Web Security

Mahalingam Ramkumar
Web Security

- Abusing cookies
- Phishing, Spreading misinformation
  - You are not seeing what you think you are!
    - HREFs, Ill-constructed strings in HREFs
    - Dynamic HTML, Scripting (giving attacker more control of the browser)
- Countermeasures
  - Securing DNS
  - Transport layer security
  - Secure Electronic Transactions
Cookies

- HTTP is a state-less protocols
- How do we move from connections to “sessions”?
- Designed by Netscape
- Cookie fields
  - Domain Name
  - Path
  - Content
  - Expiry
  - Secure
Structure of Cookies

- Content field – a series of name=value
- Secure field – cookie will be returned only if the server is secure (uses SSL)
- Expiry – Persistent and Non-persistent cookies
Cookies

- Cookies are created by web servers
- Stored in client machine (usually in some directory under the home directory)
- When a client (browser) connects to a server it checks for cookies for the server domain
- If a cookie is found, it sends the cookie to the server along with the connection request
Applications

- Banking
- Gaming sites
- Shopping carts
- News sites (setting user preferences)
- Web portals
- Just about every thing....
- Is it possible to do all this without cookies?
Cookie Abuse

- Tracking user habits
- Advertising agencies buy ad-space for big corporations on pages in major websites.
- Let us say an agency buys ads in N pages.
- In each page they add a link to some image for the banner ad –
  - Page 1 will have link adagency.com/image1.gif
  - Page N has link adagency.com/imageN.gif
  - (Every page has a unique link)
Profiling User Habits

- User starts with a “clean slate”
- User visits page i
- Sends http request to adagency.com for image_i.gif
- Adagency server sends back a cookie with a random but unique number
  - This will serve to distinguish the users
- Now user is associated with a unique cookie
- Every time the user goes to one of the N sites the cookie is sent to adagency.com
Tracking User Habits

- Adagency.com gets paid for placing ads
- On top of that they “sell” collected user information
- Cookies can be blocked
- Many websites won’t “work” though
- Fine grained control of cookies- blocking secondary cookies
  - Started with Mozilla
Spreading Misinformation

- Spoof corporate websites/news sites
- Emulex Corporation
  - Lost 2 billion dollars due to a fake email message sent to a news agency
  - Perpetrator made a million dollars
- Manipulating HREFs, Scripting
- DNS spoofing
Domain Name System

- Hierarchical name space
- Root at the top of the hierarchy
- Generic top-level-domains (gTLD) and country-code domains (ccTLDs) at the next step
- disney.cse.msstate.edu is a name of a service that can be reached
- Under the branch cse.msstate.edu, which is under the branch msstate.edu, under the thicker branch .edu
Domain Name System Zones

- A branch (including sub-branches and leaves) is a zone
- The whole DNS tree is under the root zone
- msstate.edu is a zone
- Every zone has an administrative authority (zone authority)
- Zone authority creates DNS records describing details about various services / computers in the zone
Authoritative Name Servers

- All DNS resource records (RR) pertaining to a zone are put together in a master file.
- The master file is provided to DNS servers authoritative for the zone.
- DNS servers can be queried for DNS records by resolvers.
DNS Resource Records

- Each RR is a five-tuple
  - Name, class, TTL, type, value
  - Class is always IN; TTL is the number of seconds for which an RR can be cached
  - Types: A (IP address), NS (name of an authoritative name server), MX (mail server), CNAME etc.
DNS Query and Response

- Queries and responses have the same format.
- Payloads of UDP packets
- Header (options, transaction ID)
- Four sections: QUESTION, ANSWER, AUTHORITY, ADDITIONAL
Typical Query Response Process

- Browser desires IP address of msnbc.com
- Call gethostbyname(msnbc.com)
- Call handled by a stub-resolver in the same machine
- Stub-resolver sends a DNS query (msnbc.com, A) to a local DNS server
  - Typically operated by ISP
  - IP addresses of LDNSs in /etc/resolv.conf
Typical Query Response Process

- LDNS send query (msnbc.com, A) to root DNS server (which is authoritative for the root zone)
- Root server responds with the name and address of a DNS server authoritative for .com
  - NS record for the name .com, and an A type record corresponding to the name specified in NS record
- LDNS sends the same query to the ANS for .com
  - NS record for msnbc.com and an A type record
Typical Query Response Process

- LDNS sends the same query to the ANS for .com
- The response
  - NS record for msnbc.com
    - msnbc.com 172727 IN NS ns1.msft.net
  - and an A type record
    - ns1.msft.net 172727 IN A 207.68.160.90
Typical Query Response Process

- LDNS sends the query to the ANS for msnbc.com
- The response
  - A type record for msnbc.com
    - msnbc.com 57 IN A 207.46.245.61
- LDNS sends back response to stub-resolver
- LDNSs cache records for a duration specified by the TTL
DNS Response

• QUESTION: msnbc.com A
• ANSWER: A type record for msnbc.com
• AUTHORITY: NS record for msnbc.com
• ADDITIONAL: A type record for the ANS (ns1.msft.net)
• Typically, multiple records may exist (in the master file) for the same name and type
• A set of records for the same name and type is called an RRSet
• The answer is typically an RRSet
DNS Spoofing

- Alice desires IP address of mybank.com
- Alice's stub resolver gets back the required information
- But where is the proof that
  - The DNS response came from the LDNS
  - The DNS record was created by the authority for msnbc.com
  - ns1.msft.net is indeed the ANS for the zone
DNS Spoofing

- How difficult is it for an attacker to send a misleading response to the query sent by Alice?
  - Directing Alice to computer under his control
  - And phishing personal information
- Depends on where the attacker is located
  - Internal attacker – same LAN as Alice or the LDNS or in-between them
  - External attacker – all other attacker's
DNS Cache Poisoning

- The attacker tries to give LDNS an incorrect IP address for “mybank.com”
  - Need to make the LDNS send a query for mybank.com
  - Need to know the 16-bit transaction ID used by the LDNS in the query for (mybank.com, A)
  - Need to know the UDP port number used by the LDNS to send the query
  - With all this information the attacker can send a “response” which preempts the actual response
DNS Cache Poisoning

- Not much of a challenge for an internal attacker
  - Send a query for msnbc.com to the LDNS
  - Overhear the packet sent by the LDNS
    - To find transaction ID and port number
  - Send a “response” to the LDNS
  - The real response will be dropped
DNS Cache Poisoning

- **External attacker**
  - Register a zone – say somename.com
  - Run a DNS server authoritative for somename.com
  - Send a query to the target LDNS for (a1.somename.com, A)
  - The LDNS will ultimately be directed to the attacker's DNS server
  - Attacker gets to know the transaction ID and UDP port number
  - Attacker can keep in step by sending another query (say for a2.someone.com)
DNS Cache Poisoning

• Why does this work?
  • Most DNS resolvers increase the transaction IDs sequentially
  • Most DNS resolvers use a small set of UDP port numbers for sending DNS queries
  • With the knowledge of transaction ID and port number the attacker can send a query for “mybank.com”
    - And immediately follow it up with a response
    - A few responses in practice (with different transaction IDs / UDP port numbers)
DNS Cache Poisoning

- Not just “mybank.com”
  - Can spoof as many names as the desires!
- Kaminsky attack
  - Why not spoof the response from the root server (which provides IP addresses of gTLD servers)?
  - If the attacker can send an incorrect IP address for .com all queries for .com will be sent to the attacker
  - The attacker himself provides information about ANSs for different zones!
Mitigating Attacks

- Use random transaction IDs
- Use random port numbers for sending DNS queries
- After the recent discovery by Kaminsky most DNS servers have been patched
- But still does not prevent internal attackers :(
- Authentication
  - Can prevent attacks by internal and external attackers
DNSSEC

- RFCs 4033 – 4035, 5155 (NSEC3), 2535 (obsolete)
- Zone authorities digitally sign every RRSet
- Signatures are inserted in the master file as type RRSIG records
- An RRSet in the ANSWER section will be accompanied by an RRSIG record
- Verifying the signature requires the public key of the signer
- Inserted as DNSKEY record
- How do we know the public key is authentic?
- A delegation signer (DS) record is provided by a higher authority
- Which is accompanied by a signature record RRSIG(DS)
DNSSEC Example

- Query (disney.cse.msstate.edu, A)
- The ANS returns
  - A type RRSet for the name
  - RRSIG record
- If the public key of the zone authority msstate.edu is not in cache
  - Fetch DNSKEY record
  - Fetch DS record and RRSIG (DS) record from a .org DNS server
- If public key of .org server is not in the cache
  - Fetch DNSKEY record of .org
  - Fetch DS and RRSIG(DS) from the root
NSEC

- Authenticated denial of existence
- Why
  - a.com and b.com are competitors
  - Both of them use ns1.dnsservices.net as their ANS
  - a.com bribes someone in dnsservices.net to respond to all queries from b.com with “record not found”
- Zone authorities should not need to trust DNS operators
- DNS servers should provide verifiable proof that a record for b.com does not exist
Authenticated Denial

- NSEC records corresponding to every name
- Arranged in a dictionary order of names
- Records created and signed by zone authorities
- For example a NSEC record
  - `msstate.edu   NSEC [A,NS, MX], cad.msstate.edu`
  - And an accompanying RRSIG(NSEC)
  - Shows that `cad.msstate.edu` is the name that follows `msstate.edu` (in the dictionary order)
  - Is proof that
    - Records for names like `a.msstate.edu`, `b.msstate.edu`, `cac.msstate.edu` do not exist
    - No CNAME exists for `msstate.edu`
DNS-Walk

- NSEC makes DNS susceptible to DNS-walk
- DNS records are meant to be public
- However, a record for cad.msstate.edu should be provided only if some one explicitly queries for the name
- With NSEC an attacker can query a random name and get to know of some records
  - For example a query for aa.msstate.edu reveals the existence of several records for the name msstate.edu and a record for cad.msstate.edu
DNS-Walk

- With a second query (for say cae.msstate.edu) the attacker gets a NSEC record giving details of cad.msstate.edu and perhaps cse.msstate.edu
- Attacker can walk through all records in a master file
- Useful for launching attacks (simplifies portscanning!)
DNSSEC Iterations

- Earliest version had SIG, KEY and NXT records
- Left the problem of “who is the root authority?” open
- 2005 version: RRSIG, DNSKEY, DS, NSEC
- NXT did not have types indicated
- Very soon NSEC will be replaced by NSEC3 to address the DNS-Walk problem
NSEC3

- Hashed authenticated denial of existence
- To make DNS walk difficult
- Corresponding to every name in the master file is a hash of the name
- The hashes are arranged in an order
- An NSEC3 records indicate two adjacent hashes
- Signed using RRSIG(NSEC3)
NSEC3

- Say a query for a name a.b.abc.com is received by ANS for abc.com
- Say hash of a.b.abc.com is 56789
- A NSEC3 record with hashes (55457, 58990) is proof that no record exists for the name a.b.abc.com
- Not so simple
  - Also has to prove
    - No delegation (NS record) exists for b.abc.com
    - No wild-card records of the form *.b.abc.com exists
    - Typically multiple NSEC3 records (and their RRSIG records have to be returned)
DNSSEC Adoption

- Has seen very low adoption rates
- May improve with a recent mandate (after discovery of the recent Kaminsky attack)
- Very good reasons as to why the adoption has been very slow
DNSSEC Issues

- Each signature / key takes hundreds of bytes
- Several signatures keys required for verification
- Cannot fit the entire response in a UDP packet (plain DNS restricts UDP size to 512 bytes)
- Resort to TCP - obviously expensive
- Cache memory sizes for DNS servers increases 5 – 10 fold (again due to RRSIG and KEY records)
DNSSEC Issues

- Implies one will have to upgrade hardware to migrate from plain DNS to DNSSEC
- On top of all this servers which did deploy DNSSEC saw dramatic increase in query volume (DNS walk!)
- NSEC3 does not eliminate DNS-Walk.
  - Dictionary attacks can be done to find preimages of hash enclosers
  - Just makes DNS walk more difficult
- SSL/TLS widely used
  - So?
SSL

- Secure Socket Layer
- Purpose
  - Reliable end-to-end secure service
  - Provides a “secure TCP socket”
  - Usually used with Web browsers
    - Can be used for other applications too
  - Introduced by Netscape in 1995
  - Provided options for 40 and 128 bit keys
    - 40 bits for export
  - Submitted to IETF standards
    - Result – TLS (RFC 2246)
SSL in the Protocol Stack

<table>
<thead>
<tr>
<th>Physical Layer</th>
<th>Data Link</th>
<th>Network (IP)</th>
<th>Transport (TCP)</th>
<th>Security (SSL)</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>Application</th>
<th>HP</th>
<th>CSP</th>
<th>AP</th>
<th>HTTP</th>
</tr>
</thead>
</table>
Broad Overview

- **Perform Handshake**
  - Exchange of nonces, preferences
  - Exchange of certificates, certificate chains
  - Establishment of shared key
  - Finish handshake
- **Change Cipher Spec** – To change ciphers on the fly
- **Alert protocol** for alerts
  - Unexpected message, bad record, decompression failure ..... 
- **HTTP**
- **All performed over SSL Record Layer**
Handshake

Phase 1
Security Capabilities
Session ID, Cipher Suite
Nonces

Phase 2
Server sends certificates,
certificate chains,
certificate request

Phase 3
Client sends certificates,
(optional)

Phase 4
Change Cipher Suite
Finish handshake protocol
SSL Record Protocol

Data

Fragment

Compress

Add MAC

Encrypt

Add SSL Header
Phase 1

- Version
- Random Nonces
- Session ID
- Cipher Suite
  - Crypto algorithms supported in decreasing order of preference
  - For key exchange – RSA, Fixed Diffie Helman, Ephemeral DH, Anonymous DH, Fortezza
  - For Cipher Spec
    - Algorithms – 3DES, RC4, RC2, DES, DES40, IDEA, Fortezza
    - MAC – MD5, SHA1
    - IsExportable – True or False
- Compression Method
Phase 2,3,4

- Server Authentication and Key Exchange
  - Certificate, Certificate Chains
  - Server Key exchange message
  - Certificate request
  - Server done

- Phase 3
  - Client Authentication and Key Exchange
    - 384 bit (48 byte) pre-master key
    - If login is used, it is outside the scope of the protocol
      - Can be used safely with HTTP over SSL
SET

- Secure Electronic Transaction
- For Credit card purchases over the Internet
  - Confidentiality
  - Integrity of data
  - Authentication of card holder
  - Authentication of Merchant
  - Can be used over HTTP, SSL/TLS, IPSec
Electronic Commerce Components

- Cardholder
- Issuer
- Acquirer
- Merchant
- Internet
Electronic Commerce Components

- Card holder
- Issuer
- Acquirer
- Payment Gateway
- Merchant
- Internet
Electronic Commerce Components

Card holder

Internet

CA

Issuer

Payment Network

Acquirer

Merchant

Payment Gateway
Participants

- Cardholder
- Merchant
- Issuer
- Acquirer
- Payment Gateway
- Certification Authority
SET Process

- Customer opens an account
  - Gets a certificate
- Merchants get certificates
  - Opens an account with an acquirer (bank)
  - Establishes a relationship with a payment gateway
- Customer verifies merchant
- Customer places order
  - Merchant verifies customer
- Merchant checks validity of card with payment gateway
- Payment gateway interacts with Acquirer
- Acquirer transfers funds from Issuer to Merchants account
Dual Signature (DS)

- Customers order has two parts
  - Order Information
  - Payment Information
- Merchant does not need to know credit card number (payment information)
- Acquirer bank does not need to know order information (order information)
Dual Signature

Payment Info

Order Info

H – Message Digest (SHA-1)
E – RSA Encryption
KRc – Customers Private Key
POMD – Payment and Order Info MD
Purchase Request

CHC – Cardholder Certificate

CHC – Cardholder Certificate
SET Transactions

- **Purchase request**
  - Acknowledged with purchase response by the merchant

- **Payment Authorization**
  - EPI (Encrypted PI + DS + OIMD)
  - Digital Envelope (sent by customer)
  - Merchant authorization Info
    - Transaction ID signed by merchant, encrypted with one-time key
    - One time key in an envelope encrypted with gateways public key
    - CHC and Merchants certificate
    - Gateway responds with Authorization Response
  - At this point, merchant can commit to selling goods

- **Payment Capture**
  - Capture request – Initiated by merchant. Gateway performs necessary action for transfer of funds.
  - Capture response