Proxy Caches and Web Application Security

Using the Recent Google Docs 0-Day as an Example

Tim Bass, CISSP
Chapter Leader, OWASP Thailand
+66832975101, tim@unix.com

OWASP AppSec Asia
October 21, 2008

OWASP Thailand

OWASP
The Open Web Application Security Project

9671 chapter members worldwide
4940 project members worldwide

www.owasp.org

My Contact Info and Web Places

Me	Tim Bass
Mobile, Thailand	+66832975101
Email	tim@unix.com
Blog – The CEP Blog	www.thecepblog.com
Blog – The (ISC)2 Blog	blog.isc2.org
ACIS Professional Center	www.acisonline.net
The UNIX and Linux Forums	www.unix.com
LinkedIn	www.linkedin.com/in/timbass

Our Agenda

- First, A Brief Review of the OWASP Top 10
  - #7. Broken Authentication and Session Management
- Second, A Funny Thing Happened in GoogleDocs
- Third, Proxy Caches are a Serious Threat
  - Poorly written session management code is the vulnerability
  - Simple testing scenario(s)
  - ... and a warning ....

OWASP Top 10 2007

1. Cross Site Scripting (XSS)
2. Injection Flaws
3. Insecure Remote File Include
4. Insecure Direct Object Reference
5. Cross Site Request Forgery (CSRF)
6. Information Leakage and Improper Error Handling
7. Broken Authentication and Session Management
8. Insecure Cryptographic Storage
9. Insecure Communications
10. Failure to Restrict URL Access

http://www.owasp.org/index.php/Top_10

OWASP Top 10 2007

1. Cross Site Scripting (XSS)
2. Injection Flaws
3. Insecure Remote File Include
4. Insecure Direct Object Reference
5. Cross Site Request Forgery (CSRF)
6. Information Leakage and Improper Error Handling
7. **Broken Authentication and Session Management**
8. Insecure Cryptographic Storage
9. Insecure Communications
10. Failure to Restrict URL Access

http://www.owasp.org/index.php/Top_10

Brief OWASP Top 10 Review

7. Broken Authentication and Session Management

7. Broken Authentication and Session Management

- Description
  - Flaws in HTTP authentication and session management frequently involve the failure to protect credentials and session tokens through their lifecycle.

- Affected Environments
  - All web application frameworks are vulnerable to authentication and session management flaws

7. Broken Authentication and Session Management

- Vulnerabilities
  - Flaws in main authentication mechanism
  - Password management
  - Session Timeout
- Threats
  - Proxy caches (discussed in this presentation)

7. Broken Authentication and Session Management

- Verifying Security
  - Applications should properly authenticate users and protect their session credentials
  - Ineffective: Automated scanning tools
  - Effective: Combination of code reviews and testing
- Protection
  - Maintain secure communications and credential storage
  - Use single authentication mechanism where applicable
  - Create a new session upon authentication
  - Ensure the logout link destroys all pertinent data
  - Do not expose credentials in URL or logs
  - Update: Test against aggressive proxy scenarios

7. Broken Authentication and Session Management

- Example OWASP References

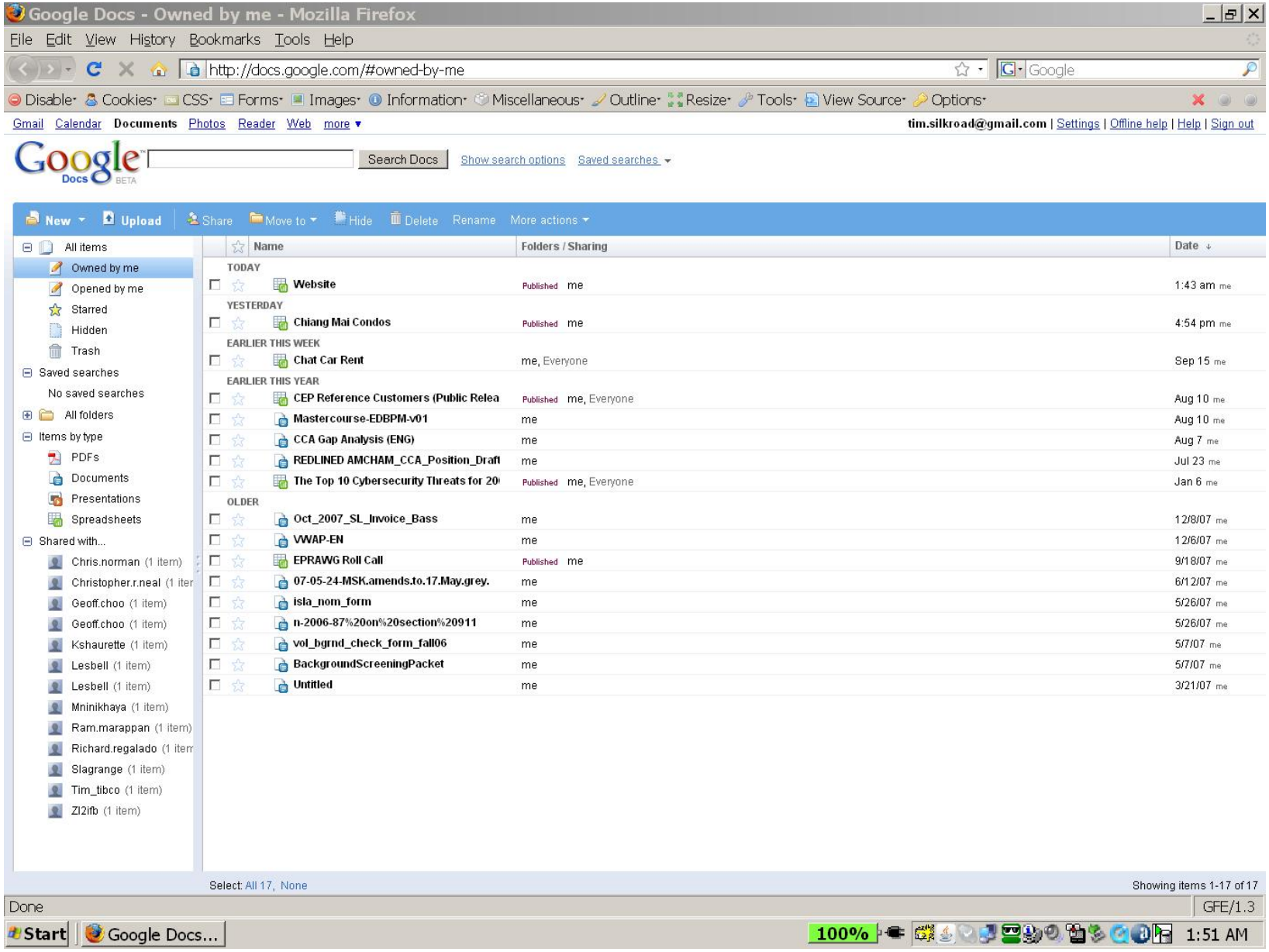
1. http://www.owasp.org/index.php/Guide_to_Authentication
2. http://www.owasp.org/index.php/Reviewing_Code_for_Authentication
3. http://www.owasp.org/index.php/Testing_for_authentication

OWASP has so many web application security tools, papers and guides, all FREE for you to use!

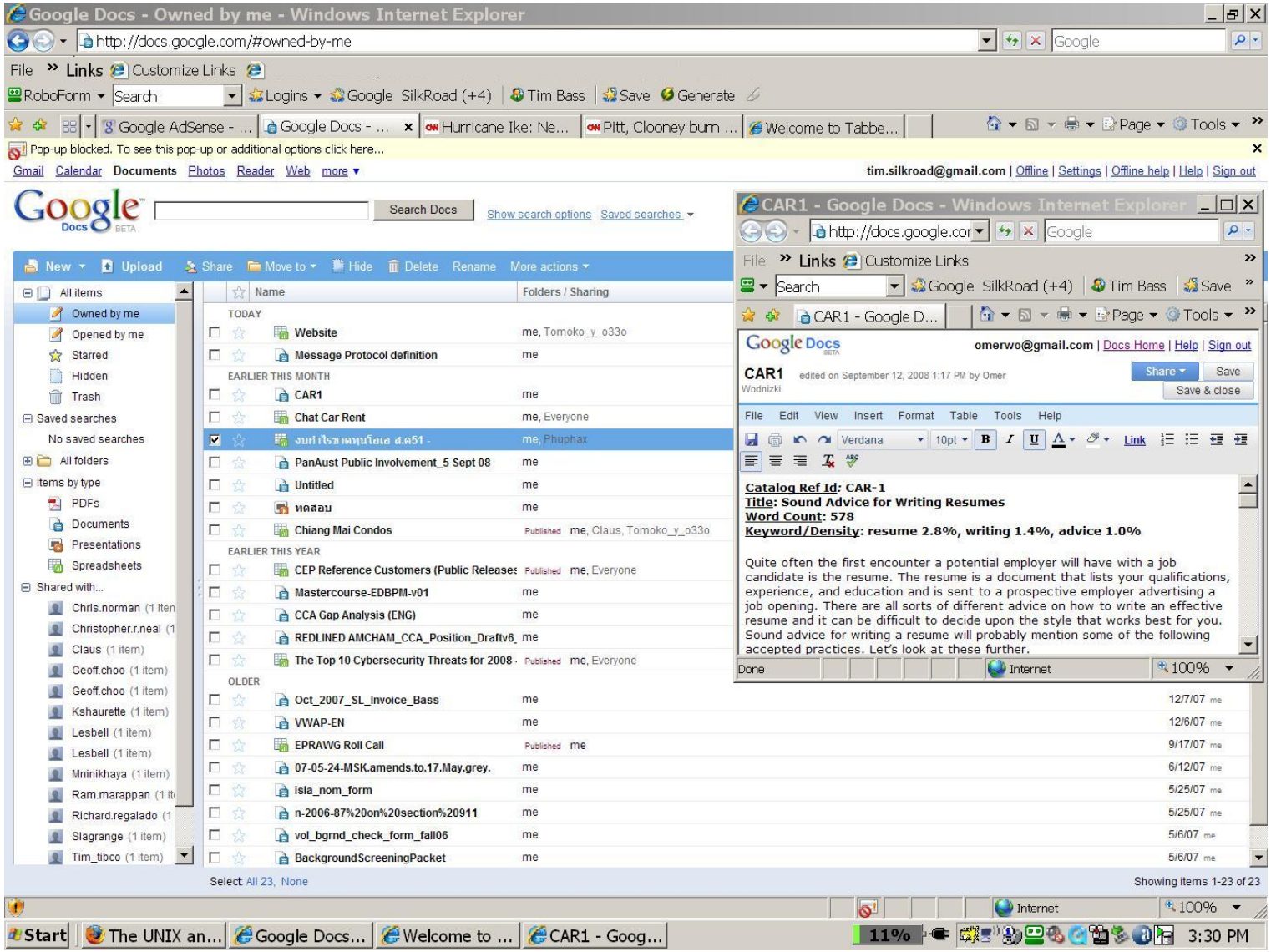
Our Agenda

- First, A Brief Review of the OWASP Top 10
  - 7. Broken Authentication and Session Management
- Second, A Funny Thing Happened in GoogleDocs
- Third, Proxy Caches are a Serious Threat
  - Poorly written session management code is the vulnerability
  - Simple testing scenario(s)
  - ... and a warning ....

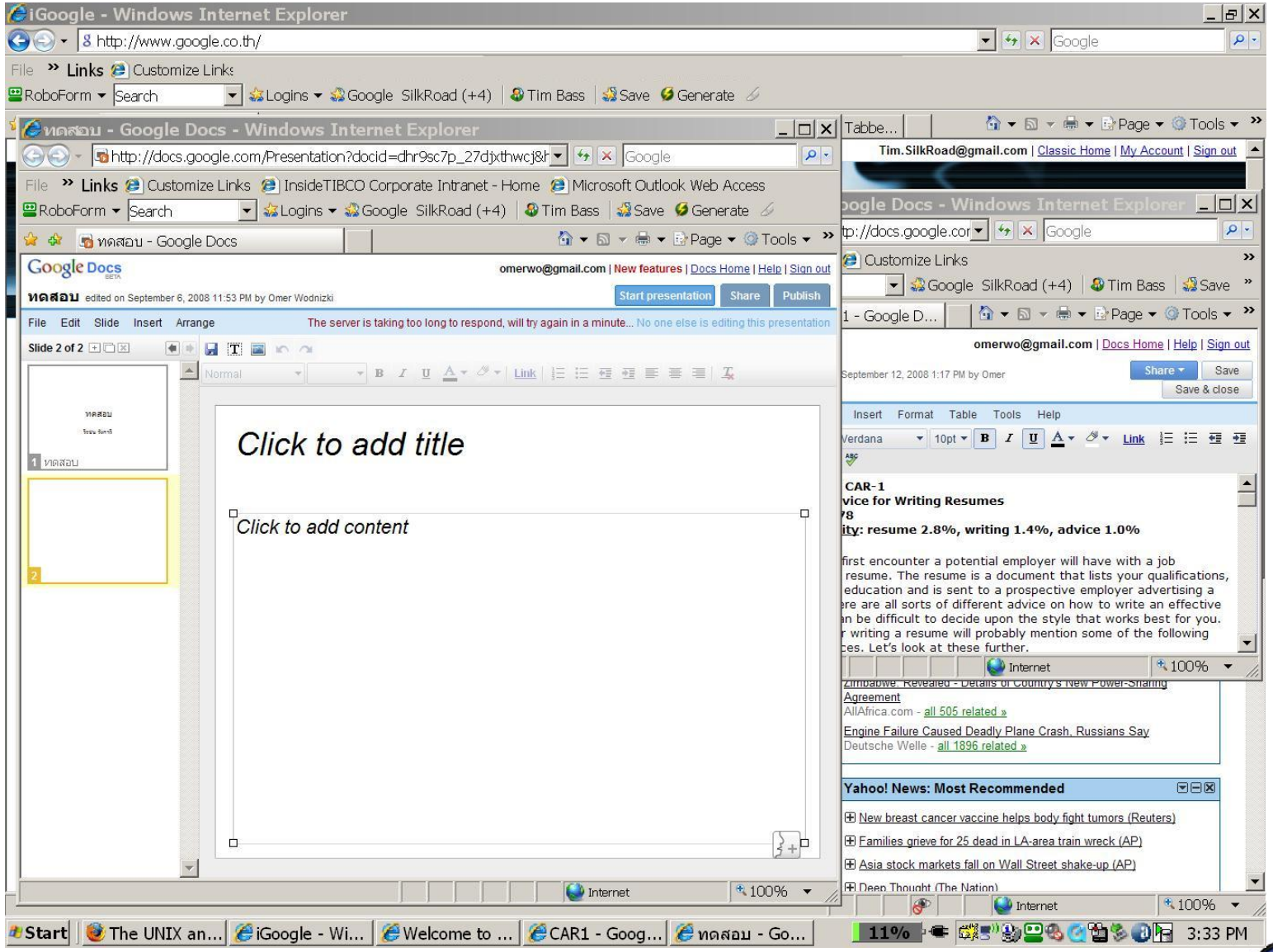
GoogleDocs Account Before.....



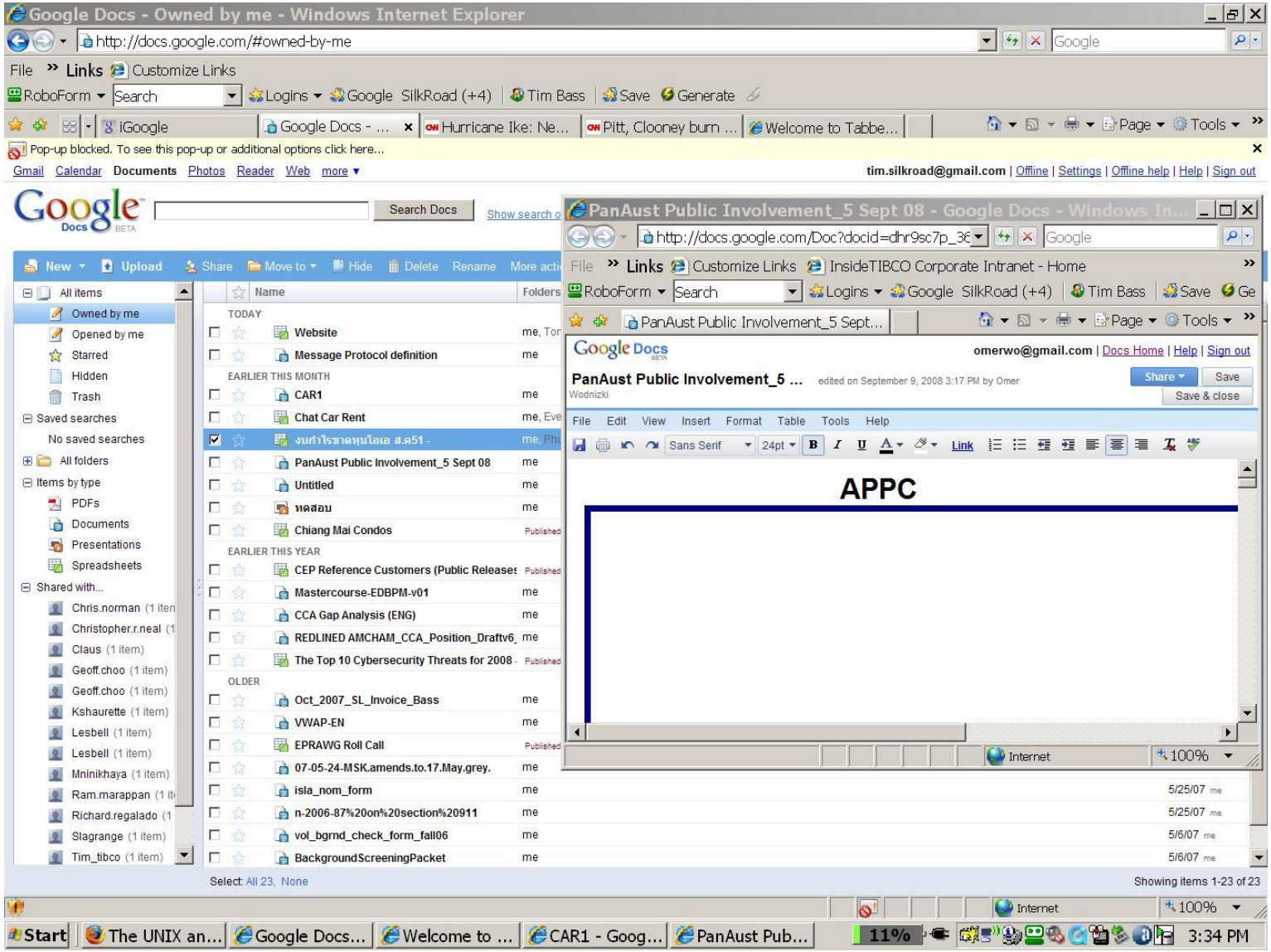
A Typical Day in GoogleDocs .....



GoogleDocs Account After .....



GoogleDocs Account After .....



Mr. Wodnizki says ....

Google Docs Security Problem Inbox | x

Tim Bass to omerwo — show details Sep 15 — Reply

Hello Mr. Wodnizki,

Today I noticed your documents in my Google Docs account !!

There seems to be a breach of security in the system. Do you see anything unusual in your account?

Yours sincerely, Tim

Reply Forward

Omer Wodnizki to me — show details Sep 15 — Reply

Dear Mr. Bass,

It's weird that you're saying that I saw documents of someone else today...

Thanks for letting me know...

Omer.

- Show quoted text -

Reply Forward

Mr. Wodnizki says .... *"I deleted all ...."*

Tim Bass to Omer — show details Sep 15 — Reply

Hi,

I would like to see if we can narrow the problem. I am a security guy. www.linkedin.com/in/timbass

Can you take a screen shot of the documents that are strange and send to me? Also, are you in Thailand? I am. I saw some of your docs in Thai language. Is that right?

Yours sincerely, Tim

www.thecepbog.com
blog.isc2.com
www.unix.com
- Show quoted text -
--
Sent from Gmail for mobile | mobile.google.com

Reply Forward

Omer Wodnizki to me — show details Sep 15 — Reply

I am in Thailand, although I don't use Thai docs... I deleted all the unfamiliar docs
- Show quoted text -

Reply Forward

Google teamwork ...

[#336216285] Google Docs help group Inbox | x

Google Help to Tim.SilkRoad show details Sep 16 Reply

Hi Tim Bass,

Thank you for posting on the Google Docs help group. In reference to the post below, could you provide us the following information:

http://groups.google.com/group/Something-in-Writely-is-Broken/browse_thread/thread/b4a125af95b60c0a#

-Were you able to access your own documents immediately after you were able to access this other user's documents, or did you have to log back in again to see your documents?

We look forward to hearing from you.

Cheers,
Marie
The Google Team

Reply Forward Invite Google Help to chat

Tim Bass to Google show details Sep 16 Reply

Hi Marie,

I could see my documents and the other users documents at the same time. The documents belonging to other people were combined with mine.

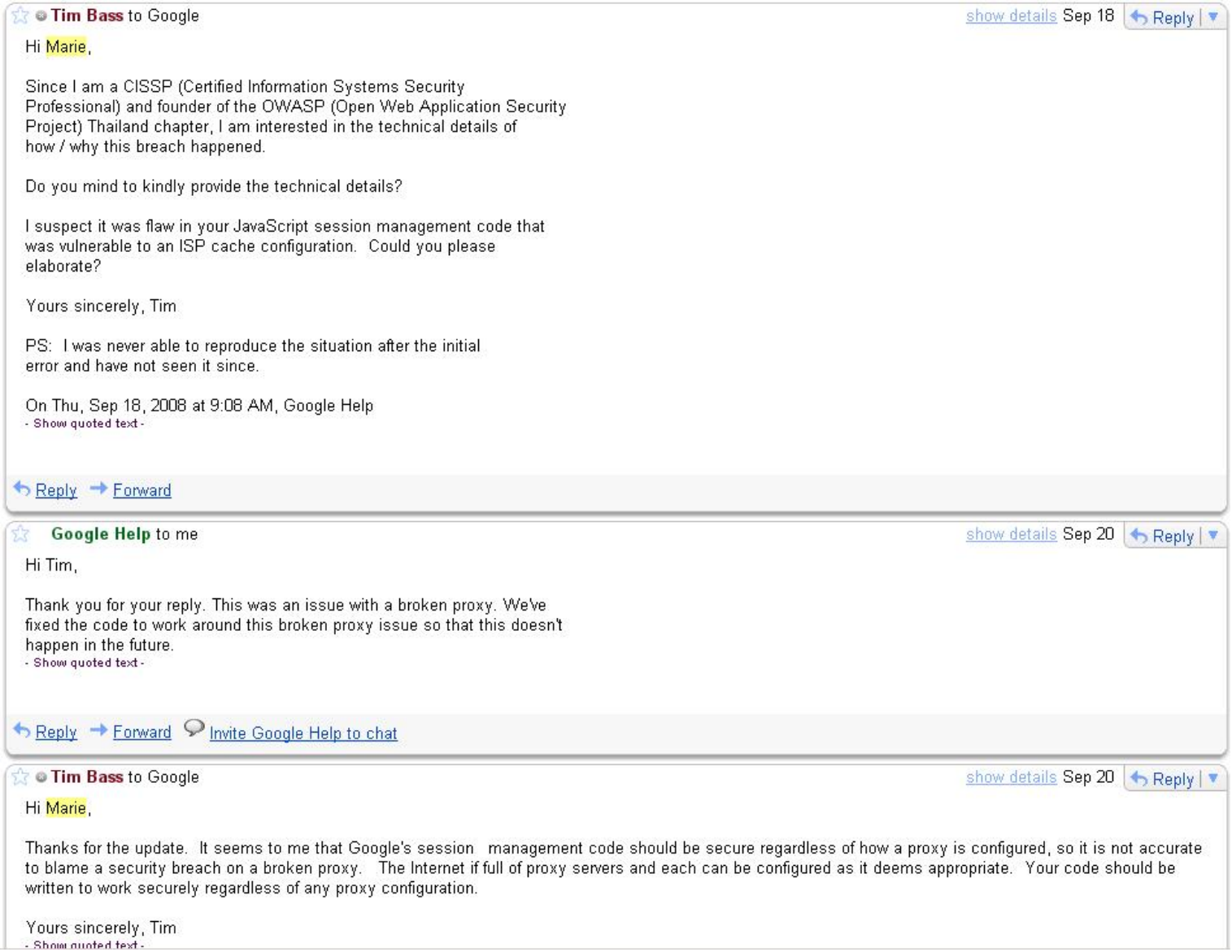
I only saw the documents belonged to other users when I clicked on the document and the other persons account information appeared.

Hope this helps.

Tim
- Show quoted text -

Reply Forward

Google says .... *“We’ve fixed the code.....”*



Tim Bass to Google — show details Sep 18 — Reply

Hi Marie,

Since I am a CISSP (Certified Information Systems Security Professional) and founder of the OWASP (Open Web Application Security Project) Thailand chapter, I am interested in the technical details of how / why this breach happened.

Do you mind to kindly provide the technical details?

I suspect it was flaw in your JavaScript session management code that was vulnerable to an ISP cache configuration. Could you please elaborate?

Yours sincerely, Tim

PS: I was never able to reproduce the situation after the initial error and have not seen it since.

On Thu, Sep 18, 2008 at 9:08 AM, Google Help
- Show quoted text -

Reply Forward

Google Help to me — show details Sep 20 — Reply

Hi Tim,

Thank you for your reply. This was an issue with a broken proxy. We've fixed the code to work around this broken proxy issue so that this doesn't happen in the future.
- Show quoted text -

Reply Forward Invite Google Help to chat

Tim Bass to Google — show details Sep 20 — Reply

Hi Marie,

Thanks for the update. It seems to me that Google's session management code should be secure regardless of how a proxy is configured, so it is not accurate to blame a security breach on a broken proxy. The Internet if full of proxy servers and each can be configured as it deems appropriate. Your code should be written to work securely regardless of any proxy configuration.

Yours sincerely, Tim
- Show quoted text -

Our Agenda

- First, A Brief Review of the OWASP Top 10
  - 7. Broken Authentication and Session Management
- Second, A Funny Thing Happened in GoogleDocs
- Third, Proxy Caches are a Serious Threat
  - Poorly written session management code is the (flaw) vulnerability
  - Simple testing scenario(s)
  - ... and a warning ....

Proxy Caches are a Serious Everyday Threat

- Proxy caches, combined with poorly written session management code, can easily lead to serious security flaws.
- Web application developers have no control over proxy caches in the Internet.
- Developers do have control of the code they write and their admin teams have configuration control of their web servers.
- Developers must assume the worst case Internet scenario with aggressive Internet cache management policies.

Caches are the Threat. Bad Code is the Flaw.

Developers Must Assume a Full Time Proxy Cache Threat Exists

- Web developers cannot know whether their content is consumed directly or via a (transparent) proxy cache.
- Developers cannot assume that the HTTP responses will be delivered to the intended client.
- Moreover, developers cannot be sure that the target browser even receives the intended content.

For example, a session ID issued to a client gets used while it is valid or until abandoned and expired. If it is served and delivered in response to an unencrypted HTTP GET request, there's no guarantee it will be consumed by the intended web browser.

Developers Must Assume a Full Time Proxy Cache Threat Exists

- For example, this fact-of-life on the Internet can result in multiple web clients being sent the same Set-Cookie HTTP headers.
- Caching proxy servers should obtain a fresh cookie for the each new client request.

Ideally, proxy caches should not cache session management cookies and distribute cached cookies to multiple clients – but they can and do.

SSL is Critical, But Not Foolproof

- SSL must be used on ALL web transactions that require confidentiality and privacy.
- However, SSL is not foolproof.

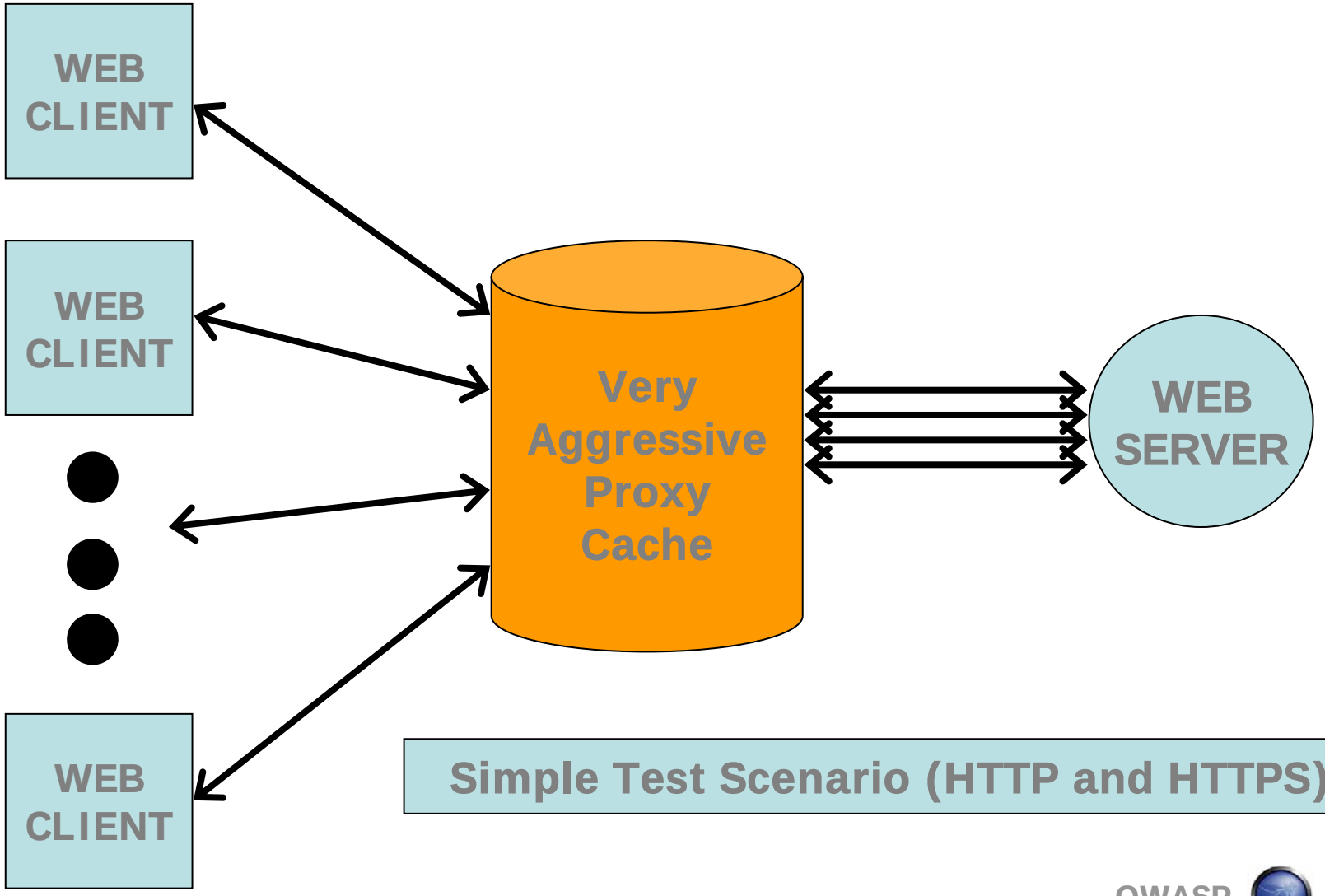
For example, web developers may not correctly set the "Encrypted Sessions Only" cookie property. Incorrectly configured “secure” servers will send HTTPS cookies in the open, unencrypted.

SSL is Critical, But Not Foolproof

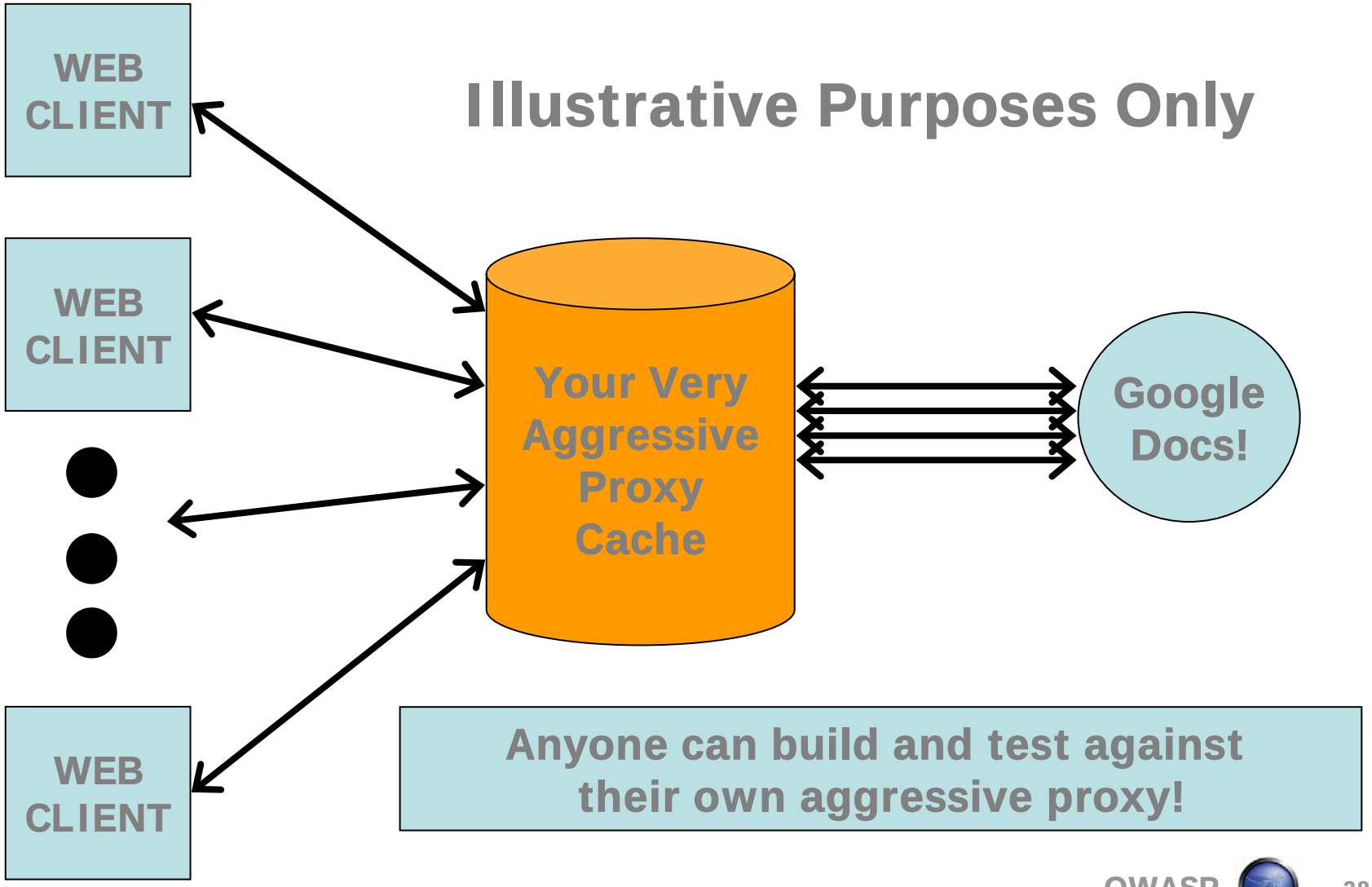
- SSL must be used on ALL web transactions that require confidentiality and privacy.
- However, SSL is not foolproof.

For example, web developers may not correctly set the "Encrypted Sessions Only" cookie property. Incorrectly configured “secure” servers will send HTTPS cookies in the open, unencrypted.

Testing Scenario- Single Server, Single Cache



Testing Scenario- Test Third Party Web Apps



Some Takeaways of this Presentation

- Criminals can easily configure aggressive caches and look for vulnerabilities in web application session management code, including unencrypted cookies.
- Criminals can then seek to attack from ISPs that have aggressive proxy cache management policies.

This means that all (risk critical) web applications should be completely tested against an aggressive proxy cache to insure that criminals cannot exploit a basic configuration in the Internet.

- This is huge.

References

- Blog Posts

A New Security Breach in Google Docs Revealed

http://www.thecepblog.com/2008/09/15/a-new-security-breach-in-google-docs-revealed/

Proxy Caches are a Challenging Threat to Internet Security

http://www.thecepblog.com/2008/10/05/proxy-caches-are-a-challenging-threat-to-internet-security/

Automated HTTPS Cookie Hijacking

http://fscked.org/blog/fully-automated-active-https-cookie-hijacking

Thank You!


OWASP AppSec Asia 2008 @Taiwan
OWASP 아시아 연차 총회
OWASP 亚洲年度会议
OWASPアジア年会
OWASP 亞洲年會
OWASP एशिया वार्षिक सम्मेलन
Hội thảo thường niên
OWASP Châu Á
การประชุมประจำปีเอเชีย OWASP
Persidangan Tahunan OWASP Asia
OWASP AppSec Asia Conference
OWASP konferensi tahunan Asia