Bypassing All Web Application Firewalls

OuTian
<outian@chroot.org>

HIT Conference 2009

Agenda

- **Introduction**
- **What is WAF**
- **Why need WAF**
- **What does WAF do**
- **How to Bypass WAF**
- **Q & A**

Introduction

- 近年來許多企業開始意識到傳統的資安設備無法防護針對 **Web** 應用程式的攻擊
- 因此紛紛開始佈署「**Web Application Firewall**」（以下簡稱**WAF**）
- 本主題要強調的是—**WAF**並不是萬靈藥，絕對沒有 **100%** 的防禦能力，不要再聽信沒有根據的謠言了！
- 在設定不當的情況下，有裝跟沒裝一樣 **…**

傳說中只要拔到獅子的鬃毛 .....

About Me

- **OuTian < outian@chroot.org >**
  - 會唸的人叫我 幺ㄟ 士一ㄢ
  - 不會唸的人叫我「黑糖」、「凹臀」、「熬湯」
- 現任
  - 敦陽科技 資安服務處 資安顧問
- 經歷 –
  - **HIT2007 –「Implementation of Web Application Firewall」**
  - **HIT2007/2008 0day Advisory**
- 專長 –
  - 滲透測試、資安設備佈署
  - **DDoS攻擊與防護、資安事件緊急應變**

What is WAF

- 深入解析HTTP、HTML、XML內容之
  - 網路硬體設備
  - 主機式軟體
- 處理 Client 與 Web Server 間之傳輸
- 用以防禦針對動態網頁應用程式之攻擊
- 避免內部之敏感訊息或資料外洩

WAF Vendors (in TW)

◆ (廠牌)	-	(產品名稱)
◆ AppliCure	-	dotDefender
◆ Armorize	-	SmartWAF
◆ Barracuda	-	Web Application Controller
◆ Cisco	-	ACE
◆ Citrix	-	NetScaler
◆ F5	-	Big-IP / ASM
◆ Imperva	-	SecureSphere
◆ Radware	-	AppWall

◆ ...others

◆ (以上依廠牌名稱排序)

WAF Vendors (Global)

- BeeWare
- BinarySEC
- Breach / ModSecurity
- Deny All
- Visonys
- ... others

常見 Web 應用程式弱點 (1)

- 程式過濾不當
  - SQL Injection
    - 竊取資料、入侵網站
  - **Cross Site Scripting**
    - 利用網站弱點竊取其他用戶資料
  - Arbitrary File Inclusion
    - 入侵網站
  - Code/Command Injection
    - 入侵網站
  - **Directory Traversal**
    - 瀏覽敏感資訊檔案
  - **Buffer Overflow**
    - 入侵網站主機

常見 Web 應用程式弱點 (2)

- 邏輯設計不良
  - Cookie Poisoning
    - 變換身份、提升權限
  - **Parameter Tampering**
    - 竄改參數，使應用程式出現不可預期反應
  - Upload File Mis-Handling
    - 植入網站木馬
  - **Information Disclosure**
    - 洩露網站資訊
  - **Weak Authentication**
    - 脆弱的認證機制

WAF v.s IDP/IPS

- 入侵偵測系統
  - **Negative Security Model**
    (負向表列黑名單)
  - 特表碼辨識
    Signature based
  - 無法解析 **SSL** 封包
  - 不追蹤 表單/**Cookie**

- 網頁防火牆
  - **Positive Security Model**
    (正向表列白名單)
  - 行為模式分析
    Behavior Modeling
  - 置入金鑰/憑證，可解析**SSL**封包
  - 會追蹤表單/**Cookie**

What does WAF do ？

- **Input Validation**
  - Protocol
  - URL
  - Parameter
  - Cookie/Session
- **Output Checks**
  - Protocol
  - Headers
  - Error Messages
  - Credit Card Number
  - Sensitive Information

Input Validation

- **Normal HTTP Request**

URL Parameter

```
GET /search?q=test HTTP/1.1
Accept: */*
Accept-Language: zh-tw
User-Agent: Mozilla/4.0
Accept-Encoding: gzip, deflate
Host: www.google.com.tw
Connection: Keep-Alive
Cookie: SESSIONID=8E938AF24D97
```

Protocol

Cookies

Protocol Protection

- Buffer Overflow
- Denial of Service
- Abnormal
  - HTTP Method
    - GET/POST/HEAD
    - CONNECT
    - PUT
    - DELETE
  - HTTP Headers
    - Host
    - User-Agent
    - Content Length

URL Protection

- Forceful Browsing
- Configuration Files
  - *.inc 、 *.cfg 、 *.log
- Database Files
  - *.sql 、 *.mdb
- Backup Files
  - *.bak 、 *.old 、 *.tmp 、 *~
- Archive Files
  - *.rar 、 *.zip 、 *.tgz
- Document Files
  - *.pdf 、 *.xls 、 …

Parameter Protection

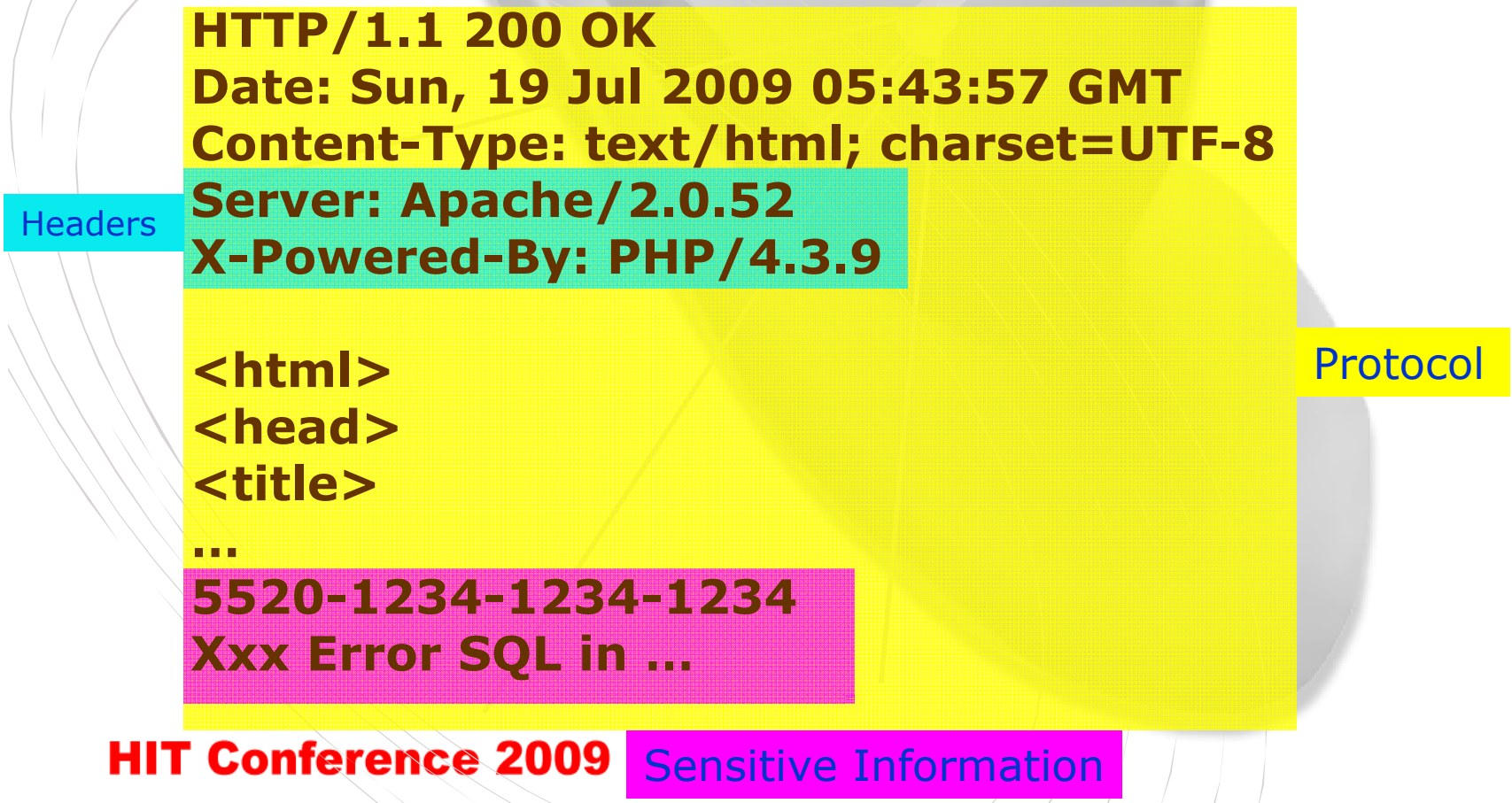
- SQL/Code/Command Injection
- Cross Site Scripting
- Arbitrary File Inclusion
- Directory Traversal
- Parameter Tampering

Cookie Protection

- Session Stealing
- Cookie Poisoning

Output Checks

- **Normal HTTP Response**



Header Protection

- 刪除、修改特定Header
  - Ex:
    - Server
    - X-Powered-By
- 部份廠牌具有 Cookie Proxy / Cookie Encryption 功能

Sensitive Information Protection

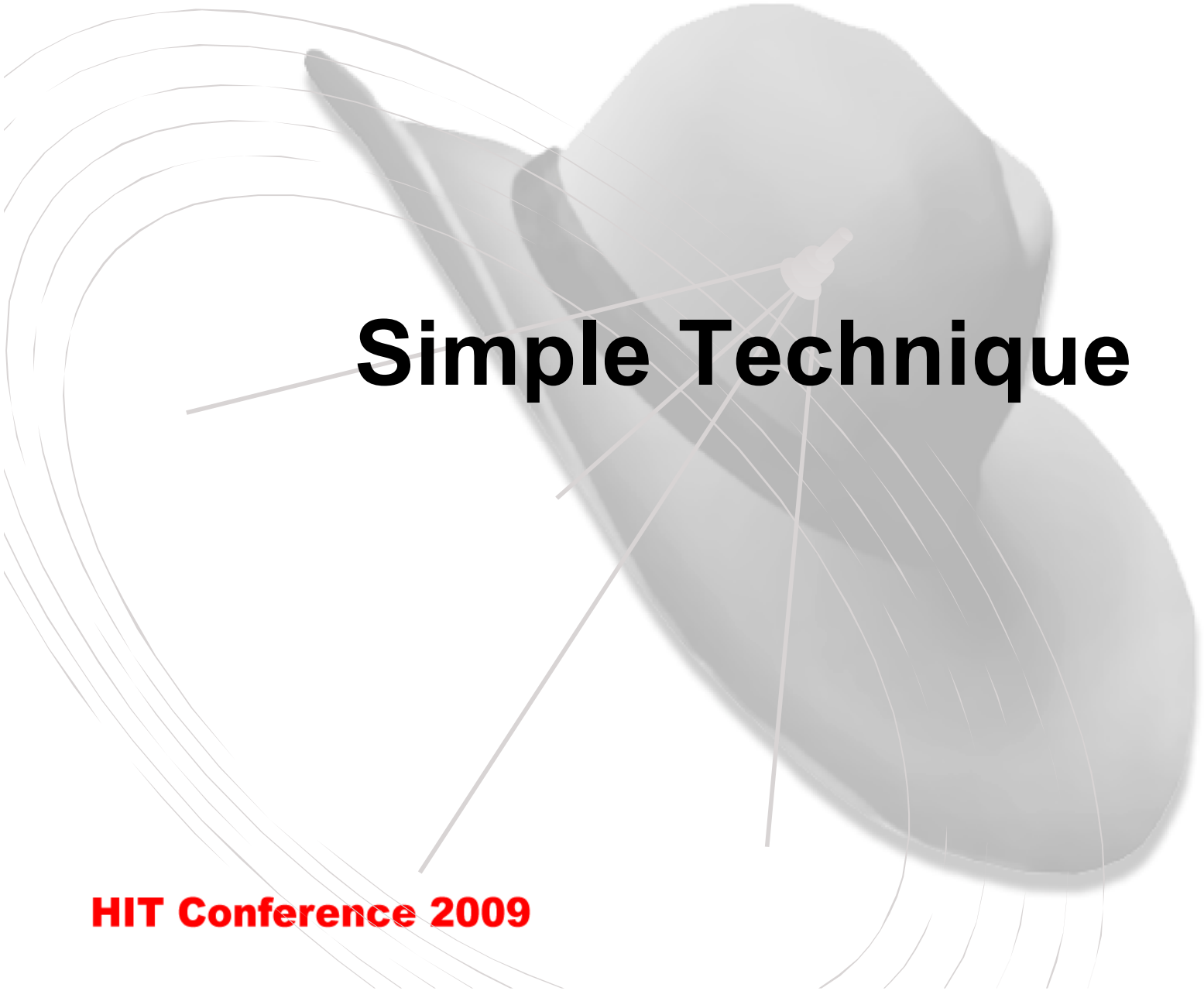
- 攔截敏感訊息
  - 信用卡卡號
  - 伺服器錯誤訊息
  - 資料庫錯誤訊息
  - 指定格式之個人資料字串
- 處理方式
  - 刪除
  - 打馬賽客（XXX or ***）
  - 攔截整個頁面

正向表列 V.S 負向表列

- 負向表列
  - 俗稱「黑名單」
  - 佈署快速
  - 容易繞過
  - 容易誤判
- 正向表列
  - 俗稱「白名單」
  - 需時間學習/設定
  - 防護嚴謹
  - 不會誤判 (除非管理者設定錯誤)

How to Bypass WAF

- Simple Technique
- Negative Model
  - Magic %
  - HTTP Parameter Pollution
  - Special Check
- Positive Model
  - Bypass Condition



Simple Technique

簡單的方法 (通常都已防範)

- 大小寫轉換 (多數WAF忽略大小寫作檢查)
  - 在 Windows 系統裡，
    test.asp == TEST.ASP
- 跳脫字元
  - 某些情況下，
    a = \a
- URL編碼 (多數WAF 先作 URL解碼後才作檢查)
  - 路徑編碼
    - /test.asp
      = /%74%65%73%74%2E%61%73%70
  - 參數編碼
    - /etc/passwd
      =
      %2F%65%74%63%2F%70%61%73%73%77%64

空白字元的替代方案

- 常見替代字串
  - （空白）= %20
  - \t (TAB) = %09
  - \n = %0A
  - \r = %0D
- in SQL
  - /**/ for MSSQL
- in XSS
  - /**/ in some case

模糊路徑

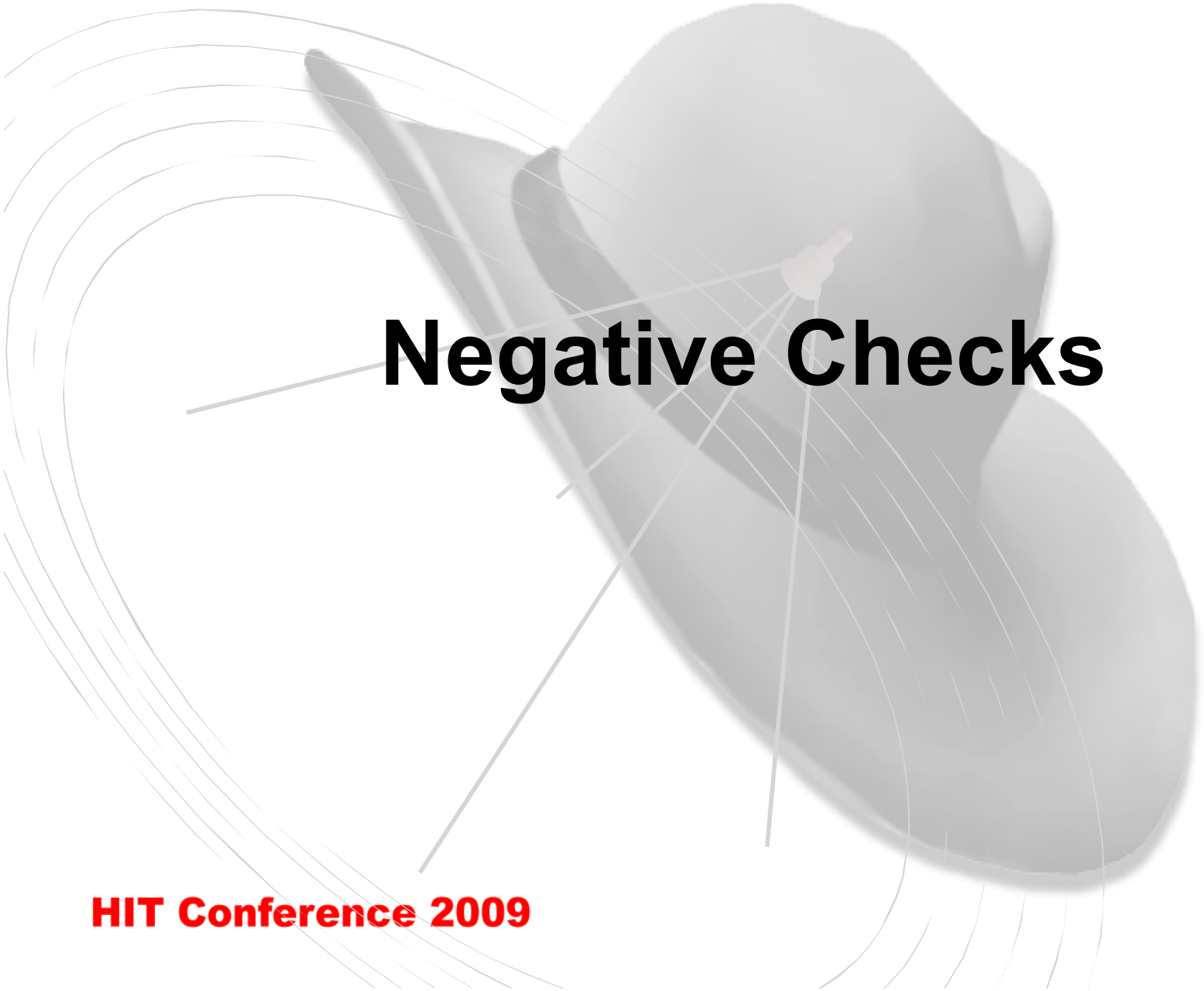
- 自我參考目錄
  - /test.asp == /./test.asp
- 雙目錄分隔線
  - /test.asp == //test.asp
- 目錄跳脫
  - /etc/passwd == /etc/./passwd
  - /etc/passwd ==/etc/xx/../passwd
- 目錄分隔符號
  - ../../cmd.exe == ..\..\cmd.exe

較複雜的編碼

- **Double Decoding**
  - /
    = %2F
    = %252F
- **Overlong characters**
  - 0xc0 0x8A
    = 0xe0 0x80 0x8A
    = 0xf0 0x80 0x80 0x8A
    = 0xf8 0x80 0x80 0x80 0x8A
- **Unicode Encoding**
  - /test.cgi?foo=../../bin/ls
    = /test.cgi?foo=..%2F../bin/ls
    = /test.cgi? foo=..%c0%af../bin/ls

Null-Byte Attacks

- %00
- Null Byte (0x00) 於程式語言之判斷函數中常用以代表字串中止
  - strcmp()
  - strcpy()
  - sprintf()
  - .... etc
- 許多字串檢查機制當偵測到 0x00 即停止對後方字串作檢查
  - /aa.php?cmd=ls%00cat%20/etc/passwd



Negative Checks

HIT Conference 2009

正常的編碼原理是這樣

- 字元編碼
  - A => %41
  - & => %26
  - ` => %27
- 正常Scope
  - %00 ~ %FF
- So
  - select
    => %73%65%6C%65%63%74

Magic %

- 當 % 後方兩碼不在正常範圍 ...
  - select
    - = sele%ct
    - = s%elect
  - ... 其他例子請自行延伸
- 可用於繞過所有黑名單檢查機制 (如SQL、XSS/... etc)
- 程式語言自動砍掉無效的 % !!!
- *** 僅 ASP 語言具有此特性 ***

Why Bypass？

注入語法	WAF 看到	ASP 解讀為
sele%ct * fr%om ...	sele%ct * fr%om ...	select * from ...
;dr%op %table xxx	;dr%op %table xxx	;drop table xxx
<scr%ipt>	<scr%ipt>	<script>
<if%rame>	<if%rame>	<iframe>

From blog.iis.net

UrlScan 3.1

Posted: Oct 31, 2008 13 comments

Average Rating ☆☆☆☆☆

Tags

Email this Post
DotNetKicks
Digg
Facebook
Del.icio.us

Wade Hilmo

Search My Blog

Running Perl on IIS 7
How IIS can help with...
Filtering for SQL Injection...

Earlier this year, it came to our attention that our customers were being subjected to a SQL Injection attack. In response to that, we updated the venerable UrlScan filter and released version 3.0 with new features that provide tools to provide some mitigation and allow users to address issues in their affected applications.

This effort has been largely successful, but it has driven attackers to come up with new techniques. Very recently, our internal security team brought it to our attention that they'd seen a new variation on the attacks. This new variation is trying to exploit a behavior in ASP's parsing of the query string for the Request.QueryString function. *Note that ASP.NET's behavior in this area is different and ASP.NET applications are not vulnerable to this specific new technique.*

The specific behavior in ASP results when the query string contains a name/value pair where the value contains a '%' sign that has not been escape encoded. Consider the following examples:

- If the client sends "page.asp?abc=xyz", calling Request.QueryString("abc") will return "xyz".
- If the client sends "page.asp?abc=x%79z", ASP will decode the value and Request.QueryString("abc") will also return "xyz".

But if you wanted the value to contain a percent, Request.QueryString requires the '%' sign to be encoded at %25. If you do not encode the '%' sign, Request.QueryString will drop it from the returned value. For example:

- If the client sends "page.asp?abc=x%25yz", Request.QueryString("abc") will return "x%yz".
- If the client sends "page.asp?abc=x%yz", Request.QueryString("abc") will drop the unescaped '%' and return "xyz".

HIT Conference 2009

HTTP Parameter Pollution

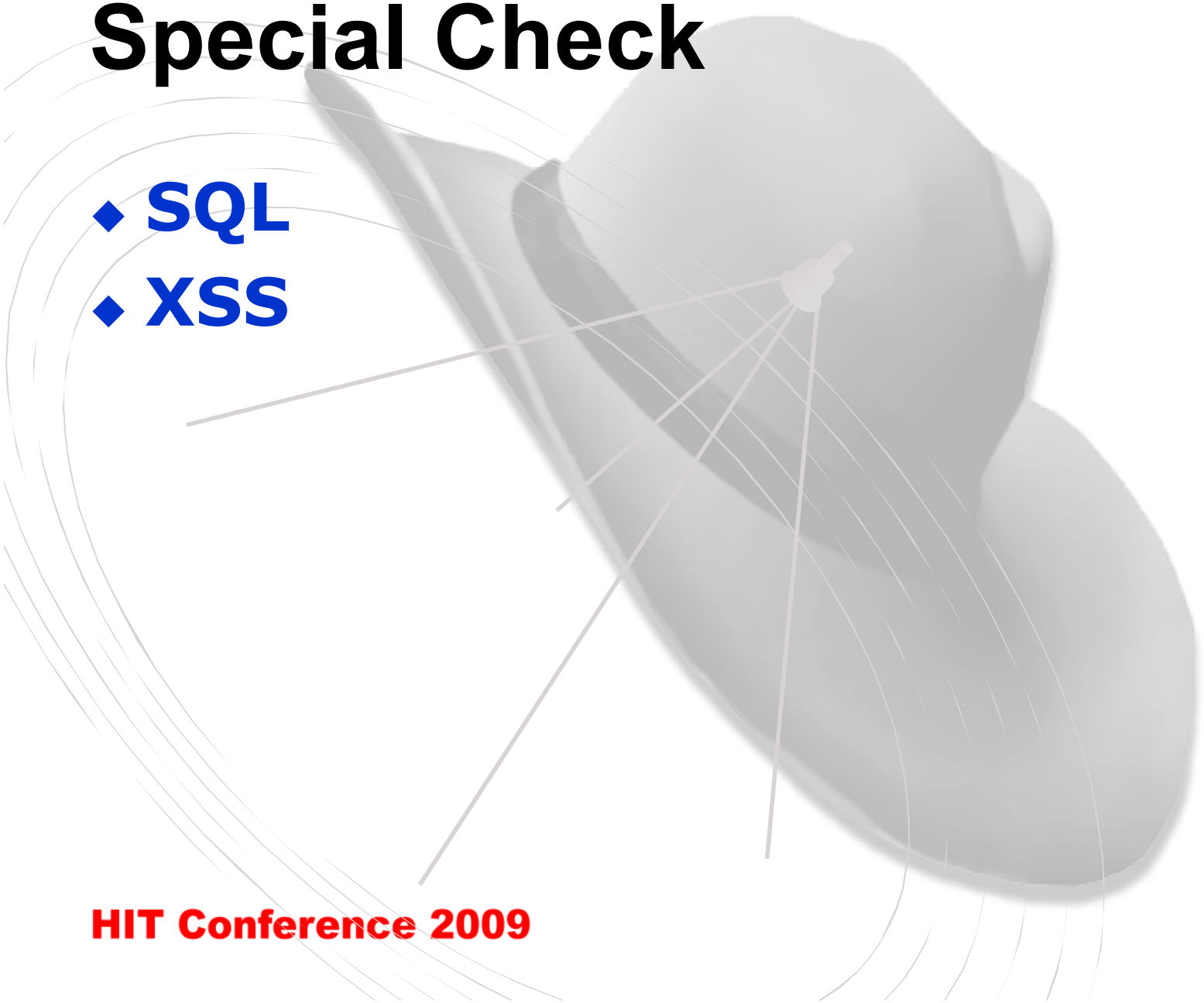
- 一般網頁應用程式中，同一個頁面、同名的參數只有一個
  - http://www.google.com.tw/search?hl=zh-TW&q=test
- 插入多個同名參數，各平台的反應不一致
  - 將各參數組合起來
  - 取第一個
  - 取最後一個
  - 成為陣列 (ARRAY)

Server enumeration

Technology/HTTP back-end	Overall Parsing Result	Example
ASP.NET/IIS	All occurrences of the specific parameter	par1=val1,val2
ASP/IIS	All occurrences of the specific parameter	par1=val1,val2
PHP/Apache	Last occurrence	par1=val2
PHP/Zeus	Last occurrence	par1=val2
JSP,Servlet/Apache Tomcat	First occurrence	par1=val1
JSP,Servlet/Oracle Application Server 10g	First occurrence	par1=val1
JSP,Servlet/Jetty	First occurrence	par1=val1
IBM Lotus Domino	Last occurrence	par1=val2
IBM HTTP Server	First occurrence	par1=val1
mod_perl,libapreq2/Apache	First occurrence	par1=val1
Perl CGI/Apache	First occurrence	par1=val1
mod_perl,lib???/Apache	Becomes an array	ARRAY(0x8b9059c)
mod_wsgi (Python)/Apache	First occurrence	par1=val1
Python/Zope	Becomes an array	['val1', 'val2']
IceWarp	Last occurrence	par1=val2
AXIS 2400	All occurrences of the specific parameter	par1=val1,val2
Linksys Wireless-G PTZ Internet Camera	Last occurrence	par1=val2
Ricoh Aficio 1022 Printer	First occurrence	par1=val1
webcamXP PRO	First occurrence	par1=val1
DBMan	All occurrences of the specific parameter	par1=val1~~val2

Special Check

- SQL
- XSS



使用方式

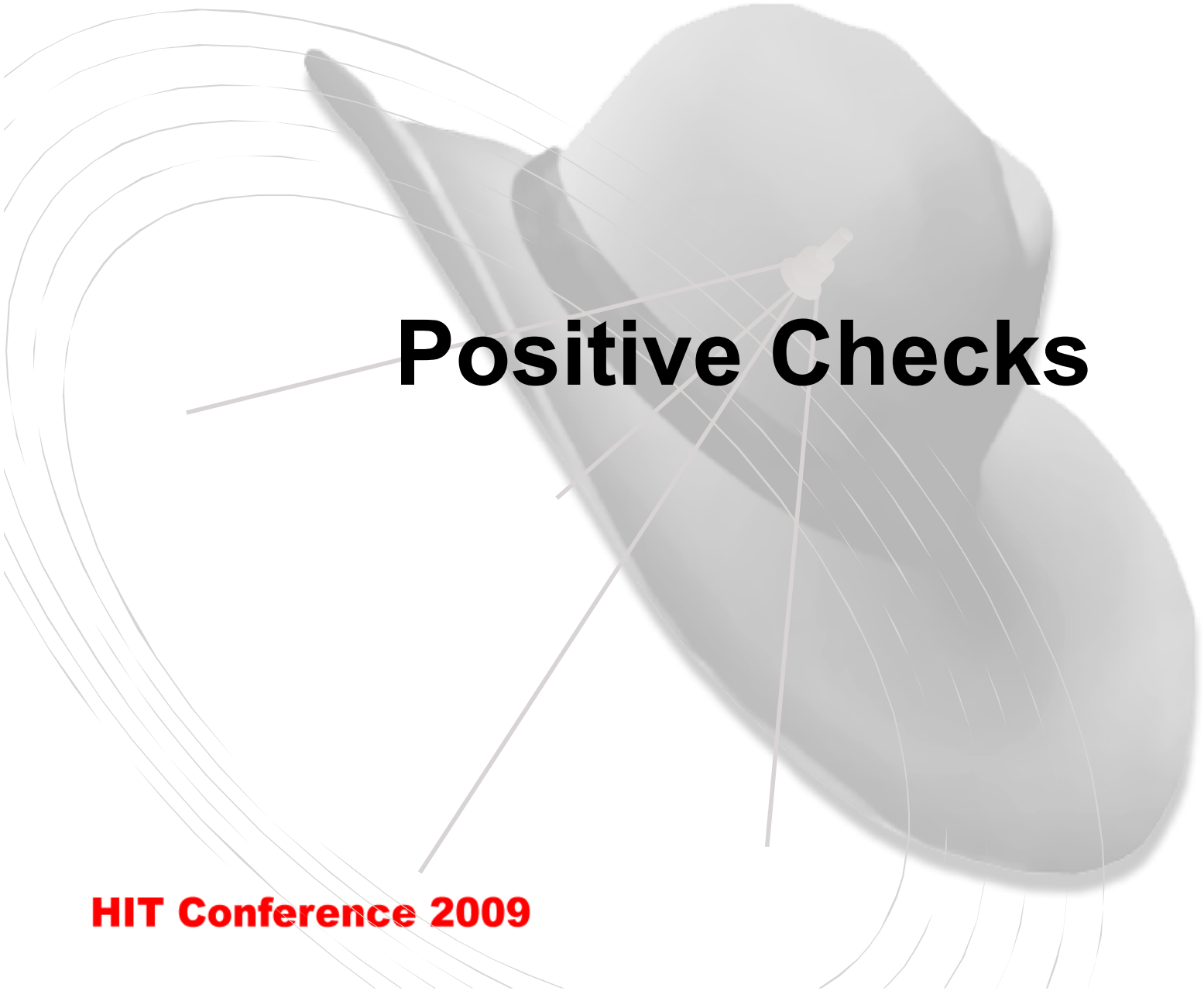
- 將欲注入的攻擊字串，拆散於同名稱之參數中
- 經過 **WAF** 時，由於並未中到任何特徵碼，因此予以放行
- 進到程式裡，將同參數之字串組合後，即變回原攻擊碼

Bypass SQL

- 多數 WAF 針對含有SQL 特徵的參數會特別深入檢查
  - `
  - ;
  - SQL 注釋
    - --
    - /*
    - #
- 設法於攻擊時不帶以上特徵
  - 攻擊數字型參數(不需 `)
  - 自行補足後方 SQL 語法，不使用注釋符號
  - Magic %

Bypass XSS

- HTML/CSS/Java Script之語法非常靈活
- 大部份 WAF 無法內建所有的pattern (容易誤擋)
- 稍微變形一下即可繞過
- XSS Cheat Sheet
  - http://ha.ckers.org/xss.html



Positive Checks

Positive Check ？

- 多數 **WAF** 雖有自動學習功能
- 可分析網站正常使用情況下的
  - **HTTP Method**
  - **URL**
  - **Parameters**
  - **Form**
  - **Cookies**
- …. 但是因為管理員很懶，絕大多數都沒有去設 **Orz**

如果有設的情況

- 規則一 –
  - http://www.test.com/news.asp
  - id
  - 限制
    - 格式為 整數 (^\d+$)
    - 長度為 1 ~ 20
- 規則二 –
  - http://www.test.com/login.asp
  - Username
  - 限制
    - 格式為英文+數字+底線 (^[_a-zA-Z0-9]+$)
    - 長度為 1 ~ 12
- …… 很多條類似的規則

繞過的方法

- ...
- ...
- 不要中到檢查的條件！
  - Policy Condition
  - URL
  - Parameter
  - ... etc

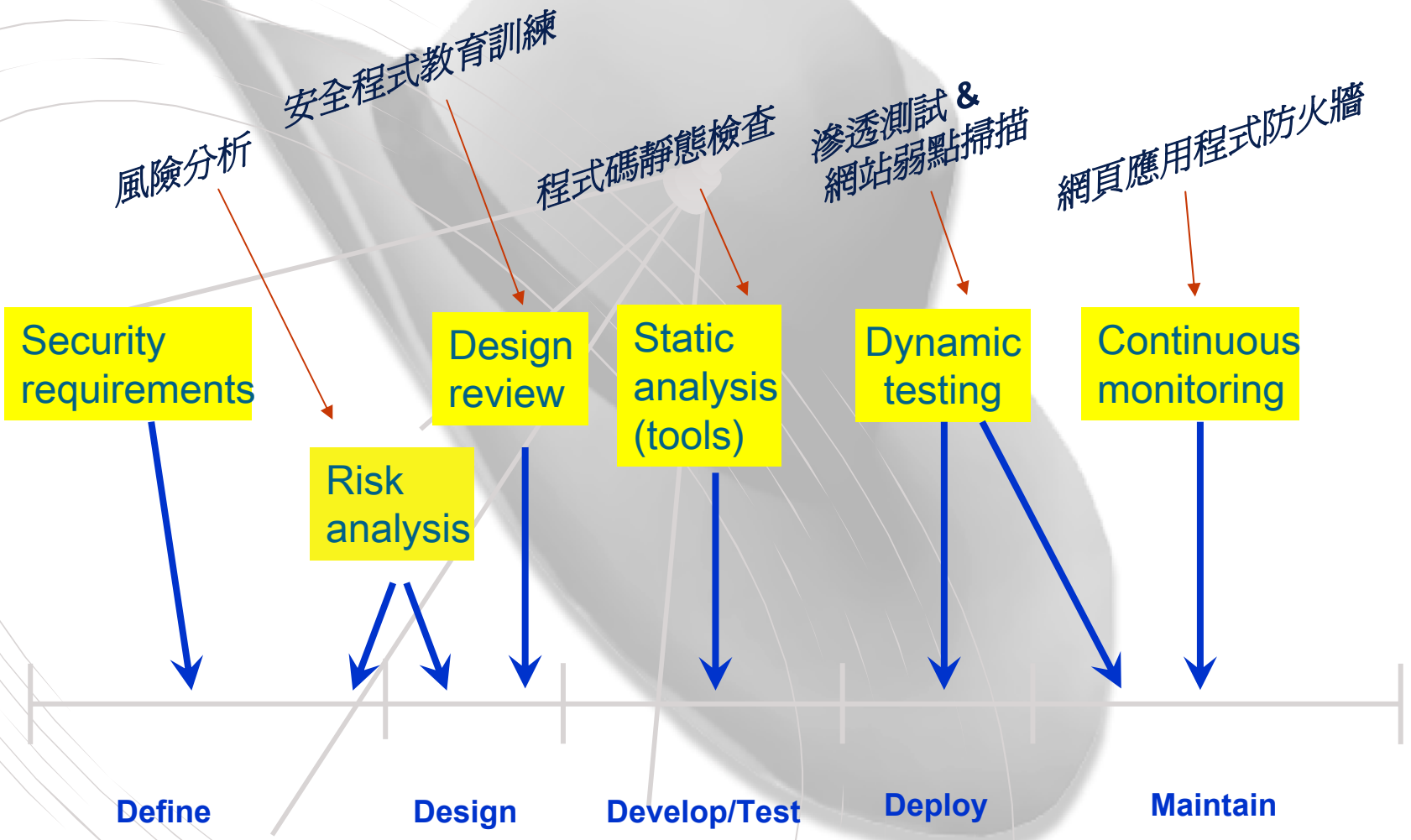
Why？

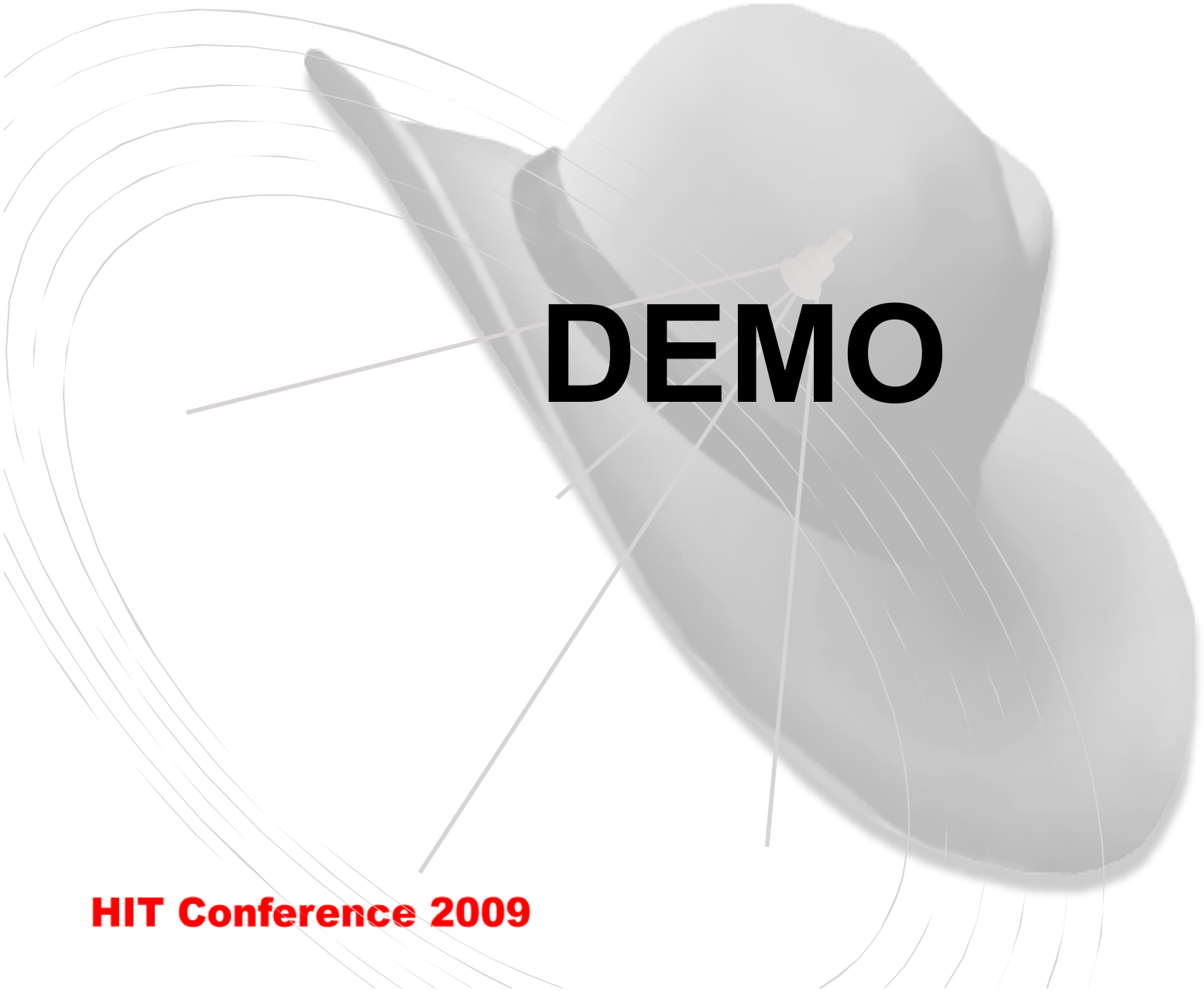
- 大網站的 URL/參數 太多太多了
- 沒設到的地方，WAF不知道格式為什麼，只好先放過、再學習
- 以先前提到之字串修改、編碼、糢糊路徑、Magic %等方法，不要中到WAF中指定之條件即可

實際案例

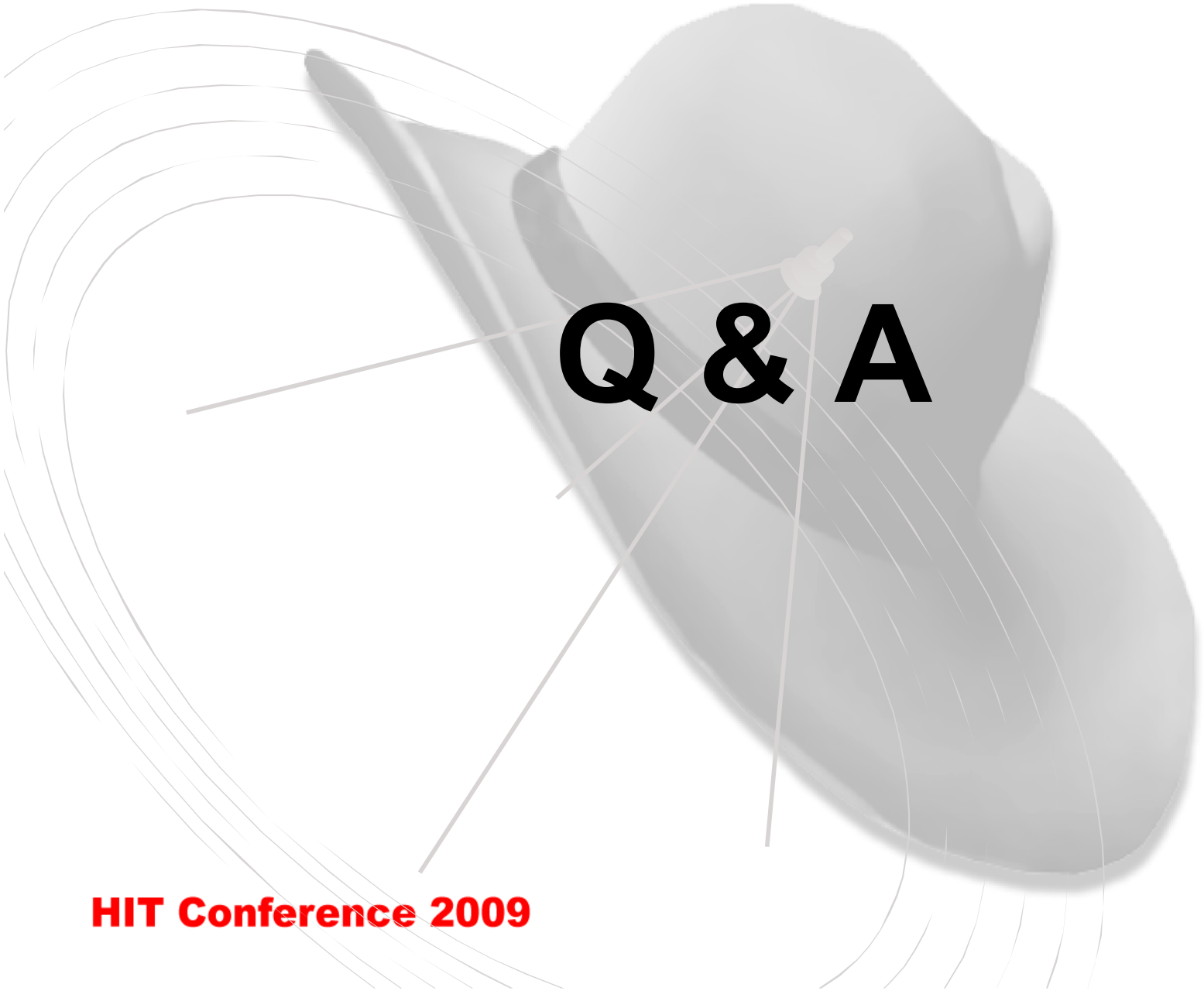
- **www.test.com = IP: x.x.x.x**
  - 設定當
    網址 == **www.test.com**
    或
    網址 == **x.x.x.x**
    時，套入某**profile**作檢查
- 繞過方法
  - 不帶 **Host Header**
  - 連接 **www.test.com:80**
  - 修改 **hosts**檔，帶入任何 **Host Header**

網頁應用程式安全防護


風險分析
安全程式教育訓練
程式碼靜態檢查
滲透測試 &
網站弱點掃描
網頁應用程式防火牆
Security requirements
Risk analysis
Design review
Static analysis (tools)
Dynamic testing
Continuous monitoring
Define
Design
Develop/Test
Deploy
Maintain



DEMO



HIT Conference 2009

Reference

- WAF Reviews
  - http://sites.google.com/a/wafreviews.com/home/Home
- OWASP AppSecEU09 Poland
  - HTTP Parameter Pollution
  - Web Application Firewalls: What the vendors do NOT want you to know
- WAFEC, or how to choose WAF technology
- Split and Join
  - http://www.milw0rm.com/papers/340
- SQL Injection Hijinks
  - http://blogs.technet.com/neilcar/archive/2008/10/31/sql-injection-hijinks.aspx