ClubHACK Mag

1st Indian "HACKING" Magazine

Issue 4 | May 2010
www.chmag.in

TechGyan
CSRF

Mom's Guide
Privacy on Facebook

LegalGyan
Real life case

ToolGyan
Imposter

Last month, Facebook introduced yet another changes in their privacy setting, this time they allowed applications and websites to have an access to general information of the user. We have been seeing a series of changes in facebook pipeline since long and whole timeline was also published by Electronic Frontier Foundation which can be seen at http://www.eff.org/deeplinks/2010/04/facebook-timeline/.In this issue we will be covering this important policy and see how a normal user can be safe. What are the changes expected by users in their privacy setting to keep the data safe. Along with this, we have our regular sections under which we are looking at a very new tool released in ClubHack2009 by our friend Lavakumar called "Imposter". Poster of the month is again on facebook privacy which was designed by me. Yes! I'm the graphics guy behind the magazine. Hope you like my creations :P

Deepranjan S More

ClubHACKMag

Issue 4, May 2010.

Team CHmag

Rohit Srivastwa
rohit@clubhack.com

Aarja Bhattacharyya
aarja@chmag.in

Abhijeet R Patil
abhijeet@chmag.in

Abhishek Nagar
abhishek@chmag.in

Deepranjan S More
deepranjan@chmag.in

Pankit Thakkar
pankit@chmag.in

Varun V Hirve
varun@chmag.in

www.chmag.in
info@chmag.in

CONTENTS

What is CSRF?

A cross-site request forgery is an attack that involves forcing a victim to send an HTTP request to a target destination without their knowledge or intent in order to perform an action as the victim. The attack basically uses the session of an authenticated user and is also known as **Session riding**.

The CSRF name was given to them by Peter Watkins in a June 2001 posting to the Bugtraq mailing list.

CSRF outlined

First of all to understand what are forged cross-site requests (CSRF)? Cross-site request forgery is a Compromised end - users currently log in at the Web application to perform the operation of non-intention of attack methods. As long as the attacker with some social engineering tricks, such as by e-mail or chat software to send the link, the attacker will be able to force a Web application user to perform the selection operation of the attacker.

For example, if users log in internet banking go to view their deposits, he did not quit the network banking system go enjoy their own forum go irrigation, if the attacker at the forum carefully construct a malicious link and to induce the user clicks on the link, then the user at the network bank account funds may be transferred to the attacker specified account.
When CSRF attacks against ordinary users, the data will be the end-user and operational directives pose a serious threat; when attacks by the end user with an administrator account when, CSRF attacks will threaten the entire Web application.

What kinds of applications are vulnerable?

Pretty much all of them. Any application using HTTP GET, HTTP POST, HTTP PUT etc. without ensuring that the request came willfully from the authenticated user and not by some trick are vulnerable.

Web applications with bad architecture

Any web application which allows some change purely based on GET requests is violating the RFC. According to the **HTTP RFC #2616** section 9.1.1 Safe Methods; GET should not have the significance of taking an action. They shouldn't be allowing GET requests to take actions.

Automating POST requests

Using AJAX techniques this can be achieved without any active user intervention and with complete stealth. Either using iframes with embedded forms or the favourite of Web2.0 XHR also known as AJAX it is trivial to automate POST requests without any user intervention whatsoever.

Any web page which is allowed to run javascript is capable of doing this. Using a very strict ruleset of Mozilla Firefox extension noscript might help you a bit. But it increases the pain of browsing immensely.

Mozilla

```
<script>
  var post_data = 'name=value';
  var xmlhttp=new
XMLHttpRequest();
  xmlhttp.open("POST",
'http://url/path/file.ext',
true);
  xmlhttp.onreadystatechange =
function () {
  if (xmlhttp.readyState == 4)
  {
  alert(xmlhttp.responseText);
  }
  };
  xmlhttp.send(post_data);
      </script>
```

Anatomy of a CSRF attack

Imagine you are logged into your web mail in one tab while browsing other websites in other tabs. Your web mail has a logout link which is simply a HTTP GET request. This is what the request look like.

```
GET http://example.com/logout.php
HTTP/1.1
```

Now a malicious person adds an image tag to a website you regularly visit. Rather than pointing to the URL of the actual image this is what he/she does.

```
<img
src="http://example.com/logout.php"
/>
```

When your browser encounters that part of the HTML it will make a HTTP GET request on your behalf and try to download the image from the URL mentioned. As soon as the HTTP GET request is made your web mail will log you out.

So what is the big deal?

What if rather than logout it pointed to adding a mail filter forwarding all mails to another account?

```
<img
src="http://example.com/addfilter.p
hp?src=*&dest=evil@evilperson.com
/>
```

Or what if the URL pointed to adding/modifying your DSL router

http://www.gnucitizen.org/blog/bt-home-flub-pwnin-the-bt-home-hub/

Maybe initiating money transfer using your bank website.

Example Exploit :-

The simplest example exploit uses an image. To understand the exploit, it is first necessary to understand how a browser requests images. Consider a very simple page:

```
<HTML>
<P>HERE IS ANY SAMPLE IMAGE:
<IMG SRC="HTTP://EXAMPLE.ORG/EXAMPLE.PNG"
/></P>
</HTML>
```

When a browser requests this page, it cannot know that the page has an image. The browser only realizes this once it parses the HTML within the response. It is at this point that the browser requests the image, and it uses a standard **GET** request to do so. This is the important characteristic. It is impossible for the target site to distinguish between a request for an image and a request for any other resource.

When requesting an image, some browsers alter the value of the **Accept** header to give a higher priority to image types. Resist the urge to rely upon this behavior for protection - I show you a more reliable approach at the end of this article.

Now, imagine that the image in a page is the following:

```
<img
src="http://example.org/buy.php?sym
bol=SCOX&shares=1000" />
```

Every user that visits this page sends a request to example.org just as if the user clicked a link to the same URL. Because the sample application uses **$_REQUEST** instead of the more specific **$_POST**, it cannot distinguish between data sent in the URL from data provided in the proper HTML form.

This is an intentional mistake that I want to highlight.
Using **$_REQUEST** unnecessarily increases your risk. In addition, if you perform an action (such as buying stocks) as a result of a **GET** request, you are violating the HTTP specification.

In particular, the convention has been established that the GET and HEAD methods SHOULD NOT have the significance of taking an action other than retrieval. These methods ought to be considered "safe". This allows user agents to represent other methods, such as POST, PUT and DELETE, in a special way, so that the user is made aware of the fact that a possibly unsafe action is being requested.

POST requests can also be forged, so do not consider a strict use of **$_POST** to be sufficient protection.

Example of CSRF

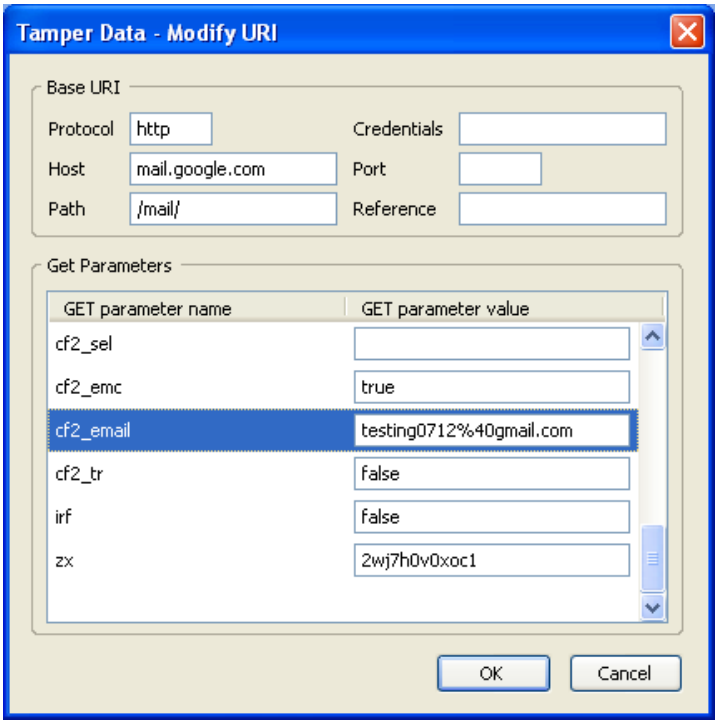
CSRF is a malicious attack that transmits unauthorized commands to a website from a trusted user. Let's take a simple example to understand this:

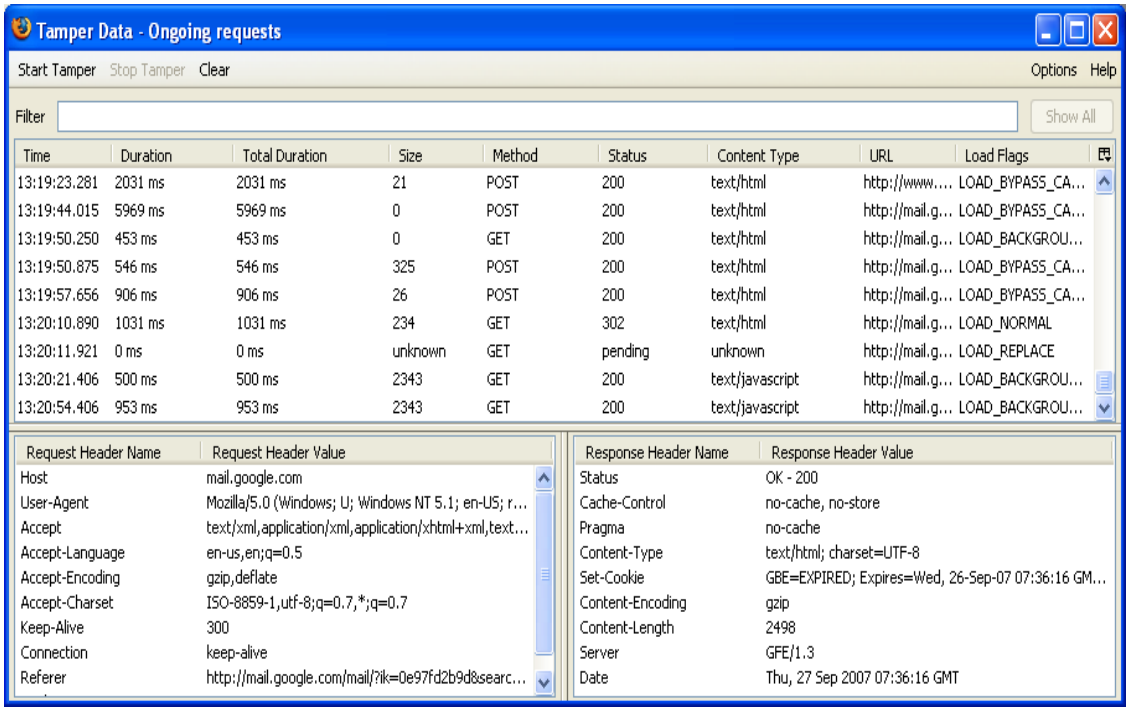
1. Sheetal is checking her email on a very safe computer (totally patched, fire-walled, with the latest updated antivirus etc etc) using a very safe browser.

2. Because she is logged in to Gmail, the Gmail authentication cookie is present on her machine.
3. She receives an email from Deepali containing a really funny joke. The email contains a link to a site which promises her lots more funny stuff. She clicks on the link and is very happy with the site that opens up.
4. What she does not realize is that this joke site has forged a POST request to the Gmail "Create Filter" wizard. This creates a filter that forwards a copy of all emails coming into Sheetal's account to Deepali!
5. Gmail accepts the request to create the filter because the genuine Gmail account holder (Sheetal) is authenticated and logged in at the moment and her session cookie is passed along with the forged request.
6. This new Gmail filter will keep stealing Sheetal's emails till she manually checks her filters and realizes that there is an unauthorized one.

I carried out a small experiment wherein I requested for a filter to forward all emails to a particular account. On analyzing the data transfer between my browser and Gmail (using Tamper Data extension for Firefox) I found that the relevant GET parameter is cf2_email.

See the following for more information on





this

Safeguard against CSRF

There are a few steps you can take to mitigate the risk of CSRF attacks. Minor steps include using **POST** rather than **GET** in HTML forms that perform actions, using **$_POST** instead of **$_REQUEST**, and requiring verification for critical actions (convenience typically increases risk, and it's up to you to decide the appropriate balance).

The most important thing you can do is to try to force the use of your own forms. If a user sends a request that looks like it is the result of a form submission, doesn't it make sense to be a little suspicious if the user has not recently requested the form?

Consider the following replacement for the HTML form in the sample application:

```
<?php

$token = md5(uniqid(rand(),
TRUE));
$_SESSION['token'] = $token;
$ SESSION['token time'] =
time();
 ?>
<form action="buy.php"
method="post">
<input type="hidden"
name="token"
   value="<?php echo $token; ?>"
   /><p>
Symbol: <input type="text"
name="symbol" /><br />
Shares: <input type="text"
name="shares" /><br />
<input type="submit" value="Buy"
/>
</p>
</form>
```

Because this form does not represent an entire script, we do not include the call to **session_start ().** Readers can safely assume that this required step takes place prior to the form.

With this simple modification, a CSRF attack must include a valid token (anti-CSRF token) in order to perfectly mimic the form submission. Because you store the user's token in the session, it is also necessary that the attacker uses the token unique to the victim. This effectively limits any attack to a single user, and it requires the attacker to obtain a valid token for another user (obtaining your own token is useless when it comes to forging requests from someone else).

This script should be initialized just like any other session variable:

```
<?php
if (!isset($_SESSION['token']))
{
 $_SESSION['token']
=md5(uniqid(rand(), TRUE));
}
?>
```

The script can be checked with a simple conditional statement:

```
<?php
if ($_POST['token'] == $_SESSION['token'])
{
 /* Valid Token */
}
?>
```

The validity of the script can also be limited

to a small window of time, such as five minutes:

```
<?php

$token_age = time() -
$_SESSION['token_time'];
 if ($token_age <= 300)
{
  /* Less than five minutes has
passed*/
}
?>
```

Challenge-Response

Challenge-Response is another defense option for CSRF. The following are some examples of challenge-response options.

Prevention Measures That Do NOT Work

- Using secret cookies.
- Only Accepting POST request.
- Checking Refer Header.
- Multi-Step Transactions.
- URL rewriting.

Akash Mahajan
http://akashm.com

Mom's GUIDE

Privacy on Facebook

According to statistics, Facebook has more than 400 million active users worldwide. If we consider Facebook to be a country, then Facebook would make the third most populous country in the world after China and India!!!

Country	Population
1. China	1,337,160,000
2. India	1,180,040,000
3. Facebook	400,000,000
4. United States	309,140,000
5. Indonesia	234,181,400
6. Brazil	192,835,000
7. Pakistan	169,333,500
8. Bangladesh	162,221,000
9. Nigeria	154,729,000
10. Russia	141,927,297

More than 25 billion pieces of content like Web Links, photo albums, videos, notes, etc are shared each month. Almost every user use applications like games and quizzes on Facebook.

But unfortunately, most of them don't know the implications of entering personal information. Almost everyone can see almost everything that the user puts up on Facebook. A lot of people have got into trouble for things they have posted on Facebook.

This Mom's Guide will show what you can do to be safe and maintain your privacy on Facebook.

1. Use/Organize Friend Lists

Many users don't know about this feature provided by Facebook.

Friend Lists basically provides organized grouping of your friends based on your personal preferences. For example, you can create a Friend List for your friends that meet for weekly book club meetings. This allows you to quickly view friends by type and send messages to your lists.

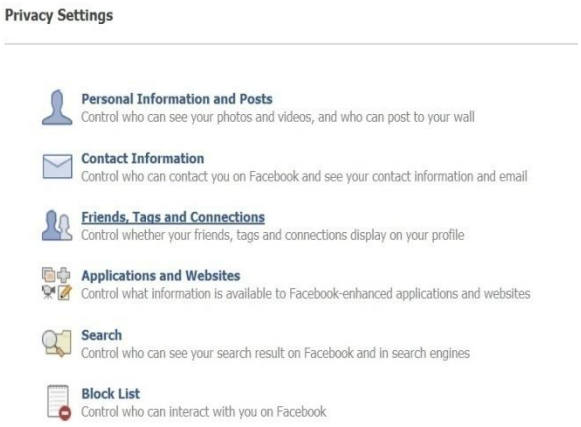
Important things to remember :-

- You can add each friend to more than one friend list.
- Friend Lists can have specific privacy policies applied to them.

Steps to create and customize Friend Lists:-

1. Click the "**Create New List**" that appears beneath the filters on the left side of your home page or your Friends page. Or, click the "**Create New List**" button from the "**All Friends**" tab of the Friends page.
2. Type the title of your list and hit enter.
3. Add friends to the list by typing their names into the "Add to List" field or selecting them from the list.
4. Select "Create List" to store your changes.

You can add privacy setting by going to Privacy Settings.



Under **Privacy Settings** you can control privacy of your Friend Lists by editing Personal Information, Contacts, Friends, etc.

2. Privacy Settings - Profile Privacy

Here you can control who can see which information. Your privacy setting should be simple and easy to understand. There are three basic levels of privacy: **Friends, Friends of Friends** and **Everyone**.

Adjust privacy of what you share.

Options available under Privacy Settings:-

1. **Personal Information and Posts:-** This covers your Bio, Birthday, Political Views, Posts and comments on them, Photos etc. According to your preference you can make this information available to Friends and Friends of Friends. It is recommended to make information like your Photos and videos available to Friends only.
2. **Contact Information:-** This covers your contact details like emails address and phone number. It is recommended to make this

information available to Friends only.

3. **Friends, Tags and Connections:-** This controls the information people can see on you profile like your Pages, list of friends, your Relationship Status, Interests, etc. These settings let you control who sees this information.
   For example, you may want to keep your relationship status a secret or let only Friends know and let Friends of Friends see your Political views and Interests.
4. **Applications and Websites:-** Lets you control what information is available to applications and websites.
5. **Search:-** Lets you control who can search on Facebook and search engines. It is recommended to uncheck "**Allow**" option of Public Search Results. The Facebook Search Results can be set according to your preference.
6. **Block List:-** Lets you control who can interact with you on Facebook. You can block any profile from your list of friends.

3. Applications Settings

Facebook allows you to add and use third party applications. These applications use your information to process results. Though Facebook has published terms of service to inform developers what is and is not acceptable behavior, but there is always the danger that people abuse the ability.

Therefore you must carefully add these applications. The only thing that can be done is to authorize only those applications you require and trust.

Go **to Account → Application Settings**.

Change the drop down from **Recently Used** to **Authorized**. Now you can see all the applications you have authorized to get access to all your profile information.

You can also check the applications Allowed to Post and Granted Additional Permissions. You can remove all the unwanted applications.

4. Disable Facebook's "Instant Personalization"

Recently, Facebook added new features that let some websites access your personal information. These sites personalize your experience using information you have made visible to everyone as well as your publicly available information. This includes your Name, Profile Picture, Gender, Current City, Networks, Friend List, and Pages

As of now Facebook has allowed only three site to use this "Instant Personalization: feature, they are – Docs.com, Pandora.com and Yepl.com.

This feature actually allows websites to display "**Like**" button. If you click this "**Like**" button, it will show all your friends on Facebook that you liked something on this particular website. This can be done even without logging into Facebook. All it requires is to login sometime before you

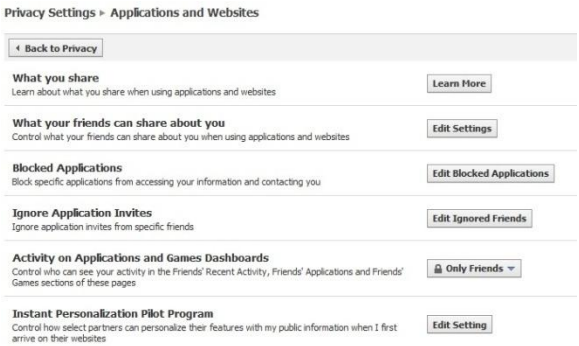
visit the site. When you arrive on these sites, you'll see a notification from Facebook at the top of the page. You can easily opt-out of experiencing this on these sites by clicking "**No Thanks**" on the blue Facebook notification on the top of partner sites.

This feature is by default on. So if don't want these website to have access to your information, you will have to specifically turn this option off.

Turn off Instant Personalization:-

Go **to Accounts → Privacy Settings → Applications and Websites**.

Click "**Edit Setting**" at "**Instant Personalization**" at the bottom.



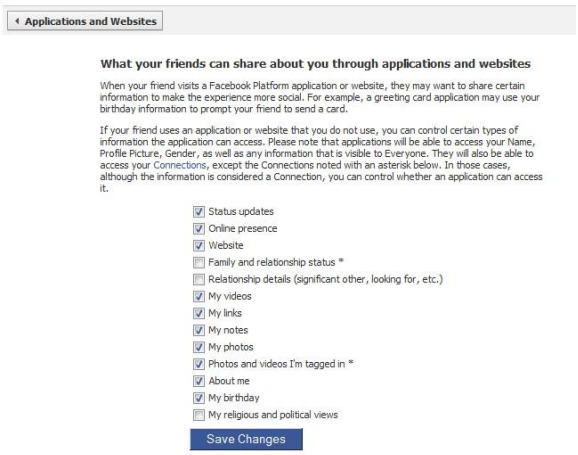
At the bottom of the page, a lone check box is provided to enable or disable the feature. By default it's on. Uncheck it to turn off the Instant Personalization feature.

It's done! Now Facebook will not share your personal information with websites for Instant Personalization until your turn it on again.

But this won't prevent your friends from sharing and using your information. Your friends may still share public Facebook information about you to personalize their experience on these partner sites unless you block the application.

So how do you stop that? Stop it by blocking those applications!

Under Applications and Websites, Edit settings for "What your friends can share about you".



If you don't want any information to be shared with specific applications, either by you or your friends, you have to uncheck to specifically block those applications by them.

Lastly, don't click the "**like**" button at any website you visit. This will prevent your information from being shared.

5. Delete/Deactivate Facebook Account

If you are quitting Facebook, then you have two options – **Deactivate** and **Delete**.

Out of the two options, Deactivation is simple. It can be done from **Settings** page. But de-activation will retain all your information on Facebook, including your

photos, videos, friends, etc. But this information is made inaccessible to other Facebook users. However, if you want to reactivate your account, you just have to login on to Facebook again.

If you don't think you will use Facebook again and would like to delete your account permanently, you will have to submit a deletion request here.

Note, unlike deactivation option, once account deleted, there is no option of recovery.

Incidents like Facebook account getting hacked or leakage of information keep happening regularly. So it is recommended to delete your account rather than deactivating it. It is better to be safe than sorry!

Conclusion

These are just few ways of maintaining your privacy and keeping your account safe.

There are many more small things to keep in mind but the above settings are most important.

So follow all the steps and be safe!

Happy Social Networking

Abhijeet Patil
abhijeet@chmag.in

Some real life scenarios

Continued from last issue, we are analyzing some of the common cyber related crimes from a legal perspective.

6. Source Code Theft

Computer source code is the most important asset of software companies. Simply put, source code is the programming instructions that are compiled into the executable files that are sold by software development companies.

As is expected, most source code thefts take place in software companies. Some cases are also reported in banks, manufacturing companies and other organizations that get original software developed for their use.

Scenario 1:

The suspect (usually an employee of the victim) steals the source code and sells it to a business rival of the victim.

Modus Operandi: If the suspect is an employee of the victim, he would usually have direct or indirect access to the source code. He would steal a copy of the source code and hide it using a virtual or physical storage device. If the suspect is not an employee of the victim, he would hack into the victim's servers to steal the source code. Or he would use social engineering to get unauthorised access to the code. He would then contact potential buyers to make the sale.

Usual motives: Illegal financial gain.

Applicable law

Before 27 October, 2009	After 27 October, 2009
Sections 43, 65 & 66 of the *Information Technology Act* and section 63 of *Copyright Act*	Sections 43, 65, 66 & 66B of the *Information Technology Act* and section 63 of *Copyright Act*

Scenario 2:

The suspect (usually an employee of the victim) steals the source code and uses it as a base to make and sell his own version of the software.

Modus Operandi: If the suspect is an employee of the victim, he would usually have direct or indirect access to the source code. He would steal a copy of the source code and hide it using a virtual or physical storage device. If the suspect is not an employee of the victim, he would hack into the victim's servers to steal the source code. Or he would use social engineering to get unauthorised access to the code.

He would then modify the source code (either himself or in association with other programmers) and launch his own software.

Usual motives: Illegal financial gain.

Applicable law

Before 27 October, 2009	After 27 October, 2009
Sections 43, 65 & 66 of the *Information Technology Act* and section 63 of *Copyright Act*	Sections 43, 65, 66 & 66B of the *Information Technology Act* and section 63 of *Copyright Act*

7. Theft of Confidential Information

Most business organizations store their sensitive information in computer systems. This information is targeted by rivals, criminals and sometimes disgruntled employees.

Scenario 1:

A business rival obtains the information (e.g. tender quotations, business plans etc) using hacking or social engineering. He then uses the information for the benefit of his own business (e.g. quoting lower rates for the tender).

Usual motives: Illegal financial gain.

Applicable law

Before 27 October, 2009	After 27 October, 2009
Sections 43 & 66 of the *Information Technology Act* and section 426 of *Indian Penal Code*	Sections 43, 66 & 66B of the *Information Technology Act* and section 426 of *Indian Penal Code*

Scenario 2:

A criminal obtains the information by hacking or social engineering and threatens to make the information public unless the victim pays him some money.

Usual motives: Illegal financial gain.

Applicable law

Before 27 October, 2009	After 27 October, 2009
Sections 43 & 66 of the *Information Technology Act* and section 384 of *Indian Penal Code*	Sections 43, 66 & 66B of the *Information Technology Act* and section 384 of *Indian Penal Code*

Scenario 3:

A disgruntled employee steals the information and mass mails it to the victim's rivals and also posts it to numerous websites and newsgroups.

Usual motives: Revenge.

Applicable law

Before 27 October, 2009	After 27 October, 2009
Sections 43 and 66 of the *Information Technology Act* and section 426 of *Indian Penal Code*	Sections 43, 66, 66B of the *Information Technology Act* and section 426 of *Indian Penal Code*

8. Software Piracy

Many people do not consider software piracy to be theft. They would never steal a rupee from someone but would not think twice before using pirated software. There is a common perception amongst normal computer users to not consider software as "property". This has led to software piracy becoming a flourishing business.

Scenario 1:

The software pirate sells the pirated software in physical media (usually CD ROMs) through a close network of dealers.

Modus Operandi: The suspect uses high speed CD duplication equipment to create multiple copies of the pirated software. This software is sold through a network of computer hardware and software vendors.

Usual motives: Illegal financial gain.

Applicable law

Before 27 October, 2009	After 27 October, 2009
Section 43 & 66 of the *Information Technology Act* and section 63 of *Copyright Act*	Section 43 & 66 of the *Information Technology Act* and section 63 of *Copyright Act*

Scenario 2:

The software pirate sells the pirated software through electronic downloads through websites, bulletin boards, newsgroups, spam etc.

Modus Operandi: The suspect registers a domain name using a fictitious name and then hosts his website using a service provider that is based in a country that does not have cyber laws. Such service providers

do not divulge client information to law enforcement officials of other countries.

Usual motives: Illegal financial gain.

Applicable law

Before 27 October, 2009	After 27 October, 2009
Section 43 & 66 of the *Information Technology Act* and section 63 of *Copyright Act*	Section 43 & 66 of the *Information Technology Act* and section 63 of *Copyright Act*

9. Music Piracy

Many people do not consider music piracy to be theft. They would never steal a rupee from someone but would not think twice before buying or using pirated music. There is a common perception amongst people that music is not "property". There is a huge business in music piracy. Thousands of unscrupulous businessmen sell pirated music at throw away prices.

Scenario 1:

The music pirate sells the pirated music in physical media (usually CD ROMs) through a close network of dealers.

Modus Operandi: The suspect uses high speed CD duplication equipment to create multiple copies of the pirated music. This music is sold through a network of dealers.

Usual motives: Illegal financial gain.

Applicable law

Before 27 October, 2009	After 27 October, 2009
Section 43 & 66 of the *Information Technology Act* and section 63 of *Copyright Act*	Section 43 & 66 of the *Information Technology Act* and section 63 of *Copyright Act*

Scenario 2:

The music pirate sells the pirated music through electronic downloads through websites, bulletin boards, newsgroups, spam emails etc.

Modus Operandi: The suspect registers a domain name using a fictitious name and then hosts his website using a service provider that is based in a country that does not have cyber laws. Such service providers do not divulge client information to law enforcement officials of other countries.

Usual motives: Illegal financial gain.

Applicable law

Before 27 October, 2009	After 27 October, 2009
Section 43 & 66 of the *Information Technology Act* and section 63 of *Copyright Act*	Section 43 & 66 of the *Information Technology Act* and section 63 of *Copyright Act*

10. Email Scams

Emails are fast emerging as one of the most common methods of communication in the modern world. As can be expected, criminals are also using emails extensively for their illicit activities.

In the first step, the suspect convinces the victim that the victim is going to get a lot of money (by way of winning a lottery or from a corrupt African bureaucrat who wants to transfer his ill gotten gains out of his home country). In order to convince the victim, the suspect sends emails (some having official looking documents as attachments).

Once the victim believes this story, the suspect asks for a small fee to cover legal expenses or courier charges. If the victim pays up the money, the suspect stops all contact.

Usual motive: Illegal financial gain.

Applicable law

Before 27 October, 2009	After 27 October, 2009
Section 420 of *Indian Penal Code*	Sections 66A and 66D of the *Information Technology Act* and section 420 of *Indian Penal Code*

Rohas Nagpal

rn@asainlaws.org

Tool GYAN

Imposter - Browser Phishing Framework

Imagine this scenario; you are sitting at the airport waiting for your boarding call. You turn on your laptop to kill time, scan for free Wi-Fi networks, connect to one of them and request for 'google.com'.

How much do you risk by doing this? 'Not much' you would say, right?

After all you have a fully patched OS, the latest antivirus, a strict personal firewall, you are logged in as a non-admin user and you would never click 'OK' on any pop-up. What could an attacker possible do?

Well, if the attacker is using Imposter then he could, in a matter of seconds do any or all of this:

1. Steal sensitive files from your system containing personal information.
2. Steal stored passwords of FaceBook, Twitter, MySpace and other accounts from the browser.
3. Place permanent backdoors in your browser for sites like Gmail, MySpace, and WordPress etc.
4. Steal your entire Gmail and MySpace inbox and Google Docs etc.

5. Steal your cookies and hijack your pre-authenticated sessions.
6. Poison your cookie store and perform session fixation attacks.
7. Steal sensitive files stored in your browser's cache.
8. Poison your browser's cache with malicious scripts.
9. List out every sleazy site you have visited by checking your LSO (Local Shared Objects), which usually never gets deleted.

All of this can be done so quickly and stealthily that you wouldn't even know what hit you.

Now that you know what Imposter can do, let's look at how it does this.

Browser Phishing:
Imposter performs all the mentioned attacks against the browser and it targets one of the most critical security features of all modern browsers, the SameOriginPolicy. Websites store a wide variety of data in the browser, these ranges from cookies, LocalSharedObjects, Cached Files, GoogleGears databases, GoogleGears LocalServer modules etc. All of this data is only accessible by the websites that store them, a concept which is called as SameOriginPolicy.
In scenarios where the victim is connected to an attacker-controlled Wi-Fi AccessPoint the attacker can serve content as any website. What this means is that now the attacker can pretend to be any website; from Google to FaceBook, as long as they are over HTTP. And so he can access the data that has been stored by those websites in the victim's browser.It's really that simple.

Imposter Internals:
Once the attacker sets up and configures Imposter as instructed in the detailed user guide - http://www.andlabs.org/tools/imposter/imposter.html, Imposter handles the rest.

Imposter has two main components that help it perform these attacks:

1) Imposter's DNS Server
2) Imposter's Web Server

Imposter's DNS server resolves every hostname to the attacker's IP address. So no matter what site the victim is requesting for, the request always comes to the Imposter's Web Server.

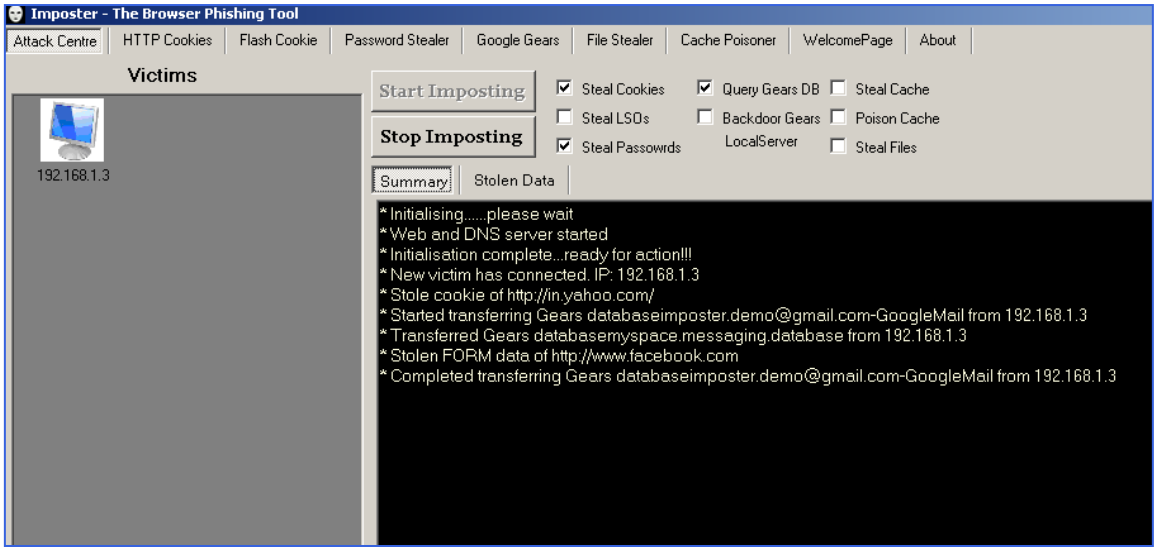
Imposter's Web Server sends a customizable harmless looking 'welcome page' to the victim. This page includes multiple iframes sourced to different domain names. Imposter's Web Server sends JavaScript payloads

Customizable 'welcome page' returned by Imposter to the victim. This page contains multiple iframes which load the payloads.

to these iframes to steal and poison offline data.

The data stolen by the JavaScript payloads is sent back to Imposter's Web Server. Imposter parses this data, processes it and stores it in a SQLite database. The entire process takes just a few seconds and does not trigger any alert or pop-up.

Permanent backdoor placement and Gmail Inbox stealing attacks are specific to users of Google Gears. Technical details on how these attacks work can be



Imposter stealing data from the victim

read from the whitepaper 'Google Gears for Attackers' - http://www.andlabs.org/whitepapers/GoogleGears_for_Attackers.pdf.

Imposter can also steal files from the victim's local file system if they are using Internet Explorer. It makes use of a special in-built sniffer for this attack. The technique behind this attack is described in detail in the whitepaper 'Flash+IE=PrisonBreak' - http://www.andlabs.org/whitepapers/F_IE_PrisonBreak.pdf.

Videos of most of the attacks are available at http://www.andlabs.org/videos.html

Imposter requires the .NET Framework to run and has currently only been tested on Windows. Imposter is a GUI tool which reduces the learning curve drastically; if you understand the theory behind the attacks you can start using Imposter in a matter of minutes. Imposter is also the only tool which can perform all the mentioned attacks out of the box.

Anti-Script Kiddies measure:

Given its ease of use, Imposter is almost too easy to be abused by Script Kiddies. To control this, Imposter does not come with any default preloaded configuration. Any person who understands the concepts behind the attacks should have no trouble configuring Imposter. However for a clueless individual it's like a gun without bullets, totally harmless.

Use in Penetration Testing:

Imposter should be a part of every Penetration tester's toolkit. I should know it well because I am a Penetration tester myself. There have been instances when clients have asked for a Penetration test on the laptops of their employees. This would almost always end up showing no results if you follow the traditional route of scan & exploit. This is because laptops rarely have services listening on them that can be accessed through inbound connections, thanks to the firewall.

What you should instead be targeting are the outbound connections. And perhaps the easiest way to get into any laptop would be through the browser. Imposter can help you check how secure the browser is against 'Browser Phishing' attacks. With browsers becoming the most prominent desktop application it is imperative that every possible attack vector, against them is tested for.

Future Plans:
I am currently working on a new module for Imposter and there are a few other related projects that are currently under active research. I cannot provide specific details about them at this point of time but do expect to hear more frequently about 'browser phishing' attacks in the future. A mono-compatible Linux variant is also in the pipeline.

If you are still reading this article then I am pretty sure you would be curious to check out Imposter. Take it for a spin and have fun with it, download link is available at http://andlabs.org/tools.html#imposter. If you have a specific feature request or some other feedback then please mail them to me and I will try my best to address them in the next version of Imposter. Until next time, browse safely!

Lavakumar Kuppan
@lavakumark

Are you patched?

We all have heard this in every advisory "patch your OS & software regularly"

In this issue of command line gyan, we'll see how we can check the status of updates/patches on windows & different flavors of Linux

Windows

Starting from this issue, we'll see a lot of windows examples using Windows Management Instrumentation Command (WMIC).

WMIC is set of extensions to the Windows Driver Model that provides an OS interface through which instrumented components provide information & notification.

For checking the status of patches, we can use the Quick Fix Engineering (QFE) alias of WMIC.

In this example we'll try to find if MS08-067 is installed on the client which was affected by conficker virus

```
C:\> wmic qfe where
hotfixid="KB958644" list
full
```

This will show whether the patch is installed or not. So the idea is to use the command with the appropriate KB number of Microsoft patch

If you want to run the same command on the remote machine you can do the same by

```
C:\> wmic qfe where
hotfixid="KB958644" list
full /node:192.168.1.25
/user:administrator
/password:i_wont_tell;)
```

Very obviously you need to have appropriate privileges on the remote machine to fetch this information.

If you want to use the same command on a number of machines, try passing a file name in /node with an @ sign

```
C:\> wmic qfe where
hotfixid="KB958644" list
full /node:@ip_list.txt
/user:administrator
```

Here if you leave the password parameter, the command will prompt for the password.

Remember, this command also prints the date of patch installation, so you can keep an eye on your admin too, with this.

Linux

For a redhat based system we'll use yum to do this task.

```
# yum list-security
```

Will list all the available security updates

&

```
# yum update --security
```

Will install all the security related updates

But the question is installation time because you want to catch your admin, right? For that you'll have to run the rpm command and get the info

```
# rpm -q --qf
"%{NAME}\t%{VERSION}\t%{INSTA
LLTIME:date}\n" mysqld
```

This will produce result like

```
mysqld 5.0.45 Wed 17 Mar
2010 02:39:56 AM IST
```

If you want to generate the list of all products with such information, simply try

```
# rpm -qa --qf "%-
30{NAME} %-15{VERSION}
%{INSTALLTIME:date}\n"
```

Which will produce a nice list in easy to read format

```
tcpdump                       3.9.4           Tue 16 Mar 2010 06:23:14 PM IST
parted                        1.8.1           Tue 16 Mar 2010 06:23:17 PM IST
kernel                        2.6.18          Tue 16 Mar 2010 06:23:23 PM IST
psgml                         1.2.5           Tue 16 Mar 2010 06:23:40 PM IST
xorg-x11-fonts-75dpi          7.1             Tue 16 Mar 2010 06:24:01 PM IST
rhel-instnum                  1.0.9           Tue 16 Mar 2010 06:24:17 PM IST
GConf2                        2.14.0          Tue 16 Mar 2010 06:24:29 PM IST
dbus                          1.1.2           Tue 16 Mar 2010 06:24:34 PM IST
avahi                         0.6.16          Tue 16 Mar 2010 06:24:39 PM IST
wpa_supplicant                0.5.10          Tue 16 Mar 2010 06:24:46 PM IST
yum-rhn-plugin                0.5.3           Tue 16 Mar 2010 06:24:51 PM IST
joystick                      1.2.15          Tue 16 Mar 2010 06:25:21 PM IST
dejavu-lgc-fonts              2.10            Tue 16 Mar 2010 06:25:34 PM IST
usermode-gtk                  1.88            Tue 16 Mar 2010 06:25:40 PM IST
poppler                       0.5.4           Tue 16 Mar 2010 06:26:01 PM IST
```

The way UBUNTU & other debian based systems are getting popular, it would be unfair if we don't mention the debian trick

```
# apt-show-versions -u
```

Isn't it pretty easy, like ubuntu itself. Interestingly in **apt**, it doesn't show you the timestamp and to get that you'll have to dig into the **/var/cache/apt/archives** and see the timestamp of download, unless that has been cleaned by the command

```
# apt-get clean
```

Happy finding the dates & happy eating the head of your admin ☺

Rohit Srivastwa
rohit@clubhack.com

Saving your identity online is on your own hands

deepranjan@chmag.in