showing a layer on top of it with dummy buttons.

Impact

Attacks like Cross-Site Request Forgery (CSRF) can be performed using Clickjacking techniques that frame a target site's content.

Solution

Two of the most popular preventions are: X-Frame-Options: This header works with modern browsers and can be used to prevent framing of the page. Note that is must be an HTTP header, the setting is ignored if it is created as an "http-equiv" meta element within the page. Framekiller: JavaScript code that prevents the malicious user from framing the page.

Detection Information

Parameter	No param has been required for detecting the information.
Authentication	In order to detect this vulnerability, no authentication has been required.
Access Path	Here is the path followed by the scanner to reach the exploitable URL:

http://google-gruyere.appspot.com/346700707728/

Payloads

#1 Request

Payload	N/A
Request	GET http://google-gruyere.appspot.com/346700707728/feed.gtl

Click this link to try to reproduce the vulnerability using above payload.Note that clicking this link may not lead to visible results, either because the vulnerability requires context to be previously set (authentication, cookies...) or because the exploitation of the vulnerability does not lead to any visible proof.

#1 Response

The response for this request did not have an "X-FRAME-OPTIONS" header present.

150081 Possible Clickjacking Vulnerability

Google Gruyere **New**

URL: http://google-gruyere.appspot.com/346700707728/feed.gtl?uid=cheddar

Finding #	1679539	**First Time Detected**	11/03/2013 23:32:23 GMT
Group	Information Disclosure	**Last Time Detected**	11/03/2013 23:32:23 GMT
CWE	-	**Last Time Tested**	11/03/2013 23:32:23 GMT
OWASP	-	**Times Detected**	1
WASP	-		

Details

Threat

An attack can trick the user into clicking on the link by framing the original page and showing a layer on top of it with dummy buttons.

Impact

Attacks like Cross-Site Request Forgery (CSRF) can be performed using Clickjacking techniques that frame a target site's content.

Solution

Two of the most popular preventions are: X-Frame-Options: This header works with modern browsers and can be used to prevent framing of the page. Note that is must be an HTTP header, the setting is ignored if it is created as an "http-equiv" meta element within the page. Framekiller: JavaScript code that prevents the malicious user from framing the page.

Detection Information

Parameter	No param has been required for detecting the information.
Authentication	In order to detect this vulnerability, no authentication has been required.
Access Path	Here is the path followed by the scanner to reach the exploitable URL:

WAS Web Application Report

http://google-gruyere.appspot.com/346700707728/
http://google-gruyere.appspot.com/346700707728/snippets.gtl?uid=cheddar

Payloads

#1 Request

Payload	N/A
Request	GET http://google-gruyere.appspot.com/346700707728/feed.gtl?uid=brie

Click this link to try to reproduce the vulnerability using above payload.Note that clicking this link may not lead to visible results, either because the vulnerability requires context to be previously set (authentication, cookies...) or because the exploitation of the vulnerability does not lead to any visible proof.

#1 Response

The response for this request did not have an "X-FRAME-OPTIONS" header present.

150081 Possible Clickjacking Vulnerability

Google Gruyere **New**

URL: http://google-gruyere.appspot.com/346700707728/login

Finding #	1679547	**First Time Detected**	11/03/2013 23:32:23 GMT
Group	Information Disclosure	**Last Time Detected**	11/03/2013 23:32:23 GMT
CWE	-	**Last Time Tested**	11/03/2013 23:32:23 GMT
OWASP	-	**Times Detected**	1
WASP	-		

Details

Threat

An attack can trick the user into clicking on the link by framing the original page and showing a layer on top of it with dummy buttons.

Impact

Attacks like Cross-Site Request Forgery (CSRF) can be performed using Clickjacking techniques that frame a target site's content.

Solution

Two of the most popular preventions are: X-Frame-Options: This header works with modern browsers and can be used to prevent framing of the page. Note that is must be an HTTP header, the setting is ignored if it is created as an "http-equiv" meta element within the page. Framekiller: JavaScript code that prevents the malicious user from framing the page.

Detection Information

Parameter	No param has been required for detecting the information.
Authentication	In order to detect this vulnerability, no authentication has been required.
Access Path	Here is the path followed by the scanner to reach the exploitable URL:

http://google-gruyere.appspot.com/346700707728/

Payloads

#1 Request

Payload	N/A
Request	GET http://google-gruyere.appspot.com/346700707728/login

Click this link to try to reproduce the vulnerability using above payload.Note that clicking this link may not lead to visible results, either because the vulnerability requires context to be previously set (authentication, cookies...) or because the exploitation of the vulnerability does not lead to any visible proof.

#1 Response

The response for this request did not have an "X-FRAME-OPTIONS" header present.

150081 Possible Clickjacking Vulnerability

Google Gruyere **New**

URL: http://google-gruyere.appspot.com/346700707728/newaccount.gtl

Finding #	1679537	**First Time Detected**	11/03/2013 23:32:23 GMT
Group	Information Disclosure	**Last Time Detected**	11/03/2013 23:32:23 GMT
CWE	-	**Last Time Tested**	11/03/2013 23:32:23 GMT
OWASP	-	**Times Detected**	1
WASP	-		

Details

Threat

An attack can trick the user into clicking on the link by framing the original page and showing a layer on top of it with dummy buttons.

Impact

Attacks like Cross-Site Request Forgery (CSRF) can be performed using Clickjacking techniques that frame a target site's content.

Solution

Two of the most popular preventions are: X-Frame-Options: This header works with modern browsers and can be used to prevent framing of the page. Note that is must be an HTTP header, the setting is ignored if it is created as an "http-equiv" meta element within the page. Framekiller: JavaScript code that prevents the malicious user from framing the page.

Detection Information

Parameter	No param has been required for detecting the information.
Authentication	In order to detect this vulnerability, no authentication has been required.
Access Path	Here is the path followed by the scanner to reach the exploitable URL:

http://google-gruyere.appspot.com/346700707728/

Payloads

#1 Request

Payload	N/A
Request	GET http://google-gruyere.appspot.com/346700707728/newaccount.gtl

Click this link to try to reproduce the vulnerability using above payload.Note that clicking this link may not lead to visible results, either because the vulnerability requires context to be previously set (authentication, cookies...) or because the exploitation of the vulnerability does not lead to any visible proof.

#1 Response

The response for this request did not have an "X-FRAME-OPTIONS" header present.

150081 Possible Clickjacking Vulnerability

Google Gruyere **New**

WAS Web Application Report

URL: http://google-gruyere.appspot.com/346700707728/snippets.gtl?uid=cheddar

Finding #	1679544	**First Time Detected**	11/03/2013 23:32:23 GMT
Group	Information Disclosure	**Last Time Detected**	11/03/2013 23:32:23 GMT
CWE	-	**Last Time Tested**	11/03/2013 23:32:23 GMT
OWASP	-	**Times Detected**	1
WASP	-		

Details

Threat
An attack can trick the user into clicking on the link by framing the original page and showing a layer on top of it with dummy buttons.

Impact
Attacks like Cross-Site Request Forgery (CSRF) can be performed using Clickjacking techniques that frame a target site's content.

Solution
Two of the most popular preventions are: X-Frame-Options: This header works with modern browsers and can be used to prevent framing of the page. Note that is must be an HTTP header, the setting is ignored if it is created as an "http-equiv" meta element within the page. Framekiller: JavaScript code that prevents the malicious user from framing the page.

Detection Information

Parameter	No param has been required for detecting the information.
Authentication	In order to detect this vulnerability, no authentication has been required.
Access Path	Here is the path followed by the scanner to reach the exploitable URL:

http://google-gruyere.appspot.com/346700707728/

Payloads

#1 Request

Payload	N/A
Request	GET http://google-gruyere.appspot.com/346700707728/snippets.gtl?uid=cheddar

Click this link to try to reproduce the vulnerability using above payload.Note that clicking this link may not lead to visible results, either because the vulnerability requires context to be previously set (authentication, cookies...) or because the exploitation of the vulnerability does not lead to any visible proof.

#1 Response

The response for this request did not have an "X-FRAME-OPTIONS" header present.

Information Gathered (9)

No Group (9)

150067 Links Discovered During User-Agent and Mobile Site Checks (1)

150067 Links Discovered During User-Agent and Mobile Site Checks

Google Gruyere

Finding #	531916	**First Time Detected**	11/03/2013 23:32:23 GMT
Group	Information Gathered	**Last Time Detected**	11/03/2013 23:32:23 GMT
CWE	-	**Last Time Tested**	11/03/2013 23:32:23 GMT
OWASP	-		
WASP	-		

Details

Threat

Links were discovered via requests using an alternate User-Agent or guessed based on common mobile device URI patterns. The scanner attempts to determine if the Web application changes its behavior when accessed by mobile devices. These checks are based on modifying the User-Agent, changing the domain name, and appending common directories.

The extra links discovered by the Web application scanner during User-Agent manipulation are provided in the Results section.

Impact

The Web application should apply consistent security measures irrespective of browser platform, type or version used to access the application. If the Web application fails to apply security controls to alternate representations of the site, then it may be exposed to vulnerabilities like cross-site scripting, SQL injection, or authorization-based attacks.

Solution

No specific vulnerability has been discovered that requires action to be taken. These links are provided to ensure that a review of the web application includes all possible access points.

Results

Unique content discovered during user-agent and common mobile device specific subdomains and paths manipulation: User-Agent: Mozilla/5.0 (iPhone; U; CPU iPhone OS 3_1_2 like Mac OS X; en-us) AppleWebKit/528.18 (KHTML, like Gecko) Version/4.0 Mobile/7D11 Safari/528.16
http://google-gruyere.appspot.com/346700707728/mobile
User-Agent: Opera/9.80 (IPhone; Opera Mini/5.0.019802/886; U; en) Presto/2.4.15
http://google-gruyere.appspot.com/346700707728/mobile

150086 Server accepts unnecessarily large POST request body (1)

150086 Server accepts unnecessarily large POST request body

Google Gruyere

Finding #	531914	**First Time Detected**	11/03/2013 23:32:23 GMT
Group	Information Gathered	**Last Time Detected**	11/03/2013 23:32:23 GMT
CWE	-	**Last Time Tested**	11/03/2013 23:32:23 GMT
OWASP	A5 Security Misconfiguration		
WASP	-		

Details

Threat

Web application scanner successfully sent a POST request with content type of application/x-www-form-urlencoded and 65536 bytes length random text data. Accepting request bodies with unnecessarily large size could help attacker to use less connections to achieve Layer 7 DDoS of web server. More information can be found at the here

Impact

Could result in successful application level (Layer 7) DDoS attack.

Solution

Limit the size of the request body to each form's requirements. For example, a search form with 256-char search field should not accept more than 1KB value. Server-specific details can be found here.

Results

Server responded 200 to unnecessarily large random request body(over 64 KB) for URL http://google-gruyere.appspot.com/346700707728/, significantly increasing attacker's chances to prolong slow HTTP POST attack.

6 DNS Host Name (1)

6 DNS Host Name

Google Gruyere

Finding #	531920	**First Time Detected**	11/03/2013 23:32:23 GMT
Group	Information Gathered	**Last Time Detected**	11/03/2013 23:32:23 GMT
CWE	-	**Last Time Tested**	11/03/2013 23:32:23 GMT
OWASP	-		
WASP	-		

Details

Threat

The fully qualified domain name of this host, if it was obtained from a DNS server, is displayed in the RESULT section.

Results

IP address	Host name
74.125.28.141	pc-in-f141.1e100.net

45038 Host Scan Time (1)

45038 Host Scan Time

Google Gruyere

Finding #	531918	**First Time Detected**	11/03/2013 23:32:23 GMT
Group	Information Gathered	**Last Time Detected**	11/03/2013 23:32:23 GMT
CWE	-	**Last Time Tested**	11/03/2013 23:32:23 GMT
OWASP	-		
WASP	-		

Details

Threat

The Host Scan Time is the period of time it takes the scanning engine to perform the vulnerability assessment of a single target host. The Host Scan Time for this host is reported in the Result section below.

The Host Scan Time does not have a direct correlation to the Duration time as displayed in the Report Summary section of a scan results report. The Duration is the period of time it takes the service to perform a scan task. The Duration includes the time it takes the service to scan all hosts, which may involve parallel scanning. It also includes the time it takes for a scanner appliance to pick up the scan task and transfer the results back to the service's Secure Operating Center. Further, when a scan task is distributed across multiple scanners, the Duration includes the time it takes to perform parallel host scanning on all scanners.

Impact

N/A

Solution

N/A

Results

Scan duration: 500 seconds

Start time: Sun, Nov 03 2013, 23:23:28 GMT

End time: Sun, Nov 03 2013, 23:31:48 GMT

150009 Links Crawled (1)

150009 Links Crawled

Google Gruyere

Finding #	531917	**First Time Detected**	11/03/2013 23:32:23 GMT
Group	Information Gathered	**Last Time Detected**	11/03/2013 23:32:23 GMT
CWE	-	**Last Time Tested**	11/03/2013 23:32:23 GMT
OWASP	-		
WASP	-		

Details

Threat

The list of unique links crawled by the Web application scanner appear in the Results section. This list may contain fewer links than the maximum threshold defined at scan launch. The maximum links to crawl includes links in this list, requests made via HTML forms, and requests for the same link made as an anonymous and authenticated user.

Impact

N/A

Solution

N/A

Results

Duration of crawl phase (seconds): 115.00
Number of links: 9
(This number excludes form requests and links re-requested during authentication.)

http://google-gruyere.appspot.com/346700707728/
http://google-gruyere.appspot.com/346700707728/feed.gtl
http://google-gruyere.appspot.com/346700707728/feed.gtl?uid=brie
http://google-gruyere.appspot.com/346700707728/feed.gtl?uid=cheddar
http://google-gruyere.appspot.com/346700707728/lib.js
http://google-gruyere.appspot.com/346700707728/login
http://google-gruyere.appspot.com/346700707728/newaccount.gtl
http://google-gruyere.appspot.com/346700707728/snippets.gtl?uid=brie
http://google-gruyere.appspot.com/346700707728/snippets.gtl?uid=cheddar

150010 External Links Discovered (1)

150010 External Links Discovered

Google Gruyere

Finding #	531915	**First Time Detected**	11/03/2013 23:32:23 GMT
Group	Information Gathered	**Last Time Detected**	11/03/2013 23:32:23 GMT
CWE	-	**Last Time Tested**	11/03/2013 23:32:23 GMT
OWASP	-		
WASP	-		

Details

Threat

The external links discovered by the Web application scanning engine are provided in the Results section. These links were present on the target Web application, but were not crawled.

Impact

N/A

Solution

N/A

Results

Number of links: 2
http://news.google.com/news/search?q=brie
http://images.google.com/images?q=cheddar+cheese

150020 Links Rejected By Crawl Scope or Exclusion List (1)

150020 Links Rejected By Crawl Scope or Exclusion List

Google Gruyere

Finding #	531919	**First Time Detected**	11/03/2013 23:32:23 GMT
Group	Information Gathered	**Last Time Detected**	11/03/2013 23:32:23 GMT
CWE	-	**Last Time Tested**	11/03/2013 23:32:23 GMT
OWASP	-		
WASP	-		

Details

Threat

One or more links were not crawled because of an explicit rule to exclude them. This also occurs if a link is malformed.

Black list and white list entries can cause links to be rejected. If a scan is limited to a specific starting directory, then links outside that directory will neither be crawled or tested.

Links that contain a host name or IP address different from the target application are considered external links and not crawled by default; those types of links are not listed here. This often happens when the scope of a scan is limited to the directory of the starting URL. The scope can be changed in the Web Application Record.

During the test phase, some path-based tests may be rejected if the scan is limited to the directory of the starting URL and the test would fall outside that directory. In these cases, the number of rejected links may be too high to list in the Results section.

Impact

Links listed here were neither crawled or tested by the Web application scanning engine.

Solution

A link might have been intentionally matched by a black or white list entry. Verify that no links in this list were unintentionally rejected.

Results

http://google-gruyere.appspot.com/crossdomain.xml
Links rejected during the test phase not reported due to volume of links.

150021 Scan Diagnostics (1)

150021 Scan Diagnostics

Google Gruyere

Finding #	531922	**First Time Detected**	11/03/2013 23:32:23 GMT
Group	Information Gathered	**Last Time Detected**	11/03/2013 23:32:23 GMT
CWE	-	**Last Time Tested**	11/03/2013 23:32:23 GMT
OWASP	-		
WASP	-		

WAS Web Application Report

Details

Threat

This check provides various details of the scan's performance and behavior. In some cases, this check can be used to identify problems that the scanner encountered when crawling the target Web application.

Impact

The scan diagnostics data provides technical details about the crawler's performance and behavior. This information does not necessarily imply problems with the Web application.

Solution

No action is required.

Results

Loaded 0 blacklist entries.
Loaded 0 whitelist entries.
HTML form authentication unavailable, no WEBAPP entry found
Collected 10 links overall.
Path manipulation: estimated time < 1 minute (115 tests, 9 inputs)
Path manipulation: 115 vulnsigs tests, completed 183 requests, 5 seconds. All tests completed.
WS enumeration: estimated time < 1 minute (9 tests, 71 inputs)
WS enumeration: 9 vulnsigs tests, completed 567 requests, 11 seconds. All tests completed.
Batch #1 URI parameter manipulation (no auth): estimated time < 1 minute (46 tests, 0 inputs)
Batch #1 URI parameter manipulation (no auth): 46 vulnsigs tests, completed 92 requests, 2 seconds. No tests to execute.
Batch #1 Form parameter manipulation (no auth): estimated time < 1 minute (46 tests, 1 inputs)
Batch #1 Form parameter manipulation (no auth): 46 vulnsigs tests, completed 92 requests, 2 seconds. All tests completed.
Batch #1 Login form parameter manipulation (no auth): estimated time < 1 minute (46 tests, 1 inputs)
Batch #1 Login form parameter manipulation (no auth): 46 vulnsigs tests, completed 276 requests, 5 seconds. No tests to execute.
Batch #1 URI blind SQL manipulation (no auth): estimated time < 1 minute (19 tests, 0 inputs)
Batch #1 URI blind SQL manipulation (no auth): 19 vulnsigs tests, completed 38 requests, 2 seconds. No tests to execute.
Batch #1 Form blind SQL manipulation (no auth): estimated time < 1 minute (19 tests, 1 inputs)
Batch #1 Form blind SQL manipulation (no auth): 19 vulnsigs tests, completed 38 requests, 2 seconds. All tests completed.
Batch #1 Login form blind SQL manipulation (no auth): estimated time < 1 minute (19 tests, 1 inputs)
Batch #1 Login form blind SQL manipulation (no auth): 19 vulnsigs tests, completed 114 requests, 10 seconds. All tests completed.
URI parameter time-based tests (no auth): estimated time < 1 minute (8 tests, 0 inputs)
Batch #1 URI parameter time-based tests (no auth): 8 vulnsigs tests, completed 16 requests, 4 seconds. No tests to execute.
Form field time-based tests (no auth): estimated time < 1 minute (8 tests, 0 inputs)
Batch #1 Form field time-based tests (no auth): 8 vulnsigs tests, completed 16 requests, 2 seconds. No tests to execute.
Batch #2 URI parameter manipulation (no auth): estimated time < 1 minute (46 tests, 0 inputs)
Batch #2 URI parameter manipulation (no auth): 46 vulnsigs tests, completed 44 requests, 1 seconds. No tests to execute.
Batch #2 URI blind SQL manipulation (no auth): estimated time < 1 minute (19 tests, 0 inputs)
Batch #2 URI blind SQL manipulation (no auth): 19 vulnsigs tests, completed 38 requests, 2 seconds. No tests to execute.
URI parameter time-based tests (no auth): estimated time < 1 minute (8 tests, 0 inputs)
Batch #2 URI parameter time-based tests (no auth): 8 vulnsigs tests, completed 16 requests, 3 seconds. No tests to execute.
HTTP call manipulation: estimated time < 1 minute (33 tests, 0 inputs)
HTTP call manipulation: 33 vulnsigs tests, completed 0 requests, 0 seconds. No tests to execute.
Open Redirect analysis: estimated time < 1 minute (1 tests, 0 inputs)
Open Redirect analysis: 1 vulnsigs tests, completed 0 requests, 0 seconds. No tests to execute.
CSRF: estimated time < 1 minute (2 tests, 2 inputs)
CSRF: 2 vulnsigs tests, completed 0 requests, 0 seconds. No tests to execute.
Batch #4 File Inclusion analysis: estimated time < 1 minute (1 tests, 9 inputs)
Batch #4 File Inclusion analysis: 1 vulnsigs tests, completed 0 requests, 0 seconds. All tests completed.
Cookie manipulation: estimated time < 1 minute (37 tests, 0 inputs)
Cookie manipulation: 37 vulnsigs tests, completed 0 requests, 0 seconds. No tests to execute.
Header manipulation: estimated time < 1 minute (37 tests, 8 inputs)
Header manipulation: 37 vulnsigs tests, completed 192 requests, 4 seconds. XSS optimization removed 192 links. Completed 192 requests of 592 estimated requests (32%). All tests completed.
Login Brute Force manipulation: estimated time < 1 minute (174 tests, 1 inputs)
Login Brute Force manipulation: 174 vulnsigs tests, completed 174 requests, 158 seconds. All tests completed.
Total requests made: 2204
Average server response time: 0.23 seconds
Most recent links:
200 http://google-gruyere.appspot.com/346700707728/login?uid=nogood&pw=+&uid=websecadm&pw=websecadm
200 http://google-gruyere.appspot.com/346700707728/login?uid=nogood&pw=+&uid=websecadm&pw=
200 http://google-gruyere.appspot.com/346700707728/login?uid=nogood&pw=+&uid=write&pw=write
200 http://google-gruyere.appspot.com/346700707728/login?uid=nogood&pw=+&uid=write&pw=
200 http://google-gruyere.appspot.com/346700707728/
200 http://google-gruyere.appspot.com/346700707728/
200 http://google-gruyere.appspot.com/346700707728/lib.js
200 http://google-gruyere.appspot.com/346700707728/login
200 http://google-gruyere.appspot.com/346700707728/newaccount.gtl
200 http://google-gruyere.appspot.com/346700707728/snippets.gtl?uid=brie

WAS Web Application Report

150115 Authentication Form found (1)

150115 Authentication Form found

Google Gruyere

Finding #	531921	**First Time Detected**	11/03/2013 23:32:23 GMT
Group	Information Gathered	**Last Time Detected**	11/03/2013 23:32:23 GMT
CWE	-	**Last Time Tested**	11/03/2013 23:32:23 GMT
OWASP	-		
WASP	-		

Details

Threat

Authentication Form was found during the web application crawling.

Impact

N/A

Solution

N/A

Results

Authentication form found at: http://google-gruyere.appspot.com/346700707728/login
Action uri: http://google-gruyere.appspot.com/346700707728/login
Fields: uid, pw, WasNoName_S_2_LoginAuthentication form found at: http://google-gruyere.appspot.com/346700707728/newaccount.gtl
Action uri: http://google-gruyere.appspot.com/346700707728/saveprofile
Fields: action, uid, is_author, pw,

WAS Web Application Report

Appendix

Web Application Details
Google Gruyere

Name	Google Gruyere
URL	http://google-gruyere.appspot.com/346700707728/
Owner	Andrew McDonnell (quays_am97)
Scope	Limit to content located at or below URL subdirectory
Operating System	-