



ĐẠI HỌC BÁCH KHOA HÀ NỘI
VIỆN CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG

Security in Application Layer

1

Security in Application layer

- DNS security
- Email security
- Web security



VIỆN CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG

2

2

Security in Application layer

- DNS security
- Email security
- Web security



VIỆN CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG

3

3

DNS History

- On ARPANet, host names were mapped to IP addresses in a **hosts.txt** file stored on a single master server
- Other machines downloaded copies of **hosts.txt** periodically
- Unix stored this information in **/etc/hosts**
- This technique didn't scale well. DNS started in 1983.



VIỆN CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG

4

Design Principles

- Distributed storage
 - DNS data split across many servers
 - Smaller storage requirements for each server
 - Faster transfer of information
 - No single point of failure
- Hierarchical organization of data
 - Allows local control of names and avoids name conflicts

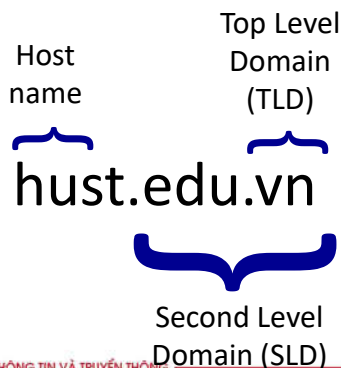
DNS Name Structure

- Up to four **labels** separated by dots form a **Fully Qualified Domain Name (FQDN)**

Host name Top Level Domain (TLD)

hust.edu.vn

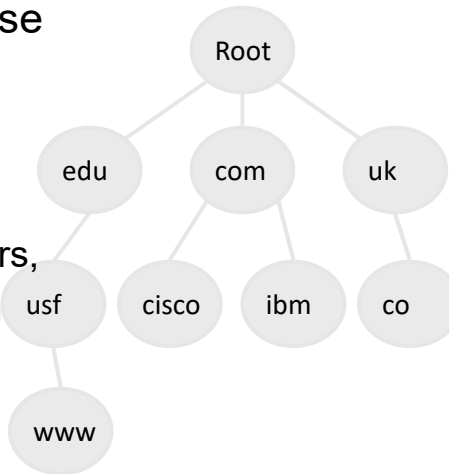
Second Level Domain (SLD)



The Domain Name System - DNS

- Basic Internet Database

- Maps names to IP addresses
- Also stores IPv6 addresses, mail servers, service locators, Enum (phone numbers), etc.



- Data organized as tree structure.



Each **zone** is the authority for its local data.

7

DNS Clients, Servers, and Resolvers

- DNS Client
 - A program like a Web browser using a domain name like **www.hust.edu.vn**
- DNS Server
 - Stores and serves DNS data
- DNS Resolver
 - Software that accepts a query from a client, queries one or more DNS servers, and replies to the client

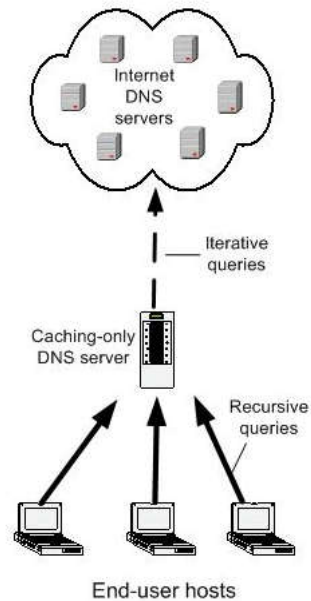


VIỆN CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG

8

DNS Servers

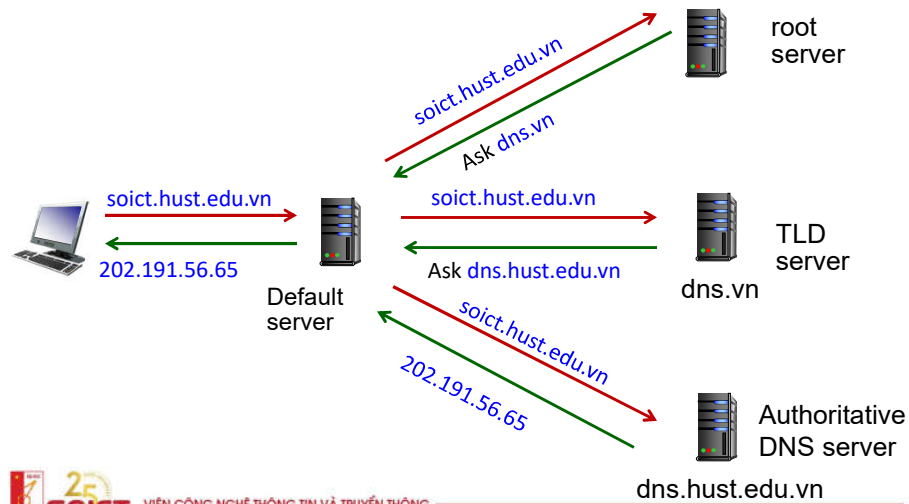
- **Authoritative** servers manage information about a domain
 - **SOA (Start Of Authority)**
- **Caching** servers store data they copied from other servers
 - Not authoritative for any domain
 - Cache records have a **Time To Live (TTL)**
 - Specified by SOA for each record



DNS Queries

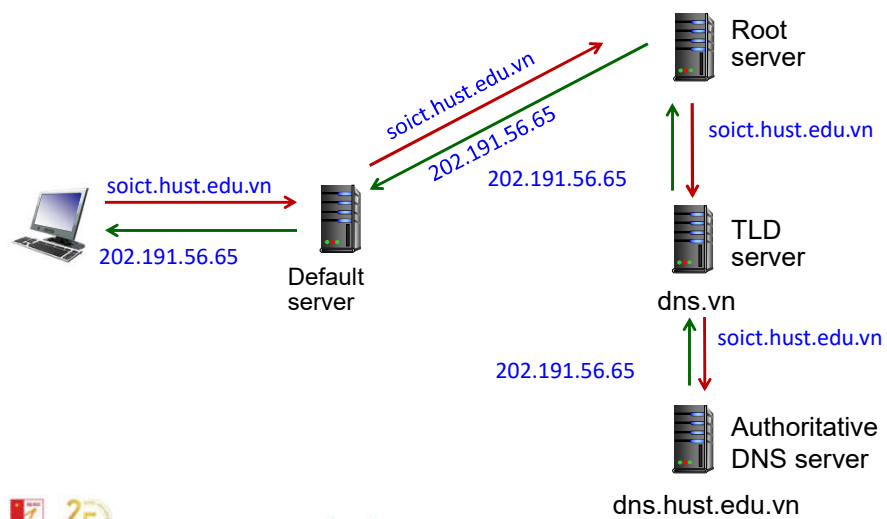
- **Recursive query**
 - Server will find the answer, even if it has to query other servers to get it
 - Server will not respond with a referral to another server
- **Iterative query**
 - If server does not have the answer, it will send a referral to another DNS server
 - Requester has to send another query to hunt for the answer

Iterative query



11

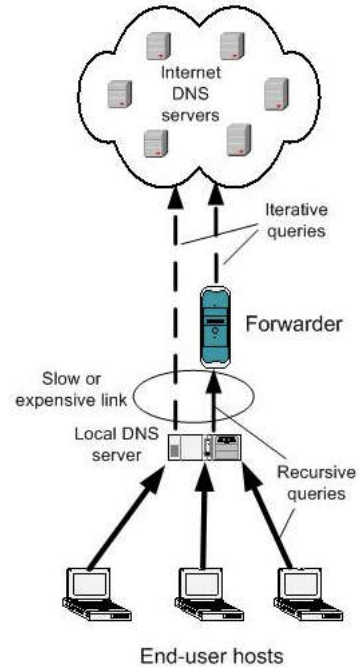
Recursive query



12

DNS Forwarder

- Only external queries are sent to the forwarder in this example
- Can reduce traffic through slow or expensive links
- Because it can cache more records



13

DNS Resolvers

- Receive requests from client applications
- Query DNS servers
- Can cache data
- **Stub resolver**
 - Resolver connected to only one recursive server
 - Cannot follow referrals
 - Part of the operating system on the end device
 - Windows stub resolver caches
 - Linux stub resolver does not cache

14

Local DNS Server

- Provided by Internet Service Provider (ISP)
- Configured at the client manually or by DHCP

```
C:\Windows\System32>ipconfig /all

Windows IP Configuration

Host Name . . . . . : W7
Primary Dns Suffix . . . . . : 
Node Type . . . . . : Hybrid
IP Routing Enabled. . . . . : No
WINS Proxy Enabled. . . . . : No
DNS Suffix Search List. . . . . : localdomain

Ethernet adapter Local Area Connection:

Connection-specific DNS Suffix . : localdomain
Description . . . . . : Intel(R) PRO/1000 MT Network Connection
Physical Address. . . . . : 00-0C-29-52-34-92
DHCP Enabled. . . . . : Yes
Autoconfiguration Enabled . . . . : Yes
IPv6 Address. . . . . : 2::4(Preferred)
Link-local IPv6 Address . . . . . : fe80::5a7:33af:ed86:b39f%11(Preferred)
IPv4 Address. . . . . : 192.168.119.219(Preferred)
Subnet Mask . . . . . : 255.255.255.0
Lease Obtained. . . . . : Sunday, September 08, 2013 12:06:16 PM
Lease Expires . . . . . : Sunday, September 08, 2013 12:36:16 PM
Default Gateway . . . . . : 192.168.119.2
DHCP Server . . . . . : 192.168.119.254
DNS Servers . . . . . : 192.168.119.2
Primary WINS Server . . . . . : 192.168.119.2
NetBIOS over Tcpip. . . . . : Enabled
```



VIỆN CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG

15

DNS Replication

- **Master server** contains primary zone files
- **Slave servers** have copies of the zone files
- **Zone transfer**
 - The process of copying the files



VIỆN CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG

16

Root Servers

- Named .
 - "dot"
- Has pointers to the top-level domains
 - **com**, **biz**, **mil**, **net**, and so on
- If a DNS server has no data in the cache, e. g. after a reboot
 - Receives a recursive query
 - Is not the SOA for that domain
 - Must query root to find the TLD servers

Root Servers

- Hundreds of servers
- Dispersed around the world
- 13 domain names
 - a.root-servers.net
 - b.root-servers.net
 - ...
 - m.root-servers.net

DNS Resource Record Types and Classes

- Each data element in DNS is called a **Resource Record (RR)**

A	IPv4 address of host
AAAA	IPv6 address of host
MX	Mail exchange
PTR	Host name corresponding to IP address
NS	Host name of SOA name server
CNAME	Canonical Name: alias
SOA	Attributes of zone
TXT	General information

Other RR Types

NAPTR	Naming Authority Pointer
SRV	Service (for specific applications)
SPF	Sender Policy Framework
	(used to control spam)
	Also included in TXT
records as a	transitional mechanism
DNSKEY, DS, RRSIG, NSEC	for
DNSSEC	

Example: dig linux.com

```
; <> DiG 9.9.2-P1 <> linux.com
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 21655
;; flags: qr rd ra; QUERY: 1, ANSWER: 2, AUTHORITY: 2,
ADDITIONAL: 3
;; QUESTION SECTION:
;linux.com. IN A
;; ANSWER SECTION:
linux.com. 1786 IN A 140.211.167.51
linux.com. 1786 IN A 140.211.167.50
;; AUTHORITY SECTION:
linux.com. 86386 IN NS ns1.linux-foundation.org.
linux.com. 86386 IN NS ns2.linux-foundation.org.
;; ADDITIONAL SECTION:
ns1.linux-foundation.org. 261 IN A 140.211.169.10
ns2.linux-foundation.org. 262 IN A 140.211.169.11
```

TTL: time to live (s) stored
in cache



VIỆN CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG

21

21

Example : dig linux.com

```
; <> DiG 9.9.2-P1 <> linux.com
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 21655
;; flags: qr rd ra; QUERY: 1, ANSWER: 2, AUTHORITY: 2,
ADDITIONAL: 3
;; QUESTION SECTION:
;linux.com. IN A
;; ANSWER SECTION:
linux.com. 1786 IN A 140.211.167.51
linux.com. 1786 IN A 140.211.167.50
;; AUTHORITY SECTION:
linux.com. 86386 IN NS ns1.linux-foundation.org.
linux.com. 86386 IN NS ns2.linux-foundation.org.
;; ADDITIONAL SECTION:
ns1.linux-foundation.org. 261 IN A 140.211.169.10
ns2.linux-foundation.org. 262 IN A 140.211.169.11
```

Delegation of authority
If ANSWER is empty, the DNS Resolver will send
the query to these DNS servers



VIỆN CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG

22

22

Example : dig linux.com

```
; <> DiG 9.9.2-P1 <> linux.com
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 21655
;; flags: qr rd ra; QUERY: 1, ANSWER: 2, AUTHORITY: 2,
ADDITIONAL: 3
;; QUESTION SECTION:
;linux.com. IN A
;; ANSWER SECTION:
linux.com. 1786 IN A 140.211.167.51
linux.com. 1786 IN A 140.211.167.50
;; AUTHORITY SECTION:
linux.com. 86386 IN NS ns1.linux-foundation.org.
linux.com. 86386 IN NS ns2.linux-foundation.org.
;; ADDITIONAL SECTION:
ns1.linux-foundation.org. 261 IN A 140.211.169.10
ns2.linux-foundation.org. 262 IN A 140.211.169.11
```

Glue records
IP address of authoritative DNS servers.



VIỆN CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG

23

23

Glue Records

- If the SOA for **hust.edu.vn** is **dns.vn** the system fails
 - Q: Where is **hust.edu.vn**?
 - A: Ask **dns.vn**
 - Q: Where is **ns.ccsf.edu**?
 - A: Ask **dns.vn**
- To prevent this, each domain has a **glue record** in their top-level domain zone specifying the IP address of the SOA



VIỆN CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG

24

Viewing Glue Records with dig

1. Find a .edu root NS server

```

. . . . . sambowne Tue Feb 10 16:57:51
~ $dig ns edu

; <<>> DiG 9.8.3-P1 <<>> ns edu
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 23660
;; flags: qr rd ra; QUERY: 1, ANSWER: 6, AUTHORITY: 0, ADDITIONAL: 0

;; QUESTION SECTION:
;edu.                                IN      NS

;; ANSWER SECTION:
edu.          170280 IN      NS      g.edu-servers.net.
edu.          170280 IN      NS      c.edu-servers.net.
edu.          170280 IN      NS      f.edu-servers.net.
edu.          170280 IN      NS      d.edu-servers.net.
edu.          170280 IN      NS      a.edu-servers.net.
edu.          170280 IN      NS      l.edu-servers.net.

```

25

IPv4 Glue for Google

• 1. NS servers for .com

```

. . . . . sambowne Tue Feb 10 16:58:37
~ $dig ns com

; <<>> DiG 9.8.3-P1 <<>> ns com
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 29
;; flags: qr rd ra; QUERY: 1, ANSWER: 13, AUTHORITY: 0, ADDITIONAL: 0

;; QUESTION SECTION:
;com.                                IN      NS

;; ANSWER SECTION:
com.          21893 IN      NS      m.gtld-servers.net.
com.          21893 IN      NS      c.gtld-servers.net.

```



VIỆN CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG

26

IPv4 Glue Records for Google

```

. . . . . sambowne Tue Feb 10 17:01:30
~ $dig NS google.com @m.gtld-servers.net

; <<>> DiG 9.8.3-P1 <<>> NS google.com @m.gtld-servers.net
;; global options: +cmd
;; Got answer:
;; -->HEADER<<-- opcode: QUERY, status: NOERROR, id: 40276
;; flags: qr rd; QUERY: 1, ANSWER: 0, AUTHORITY: 4, ADDITIONAL: 4
;; WARNING: recursion requested but not available

;; QUESTION SECTION:
;google.com.                IN      NS

;; AUTHORITY SECTION:
google.com.                 172800  IN      NS      ns2.google.com.
google.com.                 172800  IN      NS      ns1.google.com.
google.com.                 172800  IN      NS      ns3.google.com.
google.com.                 172800  IN      NS      ns4.google.com.

;; ADDITIONAL SECTION:
ns2.google.com.             172800  IN      A        216.239.34.10
ns1.google.com.             172800  IN      A        216.239.32.10
ns3.google.com.             172800  IN      A        216.239.36.10
ns4.google.com.             172800  IN      A        216.239.38.10

```

27

DNS security

- The Domain Name Service (DNS) translates human-readable names to IP addresses
 - E.g., hust.edu.vn translates to 202.191.57.199
 - DNS also provides other similar services
- It wasn't **designed** with security in mind

28

DNS Threats

- Threats to name **lookup** secrecy
 - Definition of DNS system says this data isn't secret
- Threats to DNS information **integrity**
 - Very important, since everything trusts that this translation is correct
- Threats to DNS **availability**
 - Potential to disrupt Internet service

What Could Really Go Wrong?

- DNS lookups could be **faked**
 - Meaning packets go to the wrong place
- The DNS service could be **subject** to a DoS attack
 - Or could be used to amplify one
- Attackers could “*bug*” a DNS server to learn what users are looking up

Where Does the Threat Occur?

- Unlike routing, threat can occur in several places
 - At DNS servers
 - But also at DNS clients
 - Which is almost everyone
- Core problem is that DNS responses aren't **authenticated**

31

The DNS Lookup Process

```
nslookup hust.edu.vn
```



```
ping hust.edu.vn
```

Should result in a ping packet being sent to
202.191.57.199

```
answer 202.191.57.199
```



If the answer is wrong, in standard DNS the client is *screwed*

32

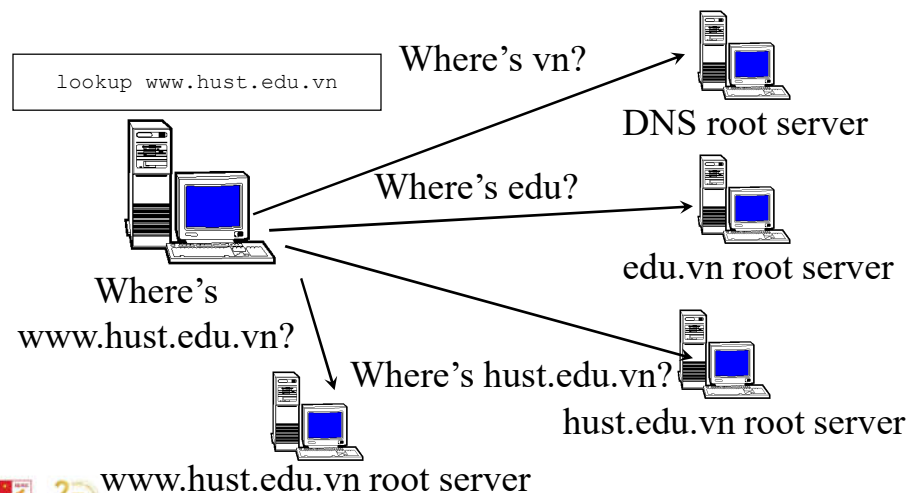
How did the DNS Server perform the Lookup?

- Ask local resolver first about name->IP mapping
 - It returns info from **cache** if any
- If info not in **cache**, resolver asks servers in DNS hierarchy that are authoritative for a given domain
 - Start from root, root sends it to the next point in hierarchy
 - End at authoritative server for the domain you are asking about (auth)
 - Resolver caches the replies



33

DNS Hierarchical Translation



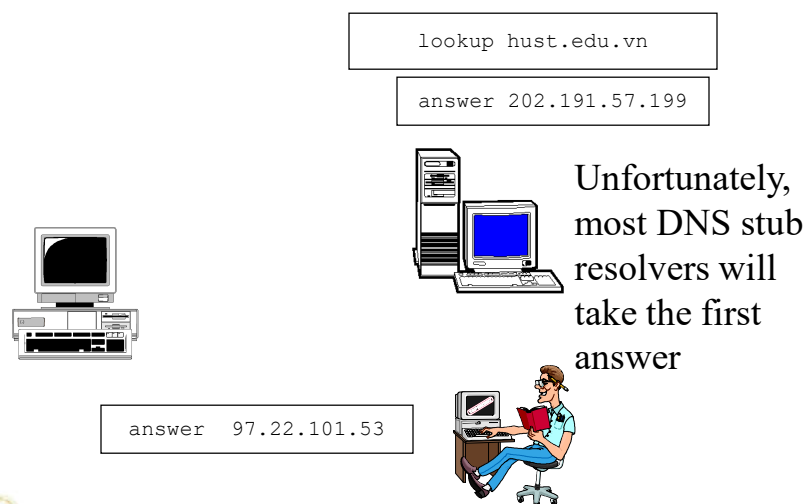
34

Where can this go wrong?

- Someone can **spoof** the answer from a DNS server
 - Relatively easy, since UDP is used
- One of the DNS servers can **lie**
- Someone can **corrupt** the database of one of the DNS servers

35

The Spoofing Problem



36

DNS hijacking

- Wait for resolver (**target**) to ask about a name from the **victim**'s domain
 - Provide your own reply **faster** than auth server
 - Provide some **extra** information (IP of auth server)
- This poisons the *cache* at that resolver, not *globally*
 - But if the resolver is at the *important/large* network the victim can lose a lot of traffic to the attacker
- Attacker's goal: impersonate the victim (phishing)



VIỆN CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG

37

Defenses

- Only accept auth if its domain is same about the domain you asked for
- Don't accept extra info in the reply if you did not ask for it



VIỆN CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG

38

Being faster is not enough

- Target will only accept replies that
 - Come to the specific port from which requests are sent
 - Come with the replyID same as the requestID
 - Attacker doesn't get the requests directly
- Can try to sniff the info
- Can try to guess the info
- If the resolver accepts the reply it will accept extra info in reply too **even if it has a different version in cache**

DNS Servers Lying

lookup hust.edu.vn



answer 97.22.101.53



...	...
...	...
...	...
...	...
...	...
hust.edu.vn	202.191.57.199
...	...
...	...
...	...
...	...

That wasn't very nice of him!

Short-circuiting waiting

- Ask the target resolver about some name from the victim's domain
 - Doesn't have to be the name you intend to hijack since DNS will take extra info in the reply
 - Asking about non-existing names guarantees they are not in resolver's cache already
- Can ask about different domains too

Hijacking with sniffing

- If on the same network as the target, attacker can try to ARP spoof the next hop
 - Position itself on the path of target's requests
 - DNS traffic uses UDP and is not encrypted, easy to see port and request ID and provide appropriate reply

Hijacking with guessing

- If target uses predictable numbers for source port and request ID attacker can perform many attacks, each time trying to guess the right port/replyID combination
 - Having randomness in one is not enough (2^{16} tries)
 - Kaminsky attack (cache poisoning 4 + no randomness in source port)

Defenses

- Randomize source port
 - Makes guessing harder but not impossible
- Use DNSSEC
 - Replies must be signed by auth
 - Everyone can check signatures
 - Auth servers for zones sign certificates when they delegate a sub-zone (e.g. .com for example.com)
 - Requires clients to implement DNSSEC to verify replies

DNS Database Corruption

lookup hust.edu.vn



answer 97.22.101.53



...	...
...	...
...	...
...	...
...	...
hust.edu.vn	97.22.101.53
...	...
...	...
...	...
...	...
...	...



VIỆN CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG

45

Security in Application layer

- DNS security
- Email security
- Web security



VIỆN CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG

46

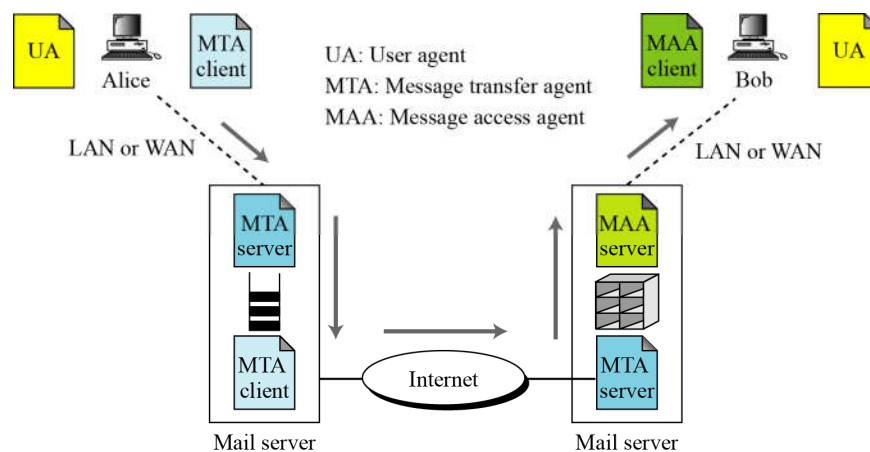
46

Why Study E-mail Security?

- After web browsing, e-mail is the most widely used network-reliant application.
- Mail servers, after web servers, are the most often attacked Internet hosts.
- Basic e-mail offers little security, counter to public perception.
- Good technical solutions are available, but not widely used.
 - If we understand why this is so, we might understand something about why security is 'hard'.

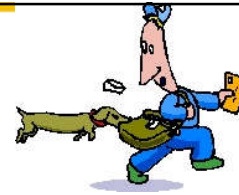
47

E-mail Architecture



48

Threats to E-mail



- Loss of confidentiality.
 - *E-mails are sent in clear over open networks.*
 - *E-mails stored on potentially insecure clients and mail servers.*
- Loss of integrity.
 - *No integrity protection on e-mails; anybody be altered in transit or on mail server.*



VIỆN CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG

49

Threats to E-mail

- Lack of data origin authentication.
 - *Is this e-mail really from the person named in the From:field?*
- Lack of non-repudiation.
 - *Can I rely and act on the content? (integrity)*
 - *If so, can the sender later deny having sent it? Who is liable if I have acted?*



VIỆN CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG

50

Threats to E-mail

- Lack of notification of receipt.
 - *Has the intended recipient received my e-mail and acted on it?*
 - *A message locally marked as 'sent' may not have been delivered.*

E-mail security

- What are the Options?
 - Secure the server to client connections (easy thing first)
 - https access to webmail
 - Protection against insecure wireless access
 - Secure the end-to-end email delivery
 - The PGPs of the world
 - Practical in an enterprise intra-network environment

E-mail security

- Email based Attacks
 - Active content attack
 - Clean up at the server
 - Buffer over-flow attack
 - Fix the code
 - Trojan Horse Attack
 - Web bugs (for tracking)
 - Mangle the image at the mail server

53

E-mail security

- Software for encrypting email messages has been widely available for more than 15 years, but the email-using public has failed to adopt secure messaging. This failure can be explained through a combination of:
 - technical,
 - community,
 - and usability factors



54

E-mail security

- **Why Don't People Use Email Security?**

- I don't because I don't care.
- I doubt any of my usual recipients would understand
- the significance of the signature.
- Never had the need to send these kinds of emails.
- I don't think it's necessary to encrypt my email.
- it's just another step & something else I don't have time



VIỆN CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG

55

E-mail security

- **Secure E-mail Standards and Products**

- PEM (privacy enhanced mail)
- X.400
- **S/MIME**
- **PGP**



VIỆN CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG

56

S/MIME

(Secure/Multipurpose Internet Mail Extension)

- Originated from RSA Data Security Inc. in 1995.
- Further development by IETF S/MIME working group at: www.ietf.org/html.charters/smime-charter.html.
- Version 3 specified in RFCs2630-2634.
- Allows flexible client-client security through encryption and signatures.
- Widely supported, e.g. in Microsoft Outlook, Netscape Messenger, Lotus Notes.



VIỆN CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG

57

Simple Mail Transfer Protocol (SMTP, RFC 822)

- **SMTP Limitations - Can not transmit, or has a problem with:**
 - executable files, or other binary files (jpeg image)
 - “national language” characters (non-ASCII)
 - messages over a certain size
 - ASCII to EBCDIC translation problems
 - lines longer than a certain length (72 to 254 characters)



VIỆN CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG

Henric Johnson

58

58

Header fields in MIME

- **MIME-Version:** Must be “1.0” -> RFC 2045, RFC 2046
- **Content-Type:** More types being added by developers (application/word)
- **Content-Transfer-Encoding:** How message has been encoded (radix-64)
- **Content-ID:** Unique identifying character string.
- **Content Description:** Needed when content is not readable text (e.g.,mpeg)

59

S/MIME Functions

- **Enveloped Data:** Encrypted content and encrypted session keys for recipients.
- **Signed Data:** Message Digest encrypted with private key of “signer.”
- **Clear-Signed Data:** Signed but not encrypted.
- **Signed and Enveloped Data:** Various orderings for encrypting and signing.

60

Algorithms Used

- **Message Digesting:** SHA-1 and MDS
- **Digital Signatures:** DSS
- **Secret-Key Encryption:** Triple-DES, RC2/40 (exportable)
- **Public-Private Key Encryption:** RSA with key sizes of 512 and 1024 bits, and Diffie-Hellman (for session keys).



VIỆN CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG

Henric Johnson

61

61

User Agent Role

- S/MIME uses Public-Key Certificates - X.509 version 3 signed by Certification Authority
- Functions:
 - **Key Generation** - Diffie-Hellman, DSS, and RSA key-pairs.
 - **Registration** - Public keys must be registered with X.509 CA.
 - **Certificate Storage** - Local (as in browser application) for different services.
 - **Signed and Enveloped Data** - Various orderings for encrypting and signing.



VIỆN CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG

Henric Johnson

62

62

PGP (Pretty Good Privacy)



- Freeware: Open PGP and variants:
 - www.openpgp.org, www.gnupg.org
- Open PGP specified in RFC 2440 and defined by IETF Open PGP working group.
 - www.ietf.org/html.charters/openpgp-charter.html
- Available as plug-in for popular e-mail clients, can also be used as stand-alone software.



VIỆN CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG

63

PGP (Pretty Good Privacy)

- *“If all the personal computers in the world—260 million—were put to work on a single PGP encrypted message, it would still take an estimated 12 million times the age of the universe, on average, to break a single message.”*



VIỆN CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG

64

What is PGP

- PGP is an e-mail security program written by Phil Zimmermann, based on the IDEA algorithm for encryption of plaintext and uses the RSA Public Key algorithm for encryption of the private key.
- PGP incorporates tools for developing a public-key trust model and public-key certificate management.



VIỆN CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG

65

PGP Services

Digital signature	DSS/SHA or RSA/SHA	A hash code of a message is created using SHA-1. This message digest is encrypted using DSS or RSA with the sender's private key and included with the message.
Message encryption	CAST or IDEA or Three-key Triple DES with Diffie-Hellman or RSA	A message is encrypted using CAST-128 or IDEA or 3DES with a one-time session key generated by the sender. The session key is encrypted using Diffie-Hellman or RSA with the recipient's public key and included with the message.



VIỆN CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG

66

PGP Services (2)

Compression	ZIP	A message may be compressed, for storage or transmission, using ZIP.
Email compatibility	Radix 64 conversion	To provide transparency for email applications, an encrypted message may be converted to an ASCII string using radix 64 conversion.
Segmentation		To accommodate maximum message size limitations, PGP performs segmentation and reassembly.

67

Fake PGP

- Since it's all open source, there are fake versions of the famous software floating about the net. Unless you're sure that your copy of the program is from a trusted source, it wouldn't be surprising to realize one day that your pass phrase was sent to an attacker via email the moment you went online! Once he has your pass phrase, he has your private key.

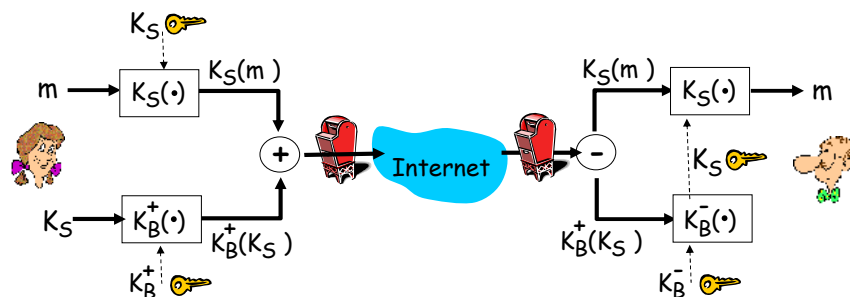
68

PGP Algorithms

- **Symmetric encryption:**
 - DES, 3DES, AES and others.
- **Public key encryption of session keys:**
 - RSA or ElGamal.
- **Hashing:**
 - SHA-1, MD-5 and others.
- **Signature:**
 - RSA, DSS, ECDSA and others.

PGP example

- Alice wants to send confidential e-mail, m , to Bob.

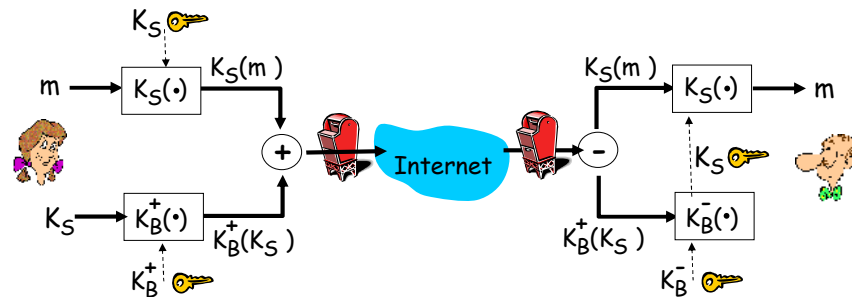


Alice:

- generates random symmetric private key, K_S .
- encrypts message with K_S (for efficiency)
- also encrypts K_S with Bob's public key.
- sends both $K_S(m)$ and $K_B^+(K_S)$ to Bob.

PGP example

- Alice wants to send confidential e-mail, m , to Bob.



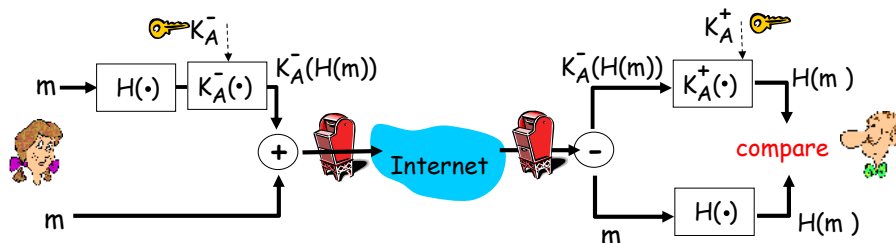
Bob:

- uses his private key to decrypt and recover K_S
- uses K_S to decrypt $K_S(m)$ to recover m

71

PGP example

- Alice wants to provide sender authentication message integrity.



- Alice digitally signs message.
- sends both message (in the clear) and digital signature.

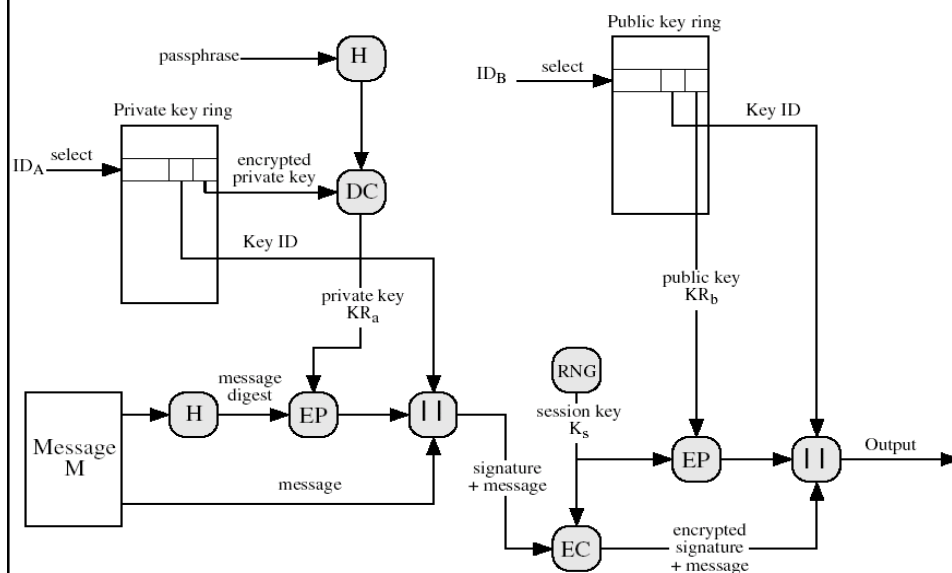
72

PGP Key Rings

- PGP supports multiple public/private keys pairs per sender/recipient.
- Keys stored locally in a *PGP Key Ring* – essentially a database of keys.
- Private keys stored in encrypted form; decryption key determined by user-entered pass-phrase.

73

PGP Message Generation



74

PGP Message Generation

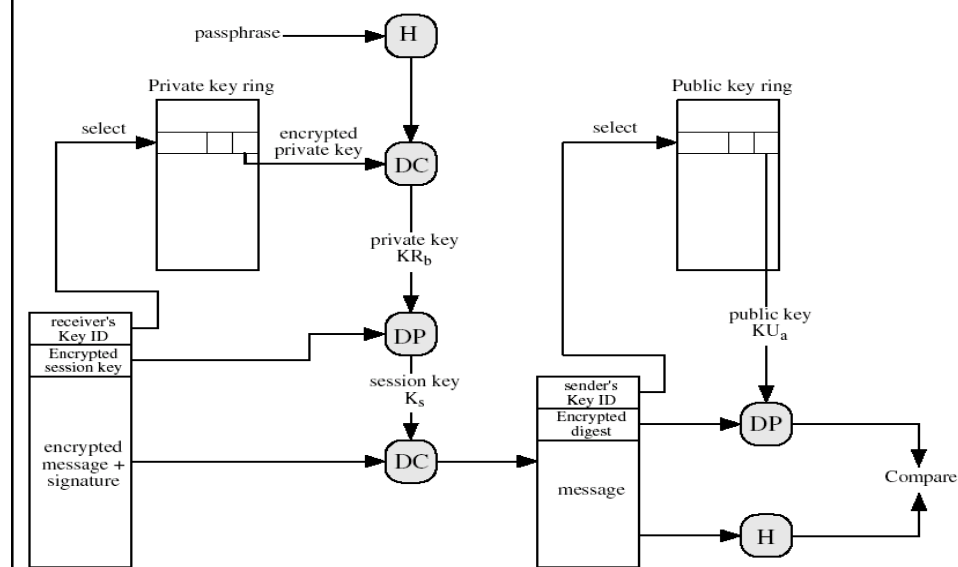
- Signs the message:
 - PGP gets sender's private key from key ring using its user id as an index.
 - PGP prompts user for passphrase to decrypt private key.
 - PGP constructs the signature component of the message.
- Encrypts the message:
 - PGP generates a session key and encrypts the message.
 - PGP retrieves the receiver public key from the key ring using its user id as an index.
 - PGP constructs session component of message



VIỆN CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG

75

PGP Message Reception



76

PGP Message Reception

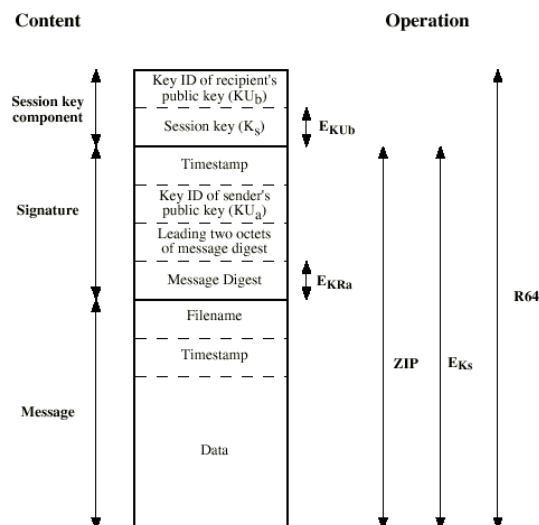
- Decrypting the message:
 - PGP get private key from private-key ring using Key ID field in session key component of message as an index.
 - PGP prompts user for passphrase to decrypt private key.
 - PGP recovers the session key and decrypts the message.
- Authenticating the message:
 - PGP retrieves the sender's public key from the public-key ring using the Key ID field in the signature key component as index.
 - PGP recovers the transmitted message digest.
 - PGP computes the message for the received message and compares it to the transmitted version for authentication.



VIỆN CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG

77

Format of PGP Message



78

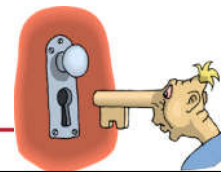
78

PGP - Key Management

- Public keys for encrypting session keys / verifying signatures.
- Private keys for decrypting session keys / creating signatures.
- Where do these keys come from and on what basis can they be trusted?



VIỆN CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG



79

PGP - Key Management

- PGP adopts a trust model called the *web of trust*.
- No centralised authority
- Individuals sign one another's public keys, these "certificates" are stored along with keys in key rings.
- PGP computes a *trust level* for each public key in key ring.
- Users interpret trust level for themselves.



VIỆN CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG

80

PGP Key Management (2)

- Trust levels for public keys dependent on:
 - Number of signatures on the key;
 - Trust level assigned to each of those signatures.
- Trust levels recomputed from time to time.

Security of PGP

- There are many known attacks against PGP.
- Attacks against cryptoalgorithms are not the main threat
- IDEA is considered strong, and while cryptoanalysis advances, it should be strong still for some time.
- RSA may or may not be strong. There are recent rumors of possible fast factorization algorithms..
- The main threats are much more simple.

PGP attack

- An attacker may socially engineer himself into a web of trust, or some trustable person may change. Then he could falsify public keys. This breaks most of the security.
- PGP binaries can be corrupted when they are obtained.
- The PGP binaries can be modified in the computer.
- The passphrase can be obtained by a Trojan. Weak passphrases can be cracked.
- On multiuser system, access to the secret key can be obtained.



83

Security in Application layer

- DNS security
- Email security
- Web security
 - Clickjacking
 - Session-Based Attacks
 - Cookie
 - Session hijacking
 - XSS



84

84

Clickjacking (UI Redressing)

[Hansen and Grossman 2008]

- Attacker overlays multiple transparent or opaque frames to trick a user into clicking on a button or link on another page



- Clicks meant for the visible page are hijacked and routed to another, invisible page



VIỆN CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG

slide
85

85

Clickjacking in the Wild

- Google search for “clickjacking” returns 624,000 results... this is not a hypothetical threat!
- Summer 2010: Facebook worm superimposes an invisible iframe over the entire page that links back to the victim's Facebook page
 - If victim is logged in, automatically recommends link to new friends as soon as the page is clicked on
- Many clickjacking attacks against Twitter
 - Users send out tweets against their will

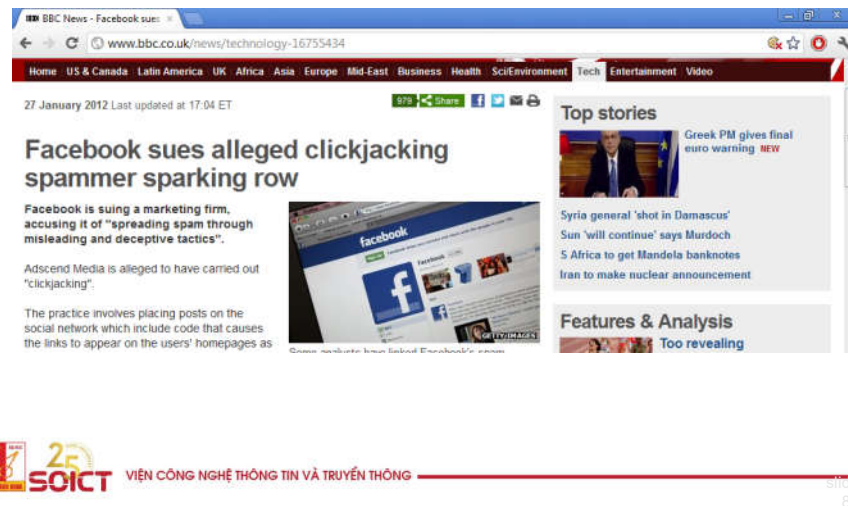


VIỆN CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG

slide
86

86

Clickjacking Meets Spamming

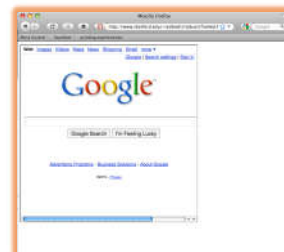


87

It's All About iFrame

- Any site can frame any other site


```
<iframe
  src="http://www.google.com/...">
</iframe>
```
- HTML attributes
 - Style
 - **Opacity** defines visibility percentage of the iframe
 - 1.0: completely visible
 - 0.0: completely invisible



88

Hiding the Target Element

- Use CSS `opacity` property ["Clickjacking: Attacks and Defenses"]
property to hide target element and make other element float under the target element
- Using CSS `pointer-events: none` property to cover other element over the target element



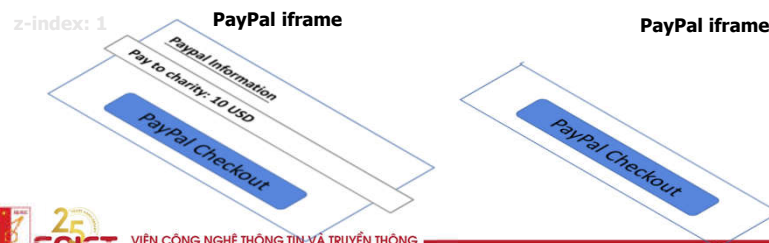
VIỆN CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG

slide
89

89

Partial Overlays and Cropping

- Overlay other elements on its own frame using CSS `z-index` property or Flash Window Mode `wmode=direct` property ["Clickjacking: Attacks and Defenses"]
- Wrap target element in a new iframe and choose CSS position offset properties



VIỆN CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG

slide
90

90

Drag-and-Drop API

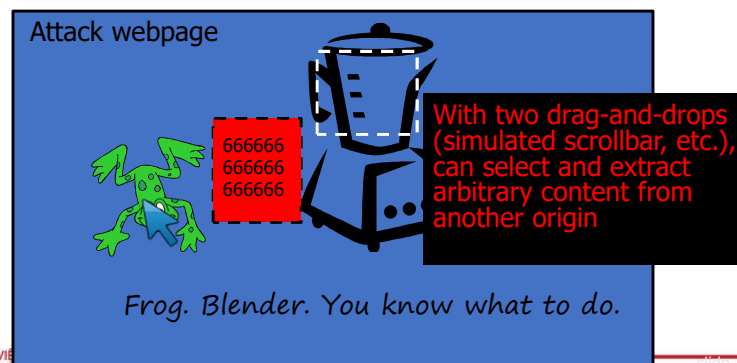
- Modern browsers support ["Next Generation Clickjacking"]
- JavaScript can use it to set data being dragged and read it when it's dropped
- Not restricted by the same origin policy: data from one origin can be dragged to a frame of another origin
 - Reason: drag-and-drop can only be initiated by user's mouse gesture, not by JavaScript on its own

91

Abusing Drag-and-Drop API

["Next Generation Clickjacking"]

1. Bait the user to click and start dragging
2. Invisible iframe with attacker's text field under mouse cursor, use API to set data being dragged
3. Invisible iframe from another origin with a form field



92

Fake Cursors

- Use CSS `cursor` property and JavaScript to simulate a fake cursor icon on the screen



VIỆN CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG

slide
93

93

Keyboard “Strokejacking”

- Simulate an input field getting focus, but actually the keyboard focus is on target element, forcing user to type some unwanted information into target element

Attacker's page

Typing Game
Type whatever screen shows to you

Xfpog95403poigr06=2kfpX

Hidden iframe within attacker's page

Bank Transfer
Bank Account: 9540
Amount: 3062 USD



VIỆN CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG

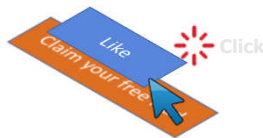
slide
94

94

Compromising Temporal Integrity

["Clickjacking: Attacks and Defenses"]

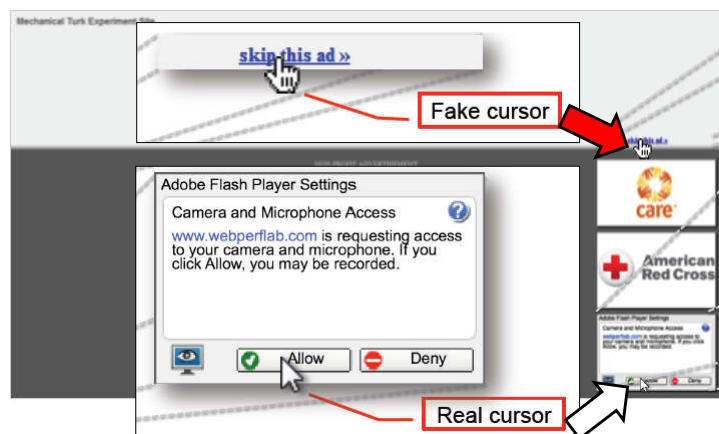
- Manipulate UI elements after the user has decided to click, but before the actual click occurs



95

Cursor Spoofing

["Clickjacking: Attacks and Defenses"]



96

Double-Click Attack

- Bait the user to perform a ^["Clickjacking: Attacks and Defenses"] double click, switch focus to a popup window under the cursor right between the two clicks



97

Whack-A-Mole Attack

- Ask the user to click as fast as possible, suddenly switch Facebook Like button



98

Solution: Frame Busting

- I am a page owner
- All I need to do is make sure that my web page is not loaded in an enclosing frame ...

Clickjacking: solved!

- Does not work for FB “Like” buttons and such, but Ok
- How hard can this be?

```
if (top != self)
  top.location.href = location.href
```



VIỆN CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG

slide
99

99

Frame Busting in the Wild

Conditional Statements

```
if (top != self)
if (top.location != self.location)
if (top.location != location)
if (parent.frames.length > 0)
if (window != top)
if (window.top !== window.self)
if (window.self != window.top)
if (parent && parent != window)
if (parent &&
    parent.frames &&
    parent.frames.length>0)
if((self.parent&&
    !(self.parent===self)) &&
    (self.parent.frames.length!=
```



slide
100

100

... Move It To Top

Counter-Action Statements
<code>top.location = self.location</code>
<code>top.location.href = document.location.href</code>
<code>top.location.href = self.location.href</code>
<code>top.location.replace(self.location)</code>
<code>top.location.href = window.location.href</code>
<code>top.location.replace(document.location)</code>
<code>top.location.href = window.location.href</code>
<code>top.location.href = "URL"</code>
<code>document.write("")</code>
<code>top.location = location</code>
<code>top.location.replace(document.location)</code>
<code>top.location.replace('URL')</code>
<code>top.location.href = document.location</code>
<code>top.location.replace(window.location.href)</code>
<code>top.location.href = location.href</code>
<code>self.parent.location = document.location</code>
<code>parent.location.href = self.document.location</code>
<code>top.location.href = self.location</code>
<code>top.location = window.location</code>
<code>top.location.replace(window.location.pathname)</code>



slide
101

101

What About My Own iFrames?

- Check: **is the enclosing frame one of my own?**
- How hard can this be?
- Survey of several hundred top websites ...
... **all** frame busting code is broken!



25
SOICT

VIỆN CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG

slide
102

102

Courtesy of Walmart

```
if (top.location != location) {
  if(document.referrer &&
    document.referrer.indexOf("walmart.com") == -1)
  {
    top.location.replace(document.location.href);
  }
}
```



103

Error in Referer Checking



From <http://www.attacker.com/walmart.com.html>
 <iframe src="http://www.walmart.com">

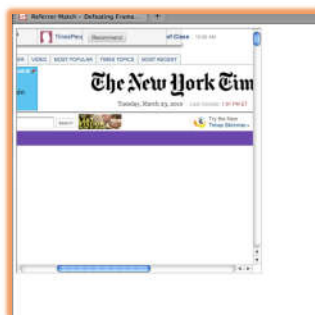
104

Courtesy of **The New York Times**

```
if (window.self !== window.top &&
    !document.referrer.match(
      /https?:\/\/(?:\w|\.)+\.nytimes\.com\/\))
{
  self.location = top.location;
}
```

105

Error in Referer Checking



From <http://www.attacker.com/a.html?b=https://www.nytimes.com/>
 <iframe src="http://www.nytimes.com">

106

Courtesy of **usbank**

```
if (self != top) {
    var domain = getDomain(document.referer);
    var okDomains = /usbank|localhost|usbnnet/;
    var matchDomain = domain.search(okDomains);

    if (matchDomain == -1) {
        // frame bust
    }
}
```

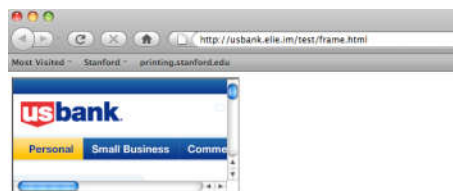


VIỆN CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG

Slide
107

107

Error in Referer Checking



From <http://usbank.attacker.com/>
<iframe src="http://www.usbank.com">



VIỆN CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG

Slide
108

108

Many Attacks on Referer Header

- Open redirect referer changer
- HTTPS->HTTP redirect changes the header
- Apparently, hard to get regular expression right
- Trust other sites to frame your pages, but what if those trusted sites can be framed themselves?



VIỆN CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG

slide
109

109

Typical Frame Busting Code

```
if(top.location != self.location) {  
    parent.location = self.location;  
}
```

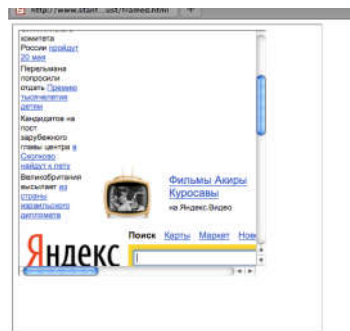


VIỆN CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG

slide
110

110

Who Is Your Daddy Parent?



Double framing!!

```
framed1.html
<iframe
src="framed2.html">
```

```
framed2.html
<iframe
src="victim.com">
```



VIỆN CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG

slide
111

111

Who Is On Top?

```
if (top.location != self.location)
    top.location = self.location
```

If **top.location** can be changed or disabled,
this code is useless



VIỆN CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG

slide
112

112

Location Clobbering

- IE 7

```
var location="clobbered";
```

- Safari

```
window.__defineSetter__("location", function(){});
```

- top.location now undefined

113

User Can Stop Frame Busting

- User can manually cancel any redirection attempt made by frame busting code
- Attacker just needs to ask...

```
<script>
```

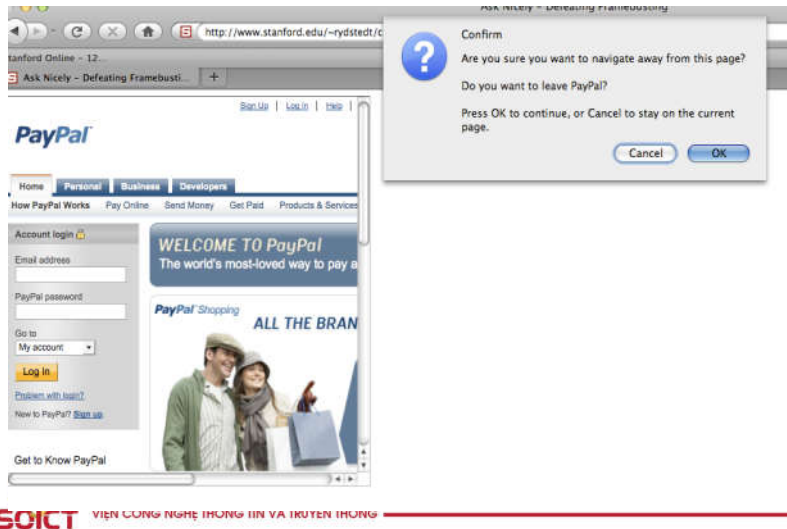
```
  window.onbeforeunload = function() {
    return "Do you want to leave PayPal?";
  }
```

```
</script>
```

```
<iframe src="http://www.paypal.com">
```

114

Ask Nicely



115

... Or Don't Even Ask

- Most browsers let **attacker cancel the relocation programmatically**

```
var prevent_bust = 0
window.onbeforeunload = function() {kill_bust++ }
setInterval(function() {
  if (kill_bust > 0) {
    kill_bust -= 2;
    window.top.location = 'http://no-content-204.com'
  }
}, 1);
```

 <iframe src="http://www.victim.com">

116

Best For Now (Still Not Good)

```
<style>html { visibility: hidden }</style>
<script>
if (self == top) {
  document.documentElement.style.visibility = 'visible';
} else {
  top.location = self.location;
}
</script>
```



VIỆN CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG

slide
117

117

These Sites Do Frame Busting

facebook

twitter

PayPal™

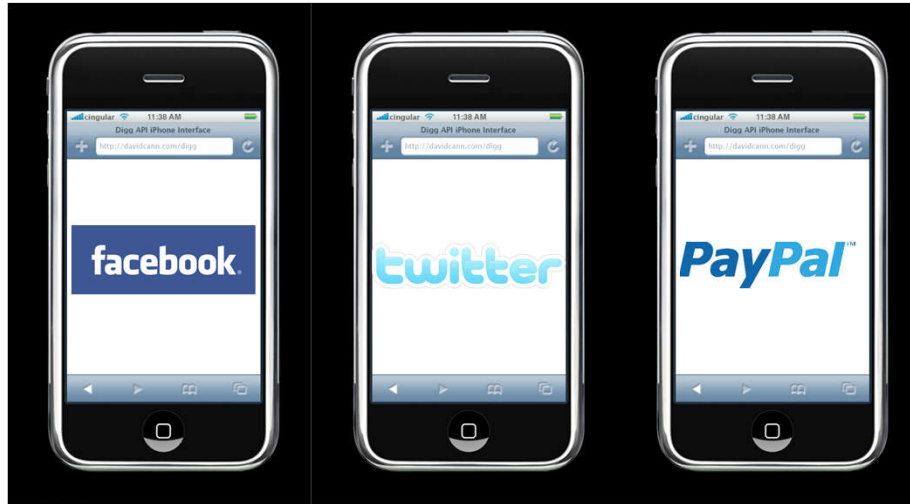


VIỆN CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG

slide
118

118

Do These?



SOICT VIỆN CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG

slide
119

119

Frame Busting on Mobile Sites

Site	URL	Framebusting
Facebook	http://m.facebook.com/	YES
MSN	http://home.mobile.msn.com/	NO
GMail	http://m.gmail.com	NO
Baidu	http://m.baidu.com	NO
Twitter	http://mobile.twitter.com	NO
MegaVideo	http://mobile.megavideo.com/	NO
Tube8	http://m.tube8.com	NO
PayPal	http://mobile.paypal.com	NO
USBank	http://mobile.usbank.com	NO
First Interstate Bank	http://firstinterstate.mobi	NO
NewEgg	http://m.newegg.com/	NO
MetaCafe	http://m.metacafe.com/	NO
RenRen	http://m.renren.com/	NO
MySpace	http://m.myspace.com	NO
Vkontakte	http://pda.vkontakte.ru/	NO
WellsFargo	https://m.wf.com/	NO
NyTimes	http://m.nytimes.com	Redirect
E-Zine Articles	http://m.ezinearticles.com	Redirect

slide
120

120

Tapjacking

- Zoom buttons in a transparent iframe so that they cover entire screen
- Hide or fake URL bar
- Make a page that masquerades as a known application to trick user into clicking

Read more:

<http://seclab.stanford.edu/websec/framebusting/>

121

SESSION-BASED ATTACK

122

Sessions

- A sequence of requests and responses from one browser to one (or more) sites
 - Session can be long (Gmail - two weeks) or short
 - without session mgmt: users would have to constantly re-authenticate
- Session mgmt:
 - Authorize user once;
 - All subsequent requests are tied to user



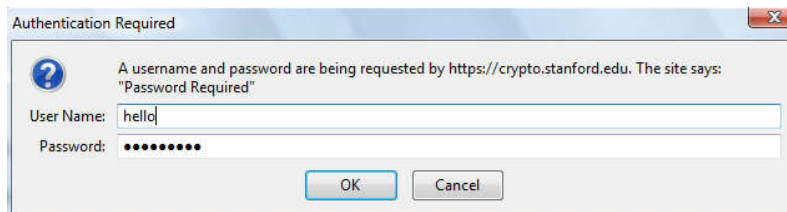
123

Pre-history: HTTP auth

HTTP request: GET /index.html

HTTP response contains:

WWW-Authenticate: Basic realm="Password Required"



Browsers sends hashed password on all subsequent HTTP requests:

Authorization: Basic ZGFddfibzsdgkjheczI1NXRleHQ=



124

HTTP auth problems

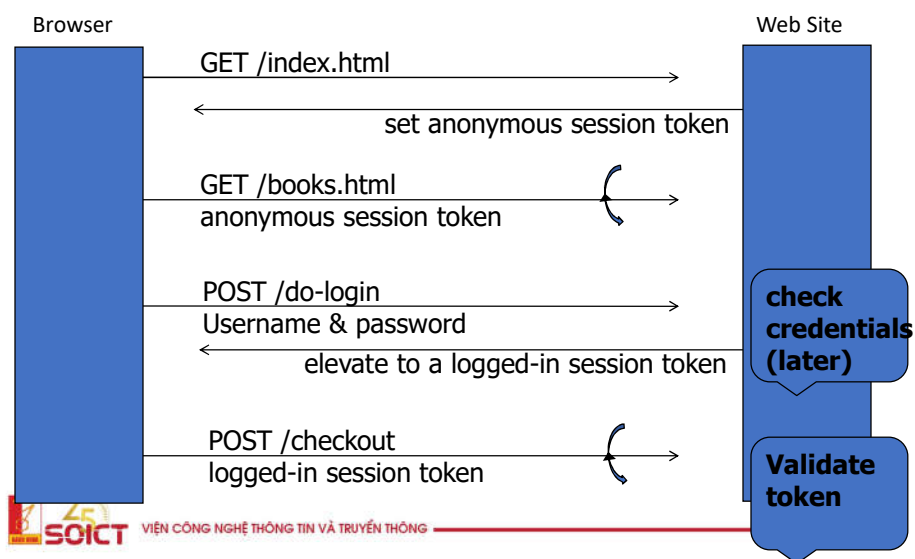
- Hardly used in commercial sites
 - User cannot log out other than by closing browser
 - What if user has multiple accounts?
 - What if multiple users on same computer?
 - Site cannot customize password dialog
 - Confusing dialog to users
 - Easily spoofed
 - Defeated using a TRACE HTTP request (on old browsers)



VIỆN CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG

125

Session tokens



VIỆN CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG

126

Storing session tokens:

Lots of options (but none are perfect)

- Browser cookie:
Set-Cookie: SessionToken=fduhye63sfdb
- Embedd in all URL links:
`https://site.com/checkout ? SessionToken=kh7y3b`
- In a hidden form field:
`<input type="hidden" name="sessionid" value="kh7y3b">`
- Window.name DOM property



VIỆN CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG

127

Storing session tokens: problems

- Browser cookie:
browser sends cookie with every request, even when it should not (CSRF)
- Embed in all URL links:
token leaks via HTTP Referer header
- In a hidden form field: short sessions only

Best answer: a combination of all of the above.



VIỆN CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG

128

The HTTP referer header

GET /wiki/John_Ousterhout HTTP/1.1

Host: en.wikipedia.org

Keep-Alive: 300

Connection: keep-alive

Referer: <http://www.google.com/search?q=john+ousterhout&ie=utf-8&oe>

Referer leaks URL session token to 3rd parties



VIỆN CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG

129

Session-Based Attacks

- HTTP is **connectionless**
 - But many/most apps want to maintain **state**
 - Using IP addresses is an **imperfect** solution
 - Why?
 - **Cookies** were invented to solve exactly this problem



VIỆN CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG

130

Using Cookies for Session Management

- The server wants to maintain information about the current client
 - **Encode** state into an alphanumeric string
 - Use `Set-cookie=string` and send to browser
 - Optionally set `expires`, `domain`, `path`, `secure`, `httponly` values as well
 - Now each time the browser wishes to connect to a given domain and path, it checks its cookie store and transmits all matching cookies



VIỆN CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG

131

Using Cookies for Session Management

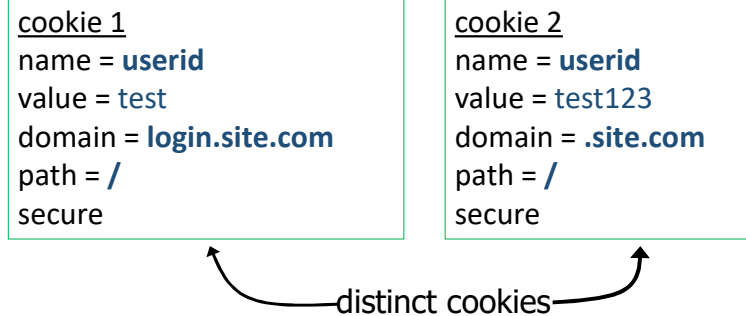
- They are essentially a **temporary password**
 - Difficult to guess, not short enough to brute-force, unique
 - These are often violated by using insufficient randomness, being too short, using counters, etc
- Many apps that lock-out password attempts fail to guard against brute force attacks on session ids
 - So short session ids are very vulnerable



VIỆN CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG

132

Cookies are identified by (name,domain,path)



- Both cookies stored in browser's cookie jar;

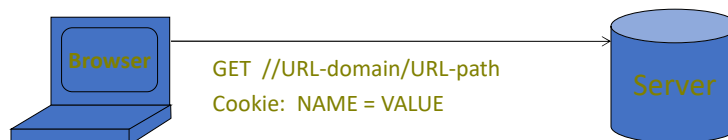
both are in scope of **login.site.com**



VIỆN CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG

133

Reading cookies on server (read SOP)



Browser sends all cookies in URL scope:

- cookie-domain is domain-suffix of URL-domain, and
- cookie-path is prefix of URL-path, and
- [protocol=HTTPS if cookie is "secure"]

Goal: server only sees cookies in its scope



VIỆN CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG

134

Client side read/write: **document.cookie**

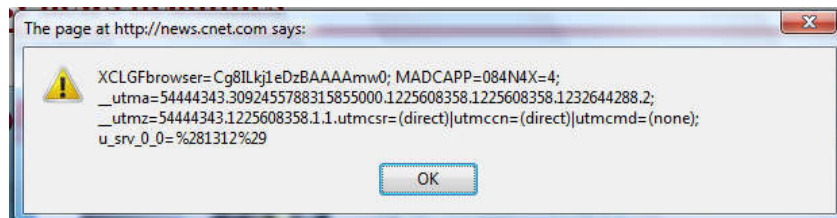
- Setting a cookie in Javascript:
`document.cookie = "name=value; expires=...; "`
- Reading a cookie: `alert(document.cookie)`
 prints string containing all cookies available
 for document (based on [protocol], domain, path)
- Deleting a cookie:
`document.cookie = "name=; expires= Thu, 01-Jan-70"`



~~document.cookie often used to customize page in Javascript~~

135

Javascript URL
 javascript: alert(**document.cookie**)



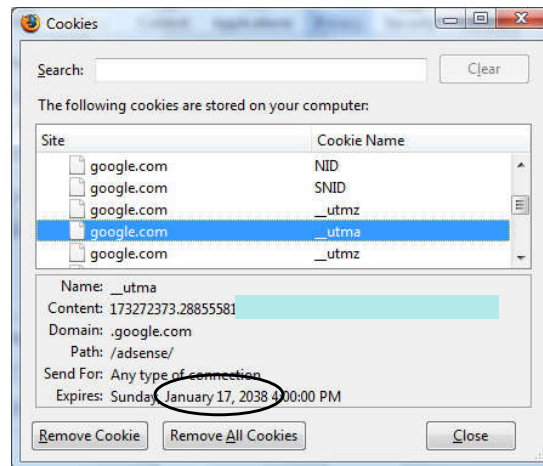
Displays all cookies for current document



VIỆN CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG

136

Viewing/deleting cookies in Browser UI



137

Example 1: login server problems

- Alice logs in at **login.site.com**
login.site.com sets session-id cookie for **.site.com**
- Alice visits **evil.site.com**
overwrites **.site.com** session-id cookie
with session-id of user “badguy”
- Alice visits **IT4262E.site.com** to submit homework.
IT4262E.site.com thinks it is talking to “badguy”

Problem: IT4262E expects session-id from login.site.com; cannot tell that session-id cookie was overwritten

138

Example 2: “secure” cookies are not secure

- Alice logs in at

```
Set-Cookie: LSID=EXPIRED;Domain=.google.com;Path=/;Expires=Mon, 01-Jan-1990 00:00:00 GMT
Set-Cookie: LSID=EXPIRED;Path=/;Expires=Mon, 01-Jan-1990 00:00:00 GMT
Set-Cookie: LSID=EXPIRED;Domain=www.google.com;Path=/accounts;Expires=Mon, 01-Jan-1990 00:00:00 GMT
Set-Cookie: LSID=cl:DQAAHsAAACn3h7GCpKUNxckr79Ce3BUC/tluaI9a7e5oPvByTr
Set-Cookie: GAUSR=dabo123@gmail.com;Path=/accounts;Secure
```

- Alice visits <http://www.google.com> (cleartext)
 - Network attacker can inject into response
Set-Cookie: LSID=badguy; secure
and overwrite secure cookie

- Problem: network attacker can re-write HTTPS cookies !



VIỆN CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG

⇒ HTTPS cookie value cannot be trusted

139

Interaction with the DOM SOP

Cookie SOP: path separation

[x.com/A](#) does not see cookies of [x.com/B](#)

Not a security measure:

DOM SOP: [x.com/A](#) has access to DOM of [x.com/B](#)

```
<iframe src="x.com/B"></iframe>
alert(frames[0].document.cookie);
```

Path separation is done for efficiency not security:

[x.com/A](#) is only sent the cookies it needs



VIỆN CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG

140

Cookie Integrity?

– User can change and delete cookie values !!

- Edit cookie file (FF3: cookies.sqlite)
- Modify Cookie header (FF: TamperData extension)

– Silly example: shopping cart software

Set-cookie: shopping-cart-total = 150 (\$)

– User edits cookie file (cookie poisoning):

Cookie: shopping-cart-total = 15 (\$)

Similar to problem with hidden fields

<INPUT TYPE="hidden" NAME=price VALUE="150">



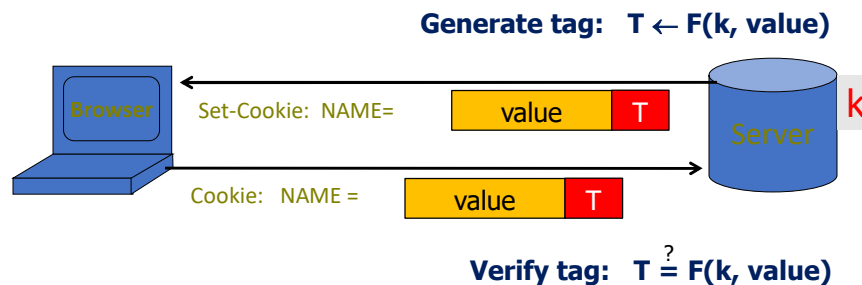
VIỆN CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG

141

Solution: cryptographic checksums

Goal: data integrity

Requires secret key k unknown to browser



"value" should also contain data to prevent cookie replay and swap

142

Example: .NET 2.0

– `System.Web.Configuration.MachineKey`

- Secret web server key intended for cookie protection
- Stored on all web servers in site

Creating an encrypted cookie with integrity:

```
– HttpCookie cookie = new HttpCookie(name, val);
  HttpCookie encodedCookie =
    HttpSecureCookie.Encode (cookie);
```

Decrypting and validating an encrypted cookie:

```
– HttpSecureCookie.Decode (cookie);
```



VIỆN CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG

143

SESSION HIJACKING

Attacker waits for user to login;

then attacker obtains user's Session Token
and "hijacks" session



VIỆN CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG

144

1. Predictable tokens

- Example: counter (Verizon Wireless)
 - ⇒ user logs in, gets counter value, can view sessions of other users
- Example: weak MAC (WSJ)
 - token = {userid, $MAC_k(\text{userid})$ }
 - Weak MAC exposes k from few cookies.

Session tokens must be unpredictable to attacker:
Use underlying framework.

Rails: token = MD5(current time, random nonce)

145

2. Cookie theft

- Example 1: login over SSL, but subsequent HTTP
 - What happens at wireless Café ?
 - Other reasons why session token sent in the clear:
 - HTTPS/HTTP mixed content pages at site
 - Man-in-the-middle attacks on SSL
- Example 2: Cross Site Scripting (XSS) exploits
- Amplified by poor logout procedures:
 - Logout must invalidate token on server



25

VIỆN CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG

146

Session fixation attacks

- Suppose attacker can set the user's session token:
 - For URL tokens, trick user into clicking on URL
 - For cookie tokens, set using XSS exploits
- Attack: (say, using URL tokens)
 1. Attacker gets anonymous session token for site.com
 2. Sends URL to user with attacker's session token
 3. User clicks on URL and logs into site.com
 - this elevates attacker's token to logged-in token
 4. Attacker uses elevated token to hijack user's session.



VIỆN CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG

147

Session fixation: lesson

- When elevating user from anonymous to logged-in, always issue a new session token
- Once user logs in, token changes to value unknown to attacker.
 - ⇒ Attacker's token is not elevated.



VIỆN CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG

148

Session token 1: minimal client-side state

1. SessionToken = [random unpredictable string]
(no data embedded in token)
 2. Server stores all data associated to SessionToken:
userid, login-status, login-time, etc.
- Can result in server overhead:
 - When multiple web servers at site,
lots of database lookups to retrieve user state.



VIỆN CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG

149

SessionToken 2: lots of client-side state

SID = [userID, exp. time, data]

where data = (capabilities, user data, ...)

SessionToken = Enc-then-MAC (k, SID)

k: key known to all web servers in site.

- ◆ Server must still maintain some user state:
 - e.g. logout status (should be checked on every request)
- Note that nothing binds SID to client's machine



VIỆN CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG

150

Binding SessionToken to client's computer;
mitigating cookie theft

approach: embed machine specific data in SID

- **Client IP Address:**

- Will make it harder to use token at another machine
- But honest client may change IP addr during session
 - client will be logged out for no reason.

- **SSL session key:**

- Same problem as IP address (and even worse)

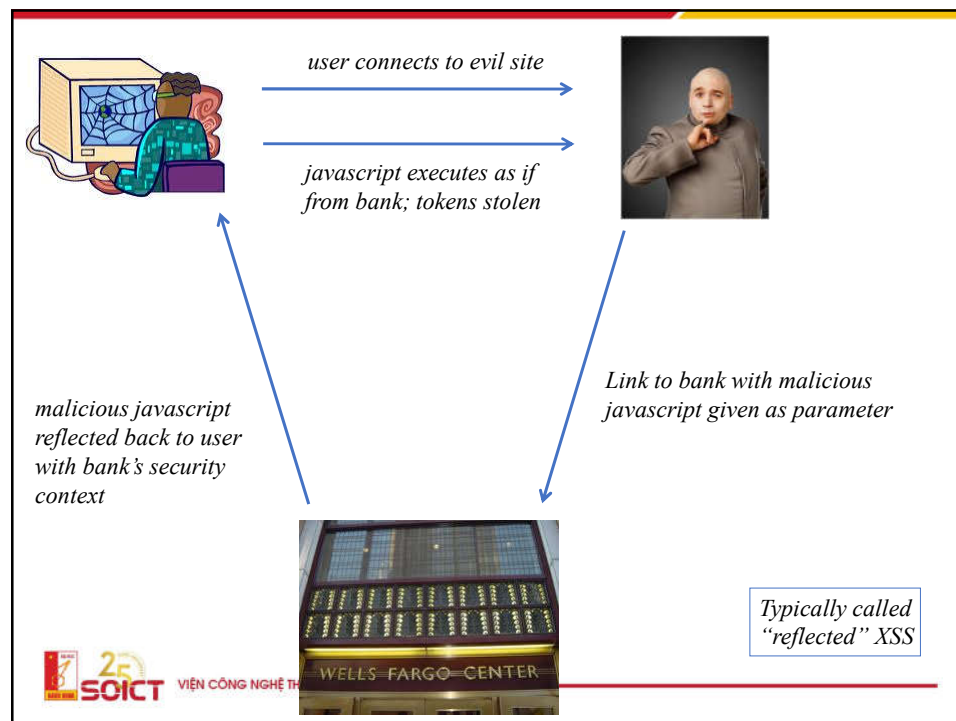
Cross-Site Scripting (XSS)

- XSS is a very common vulnerability
 - Would be vulnerability of the decade except SQL injections are often far more serious
 - XSS is used for a client to attack another client, not to attack a server
- An XSS vulnerability is as simple as echoing back user-input without sanitizing
 - Ex: You submit: "XYZ!! (2" to a search engine and it replies with "XYZ!! (2 no results found"

XSS

- The idea of XSS is for an attacker to inject malicious javascript into a security context that it does not own
 - And, as we know, this means things like session tokens can be sent anywhere we like

153



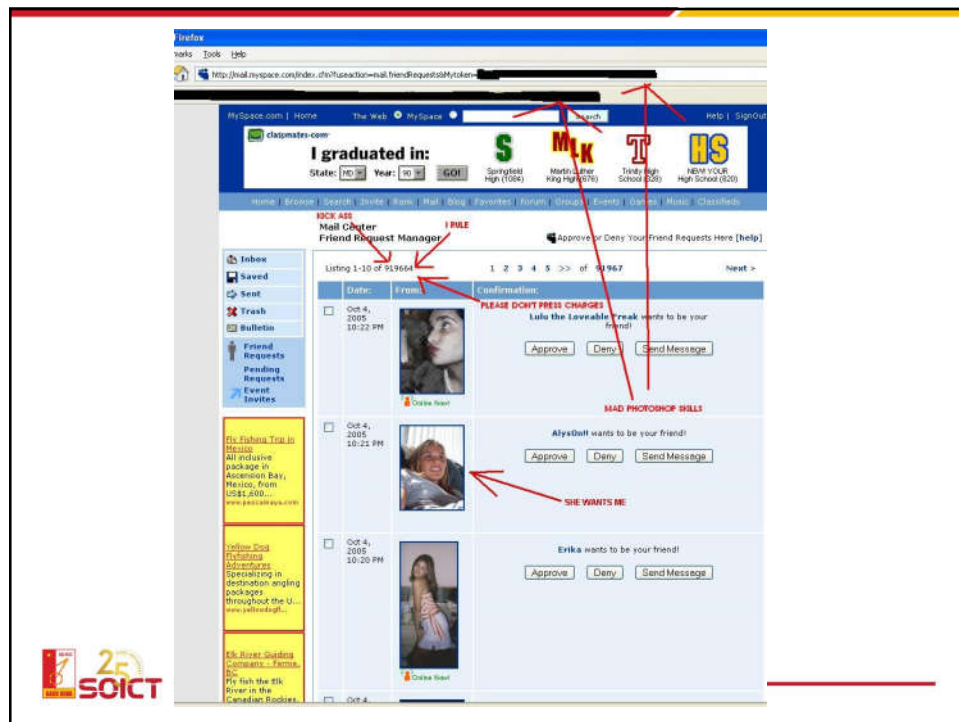
154

Stored XSS

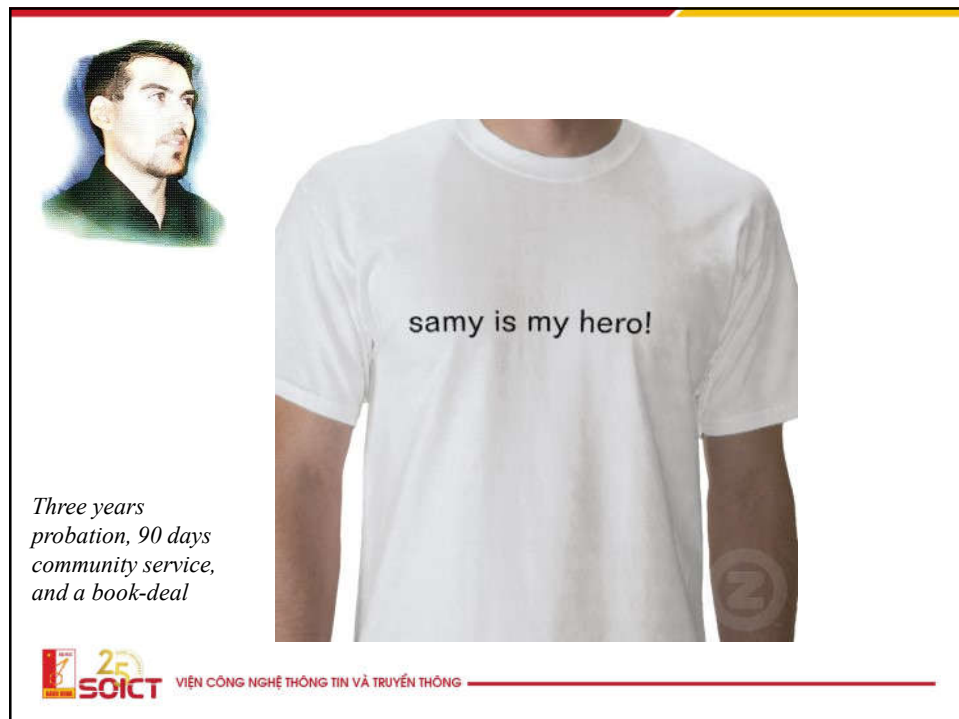
- Stored XSS is very similar
 - Instead of using a reflection bug, the attacker **stores** javascript in a place where the victim is likely to read it (and thereby execute it)
 - It's usually the server's responsibility to sanitize user input before storing it
 - Consider a public forum where various users post their thoughts
 - And their exploit code...
 - Stored XSS is usually considered **more** serious
 - No need to induce the user to establish a session then visit drevil's site, which can be hard some times

Samy XSS worm

- Oct, 2005: myspace had an XSS vulnerability
 - They used <script> filters, but 19-yr old Samy Kamkar found a way to bypass all of them
 - He built an AJAX app so that every view of his profile added him as a friend and posted "...and most of all, Samy is my hero" to their page
 - He also had the worm install itself so that any viewer of the page would propagate the worm



157



158

Finding XSS Vulnerabilities

- Try entering

`><script>alert(document.cookie)</script>` in every user-input vector and monitor for appearance of this string from the site

- If string comes back unmodified, jackpot
 - This is automatable
 - Some XSS vulns will not be found by this technique, however, since `<script>` is often filtered out (as are `<` `>` `/`)
 - Some XSS filters will miss `><script>`, `><ScRiPt>`, `%3e%3cscript%3e`, `><scr<script>ipt>`, `%00><script>`

Good Practices (for security)

- Create a session token on **first visit**
- When performing authentication, **destroy** the old session and **create** a new one
- **Expire** sessions after a **short** period (30-60 mins)
- **Destroy** sessions after **logout**
- Use **SSL** and mark session cookies as **secure**
- Monitor `User-Agent` header; it shouldn't change during a session



161