BadSense
Using Google’s advertising mechanisms as a hacking tool

O. Lutzky    T. Meiri    G. Treger

Department of Computer Science
Technion — Israel Institute of Technology

September 15th, 2008
Attacks du-jour

- AdSense-out
- **AdSense-in**
- AdWords-hokey-pokey
Attacks du-jour

- AdSense-out
- AdSense-in
- AdWords-hokey-pokey
Attacks du-jour

- AdSense-out
- AdSense-in
- AdWords-hokey-pokey
Outline

1 Introduction
   - Google and Hacking
   - Google’s Advertising Mechanisms
   - Technical Details

2 AdSense-based attacks
   - AdSense-Out
   - AdSense-In

3 AdWords-based attack
   - AdWords-Hokey-Pokey
Outline

1. Introduction
   - Google and Hacking
   - Google’s Advertising Mechanisms
   - Technical Details

2. AdSense-based attacks
   - AdSense-Out
   - AdSense-In

3. AdWords-based attack
   - AdWords-Hokey-Pokey
Google — A Useful Hacking Tool

Previously on Google hacking…

- Google has previously been used as a passive hacking tool, for locating vulnerable sites.

- A very useful resource for this sort of usage is available at http://johnny.ihackstuff.com.

- These attacks mostly search for strings appearing on vulnerable websites, but are limited to “legitimate” links which appear on highly-ranked Google sites.

- The Google search crawler, GoogleBot, visits relatively-unknown sites at very low frequency, limiting attack potential.
Google — A Useful Hacking Tool

Previously on Google hacking...

- Google has previously been used as a passive hacking tool, for locating vulnerable sites.
- A very useful resource for this sort of usage is available at http://johnny.ihackstuff.com.
- These attacks mostly search for strings appearing on vulnerable websites, but are limited to “legitimate” links which appear on highly-ranked Google sites.
- The Google search crawler, GoogleBot, visits relatively-unknown sites at very low frequency, limiting attack potential.
Google — A Useful Hacking Tool

Previously on Google hacking...

- Google has previously been used as a passive hacking tool, for locating vulnerable sites.
- A very useful resource for this sort of usage is available at http://johnny.ihackstuff.com.
- These attacks mostly search for strings appearing on vulnerable websites, but are limited to “legitimate” links which appear on highly-ranked Google sites.
- The Google search crawler, GoogleBot, visits relatively-unknown sites at very low frequency, limiting attack potential.
Our target
Google hacking — activate!

Our objective:
Use Google as an *active* part of an attack.

- Google is unlikely to be blocked from entering a site
- A degree of anonymity — webmaster only sees google performing the attack
- Google may be trusted to enter protected segments of a website
Our objective:

Use Google as an **active** part of an attack.

- Google is unlikely to be blocked from entering a site
- A degree of anonymity — webmaster only sees Google performing the attack
- Google may be trusted to enter protected segments of a website
Our objective:
Use Google as an *active* part of an attack.

- Google is unlikely to be blocked from entering a site
- A degree of anonymity — webmaster only sees google performing the attack
- Google may be trusted to enter protected segments of a website
Our objective:

Use Google as an **active** part of an attack.

- Google is unlikely to be blocked from entering a site
- A degree of anonymity — webmaster only sees google performing the attack
- Google may be trusted to enter protected segments of a website
Outline

1. Introduction
   - Google and Hacking
   - Google’s Advertising Mechanisms
   - Technical Details

2. AdSense-based attacks
   - AdSense-Out
   - AdSense-In

3. AdWords-based attack
   - AdWords-Hokey-Pokey
AdWords

Ad•Word  n., pl. -s

- AdWords allows customers to buy ads to be displayed.
- Bids are placed on keywords; payment is usually per-click.
- Bids can be placed on specific URLs as well — in this case, payment may be per-impression. This is known as site-targeted advertising.
AdSense

AdSense n.

- AdSense allows webmasters to place an ad “billboard” on their site, sharing Google’s revenue per click.
- When a user enters a site showing AdSense ads, the browser runs a script which requests a relevant ad for the page.
- Google immediately sends MediaBot to scan the reported “current page” for relevant keywords, as part of the ad selection process.
AdSense

AdSense n.

- **AdSense** allows webmasters to place an ad “billboard” on their site, sharing Google’s revenue per click.

- When a user enters a site showing AdSense ads, the browser runs a script which requests a relevant ad for the page.

- Google *immediately* sends *MediaBot* to scan the reported “current page” for relevant keywords, as part of the ad selection process.
Outline

1. Introduction
   - Google and Hacking
   - Google’s Advertising Mechanisms
   - Technical Details

2. AdSense-based attacks
   - AdSense-Out
   - AdSense-In

3. AdWords-based attack
   - AdWords-Hokey-Pokey
HTTP GET requests are the simpler form of HTTP communication; all parameters (if any) are passed as part of the URL. This is the ordinary request sent when visiting a page.

Example: http://www.google.com/search?q=differential+cryptanalysis%3F&btnG=Search
HTTP POST requests can have additional parameters in the request body.

POST requests are usually generated by web forms.

It is considered “best practice” to place all potentially destructive operations behind a POST request. (For example: Some applications pre-fetch all links on a page without user intervention).

For our demonstration, we’ll assume the victim site does not always follow these practices.
HTTP POST requests can have additional parameters in the request body.

POST requests are usually generated by web forms.

It is considered “best practice” to place all potentially destructive operations behind a POST request. (For example: Some applications pre-fetch all links on a page without user intervention).

For our demonstration, we’ll assume the victim site does not always follow these practices.
HTTP POST requests can have additional parameters in the request body.

POST requests are usually generated by web forms.

It is considered “best practice” to place all potentially destructive operations behind a POST request. (For example: Some applications pre-fetch all links on a page without user intervention).

For our demonstration, we’ll assume the victim site does not always follow these practices.
Some sites are naive when passing user data to SQL.

For example, when authenticating a user, the site might use the following code:

```sql
query("SELECT COUNT(*) FROM users WHERE user='" + $POST['user'] + "' AND pass='" + $POST['password'] + "'"")
```
Some sites are naive when passing user data to SQL.

For example, when authenticating a user, the site might use the following code:

```sql
query("SELECT COUNT(*) FROM users WHERE user = " + $POST['user'] + " AND pass = " + $POST['password']")
```

Comic from [http://xkcd.com/327](http://xkcd.com/327), Licensed Creative Commons BY-NC, some rights reserved
Introduction

AdSense-based attacks

AdWords-based attack

Summary

Outline

1. Introduction
   - Google and Hacking
   - Google’s Advertising Mechanisms
   - Technical Details

2. AdSense-based attacks
   - AdSense-Out
   - AdSense-In

3. AdWords-based attack
   - AdWords-Hokey-Pokey
The AdSense mechanism detailed
MediaBot

Site with AdSense ads

Browser

Google & MediaBot
MediaBot

Site with AdSense ads

Browser

Google & MediaBot

Browser requests page from website
Website returns page HTML; includes reference to AdSense script
Introduction
AdSense-based attacks
AdWords-based attack
Summary

MediaBot

Site with AdSense ads

Browser requests AdSense script from Google
Site with AdSense ads

Browser

Google & MediaBot

Google sends AdSense script to the browser, which runs it
MediaBot

Site with AdSense ads

Browser

Google & MediaBot

Script reports current page, requesting an appropriate ad from Google
Google sends the site a request for the reported page
Site returns the requested page; Google scans it for keywords
MediaBot

Site with AdSense ads

Browser

Google & MediaBot

Google picks an ad and sends it to the browser
Browser requests page from website.

Website returns page HTML; includes reference to AdSense script.

Browser requests AdSense script from Google.

Google sends AdSense script to the browser, which runs it.

Script reports current page, requesting an appropriate ad from Google.

MediaBot sends the site a request for the reported page.

Site returns the requested page; Google scans it for keywords.

Google picks an ad and sends it to the browser.
AdSense summary

1. Browser requests page from website.
2. Website returns page HTML; includes reference to AdSense script.
3. Browser requests AdSense script from Google.
4. Google sends AdSense script to the browser, which runs it.
5. Script reports current page, requesting an appropriate ad from Google.
6. MediaBot sends the site a request for the reported page.
7. Site returns the requested page; Google scans it for keywords.
8. Google picks an ad and sends it to the browser.
AdSense summary

1. Browser requests page from website.
2. Website returns page HTML; includes reference to AdSense script.
3. Browser requests AdSense script from Google.
4. Google sends AdSense script to the browser, which runs it.
5. Script reports **current page**, requesting an appropriate ad from Google.
6. MediaBot sends the site a request for the **reported page**.
7. Site returns the requested page; Google scans it for keywords.
8. Google picks an ad and sends it to the browser.
There is currently no a priori verification of which pages can be reported as the “current page”.

There is no filtering of GET parameters (this makes sense, as ads can be shown on internal parts of dynamic sites).

There is a limit on the reported URL length. . . .

. . . but MediaBot follows HTTP redirects.
There is currently no a priori verification of which pages can be reported as the “current page”.

There is no filtering of GET parameters (this makes sense, as ads can be shown on internal parts of dynamic sites).

There is a limit on the reported URL length…

…but MediaBot follows HTTP redirects.
There is currently no a priori verification of which pages can be reported as the “current page”.

There is no filtering of GET parameters (this makes sense, as ads can be shown on internal parts of dynamic sites).

There is a limit on the reported URL length. . .

. . . but MediaBot follows HTTP redirects.
AdSense-out

- There is currently no a priori verification of which pages can be reported as the “current page”.
- There is no filtering of GET parameters (this makes sense, as ads can be shown on internal parts of dynamic sites).
- There is a limit on the reported URL length…
- …but MediaBot follows HTTP redirects.
Our attack:

Request ads for SQL-injecting (or otherwise malicious) URLs

- We can generate these requests by manually crafting HTTP requests simulating an ad displayed on the malicious URL.
- Alternatively, we can pass the `google_page_url` parameter to the AdSense script.
Our attack:

Request ads for SQL-injecting (or otherwise malicious) URLs

- We can generate these requests by manually crafting HTTP requests simulating an ad displayed on the malicious URL.
- Alternatively, we can pass the `google_page_url` parameter to the AdSense script.
Our attack:

Request ads for SQL-injecting (or otherwise malicious) URLs

- We can generate these requests by manually crafting HTTP requests simulating an ad displayed on the malicious URL.
- Alternatively, we can pass the `google_page_url` parameter to the AdSense script.
DEMO I

AdSense-out with SQL injection
Outline

1. Introduction
   - Google and Hacking
   - Google’s Advertising Mechanisms
   - Technical Details

2. AdSense-based attacks
   - AdSense-Out
   - AdSense-In

3. AdWords-based attack
   - AdWords-Hokey-Pokey
Browser requests page from website.
2 Website returns page HTML; includes reference to AdSense script.
3 Browser requests AdSense script from Google.
4 Google sends AdSense script to the browser, which runs it.
5 Script reports **current page**, requesting an appropriate ad from Google.
6 MediaBot sends the site a request for the **reported page**.
7 Site returns the requested page; Google scans it for keywords.
8 Google picks an ad and sends it to the browser.
Browser requests page from website.
Website returns page HTML; includes reference to AdSense script.
Browser requests AdSense script from Google.
Google sends AdSense script to the browser, which runs it.
Script reports **current page**, requesting an appropriate ad from Google.
MediaBot sends the site a request for the **reported page**.
Site returns the requested page; Google scans it for keywords.
Google picks an ad and sends it to the browser.
Retrieving data from the attack

- Ordinary SQL injection attacks provide the attacker with some information
  - Success status of the attack
  - Useful SQL errors
  - Yielding hidden information using UNION:
    
    ```sql
    SELECT name, description FROM products
    WHERE name='' OR 1=0 UNION SELECT name, creditcard FROM customers --'
    
    From AdSense requests we get... an advertisement... relevant to keywords on the “current page”.

Ohad Lutzky, Tom Meiri, Guy Treger, Another BadSense
Retrieving data from the attack

- Ordinary SQL injection attacks provide the attacker with some information
  - Success status of the attack
  - Useful SQL errors
  - Yielding hidden information using UNION:
    ```sql
    SELECT name, description FROM products
    WHERE name=' OR 1=0 UNION SELECT name, creditcard FROM customers --'
    ```
  - From AdSense requests we get... an advertisement... relevant to keywords on the “current page”.

Ohad Lutzky, Tom Meiri, Guy Treger, Another BadSense
Retrieving data from the attack

- Ordinary SQL injection attacks provide the attacker with some information
  - Success status of the attack
  - Useful SQL errors
  - Yielding hidden information using UNION:
    
    ```
    SELECT name, description FROM products
    WHERE name=''' OR 1=0 UNION SELECT name, creditcard FROM customers --'
    ```

- From AdSense requests we get... an advertisement... relevant to keywords on the “current page”.
Retrieving data from the attack

- Ordinary SQL injection attacks provide the attacker with some information
  - Success status of the attack
  - Useful SQL errors
  - Yielding hidden information using UNION:
    
    ```sql
    SELECT name, description FROM products
    WHERE name=' OR 1=0 UNION SELECT name, creditcard FROM customers --'
    
    
    From AdSense requests we get... an advertisement... relevant to keywords on the “current page”.
Retrieving data from the attack

- Ordinary SQL injection attacks provide the attacker with some information
  - Success status of the attack
  - Useful SQL errors
  - Yielding hidden information using UNION:
    ```sql
    SELECT name, description FROM products
    WHERE name=' ' OR 1=0 UNION SELECT name, creditcard FROM customers --'
    ```

- From AdSense requests we get... an advertisement... relevant to keywords on the “current page”.

Ohad Lutzky, Tom Meiri, Guy Treger, Another BadSense
Retrieving data from the attack

- Ordinary SQL injection attacks provide the attacker with some information
  - Success status of the attack
  - Useful SQL errors
  - Yielding hidden information using UNION:
    ```sql
    SELECT name, description FROM products
    WHERE name=' ' OR 1=0 UNION SELECT name, creditcard FROM customers --'
    ```
- From AdSense requests we get... an advertisement... relevant to keywords on the “current page”.

Ohad Lutzky, Tom Meiri, Guy Treger, Another BadSense
Ordinary SQL injection attacks provide the attacker with some information
- Success status of the attack
- Useful SQL errors
- Yielding hidden information using UNION:
  
  ```sql
  SELECT name, description FROM products
  WHERE name=' OR 1=0 UNION SELECT name, creditcard FROM customers --'
  ```

From AdSense requests we get... an advertisement... relevant to keywords on the “current page”.

Ohad Lutzky, Tom Meiri, Guy Treger, Another BadSense
Planting information in advertisement

Our attack:
“Plant” an ad topic in the returned page in case of a successful attack

or:

Our attack:
“Plant” an ad topic in the returned page if a binary SQL query returns true
Our attack:
“Plant” an ad topic in the returned page in case of a successful attack

or:

Our attack:
“Plant” an ad topic in the returned page if a binary SQL query returns true
Naive example — how many NULLs?

In order to perform a UNION attack, we need to know how many fields the origin table has.

http://victim.com/search?q=' AND 1=0 UNION SELECT null,null,null null --

http://victim.com/search?q=' AND 1=0 UNION SELECT 'car automobile mechanic clutch muffler exhaust',null,null null --
Naive example — how many NULLs?

In order to perform a UNION attack, we need to know how many fields the origin table has.

http://victim.com/search?q=’ AND 1=0 UNION SELECT null,null,null,null --

http://victim.com/search?q=’ AND 1=0 UNION SELECT ‘car automobile mechanic clutch muffler exhaust’,null,null,null --
Naive example — how many NULLs?

In order to perform a UNION attack, we need to know how many fields the origin table has.

http://victim.com/search?q=' AND 1=0 UNION SELECT null,null,null null --

http://victim.com/search?q=' AND 1=0 UNION SELECT ‘car automobile mechanic clutch muffler exhaust’,null,null null --
First caveat

MediaBot also chooses keywords\(^1\) from the URL, so car-related ads will always be shown.

Solution:

```
http://victim.com/search?q=' AND 1=0 UNION SELECT CAST(0x63006100720020006100750074006f006d006f00620069006c00650020006d0065006300680061006e0069006300200063006c00750074006300680020006d007500660066006c006500720020006500780068006100750073007400 as nvarchar(1000)),null,null --
```

\(^1\)Even URLEncoded ones!
First caveat

MediaBot also chooses keywords\(^1\) from the URL, so car-related ads will always be shown.

Solution:

\[
\text{http://victim.com/search?q=' AND 1=0 UNION SELECT CAST(0x63006100720020006100750074006f006d006f00620069006c00650020006d0065006300680061007500730074006300680020006d007500660066006c006500720020006500780068006100750073007400 as nvarchar(1000)),null,null --}
\]

\(^1\)Even URLEncoded ones!
Second caveat

Using SQL CAST requires extremely long URLs.

Solution: http://tinyurl.com/566664
Using SQL CAST requires extremely long URLs.

Solution: http://tinyurl.com/566664
DEMO II

AdSense-in
Introduction

AdSense-based attacks

AdWords-based attack

Summary

Outline

1. Introduction
   - Google and Hacking
   - Google’s Advertising Mechanisms
   - Technical Details

2. AdSense-based attacks
   - AdSense-Out
   - AdSense-In

3. AdWords-based attack
   - AdWords-Hokey-Pokey
Cross-site attacks
And now for something completely different.

Many destructive operations can only be done by a logged-in user, on his own account.

Many browsers today have shared sessions between multiple windows or tabs.

Some attackers use this fact in order to conduct a cross-site attack

1. The victim is already logged into a certain site.
2. The victim visits attacker.com with the same browser.
3. Attacker.com causes the browser to perform a destructive operation in the other site.
Many destructive operations can only be done by a logged-in user, on his own account.

Many browsers today have shared sessions between multiple windows or tabs.

Some attackers use this fact in order to conduct a cross-site attack

1. The victim is already logged into a certain site.
2. The victim visits attacker.com with the same browser.
3. attacker.com causes the browser to perform a destructive operation in the other site.
Cross-site attacks
And now for something completely different... 

Many destructive operations can only be done by a logged-in user, on his own account.

Many browsers today have shared sessions between multiple windows or tabs.

Some attackers use this fact in order to conduct a cross-site attack

1. The victim is already logged into a certain site.
2. The victim visits attacker.com with the same browser.
3. attacker.com causes the browser to perform a destructive operation in the other site.
Cross-site attacks
And now for something completely different... 

- Many destructive operations can only be done by a logged-in user, on his own account.
- Many browsers today have shared sessions between multiple windows or tabs.
- Some attackers use this fact in order to conduct a cross-site attack
  1. The victim is already logged into a certain site.
  2. The victim visits attacker.com with the same browser.
  3. attacker.com causes the browser to perform a destructive operation in the other site.
Cross-site attacks
And now for something completely different…

- Many destructive operations can only be done by a logged-in user, on his own account.
- Many browsers today have shared sessions between multiple windows or tabs.
- Some attackers use this fact in order to conduct a cross-site attack
  1. The victim is already logged into a certain site.
  2. The victim visits attacker.com with the same browser.
  3. attacker.com causes the browser to perform a destructive operation in the other site.
Cross-site attacks
And now for something completely different…

Many destructive operations can only be done by a logged-in user, on his own account.

Many browsers today have shared sessions between multiple windows or tabs.

Some attackers use this fact in order to conduct a cross-site attack

1. The victim is already logged into a certain site.
2. The victim visits attacker.com with the same browser.
3. attacker.com causes the browser to perform a destructive operation in the other site.
Cross-site attacks
And now for something completely different…

- Many destructive operations can only be done by a logged-in user, on his own account.
- Many browsers today have shared sessions between multiple windows or tabs.
- Some attackers use this fact in order to conduct a cross-site attack
  1. The victim is already logged into a certain site.
  2. The victim visits attacker.com with the same browser.
  3. attacker.com causes the browser to perform a destructive operation in the other site.
Cross-site attacks (contd.)

- AdWords allows advertisers to select specific URLs to advertise on.

Our attack:
Advertise a malicious URL, requesting that the ad be shown on a page which suggests that the user is logged into the attacked site.

- AdWords allows any GET parameters to be passed as the URL, but the URL is limited to 1024 characters. Longer URLs will require advertising a malicious site rather than a malicious URL.
AdWords allows advertisers to select specific URLs to advertise on.

**Our attack:**
Advertise a malicious URL, requesting that the ad be shown on a page which suggests that the user is logged into the attacked site.

AdWords allows any GET parameters to be passed as the URL, but the URL is limited to 1024 characters. Longer URLs will require advertising a malicious site rather than a malicious URL.
Cross-site attacks (contd.)

- AdWords allows advertisers to select specific URLs to advertise on.

Our attack:

Advertise a malicious URL, requesting that the ad be shown on a page which suggests that the user is logged into the attacked site.

- AdWords allows any GET parameters to be passed as the URL, but the URL is limited to 1024 characters. Longer URLs will require advertising a malicious site rather than a malicious URL.
DEMO III

Cross-site AdWords attack
Summary

- Google’s advertising mechanism can be used as an **active** attack tool.
- Websites can leak information via contextual advertising services.
- Always take Google utilities into account when considering the security of a website.

Outlook

- Can other Google utilities be used for active attacks?
- Is it Google’s responsibility to protect against these attacks?
Google’s advertising mechanism can be used as an **active** attack tool.

**Websites can leak information via contextual advertising services.**

Always take Google utilities into account when considering the security of a website.

**Outlook**
- Can other Google utilities be used for active attacks?
- Is it Google’s responsibility to protect against these attacks?
Google’s advertising mechanism can be used as an active attack tool.

Websites can leak information via contextual advertising services.

Always take Google utilities into account when considering the security of a website.

Outlook

- Can other Google utilities be used for active attacks?
- Is it Google’s responsibility to protect against these attacks?
Google’s advertising mechanism can be used as an active attack tool.

Websites can leak information via contextual advertising services.

Always take Google utilities into account when considering the security of a website.

Outlook
- Can other Google utilities be used for active attacks?
- Is it Google’s responsibility to protect against these attacks?
REMEMBER:

Always sanitize your database inputs!

Put all destructive operations behind a POST request!