Internet Security (188.366)

Internet Applications

Adrian Dabrowski, Thomas Frühwirth, Aljosha Judmayer, Georg Merzdovnik, and **Johanna Ullrich**

inetsec@seclab.tuwien.ac.at

Internet Services

- Internet infrastructure
  - traditional (old) services
  - emerged to satisfy needs from the beginning of the Internet
  - provides basic services that other applications rely upon
  - often no or little security in mind
  - can cause indirect security problems, or prepare and enable attacks against target application
- Particular services
  - remote access (telnet, rservices, ssh)
  - name resolution (DNS)
  - file transfer (FTP)
  - mail transfer (SMTP)
  - time synchronization (NTP)

Changing Threat Models

- Then
  - Internet (ARPANET, 1969) connected hosts located at a handful of research organizations.
  - Only people working at these organizations had accounts on these hosts (trusted users)
  - Only few administrators had root accounts on these hosts (trusted administrators)
- We are still using protocols designed in that era
  - Telnet (RFC 137, 1971), FTP (RFC 114, 1971), SMTP (RFC 821, 1982), DNS (RFC 882-883 1983), NTP (RFC 958, 1985)
- Now:
  - Anyone can connect

Part 0.9

Remote Access

Remote Access

- telnet, rlogin
  - horrible security
  - plaintext passwords
  - connection hijacking
  - fortunately, it is virtually not used anymore

- ssh
  - secure replacement
  - ssh version 1 (1995)
  - insecure because of possibility to insert data into remote stream
  - ssh version 2 (1996) is current, and should be used
  - Port tunneling
  - Remote Copy

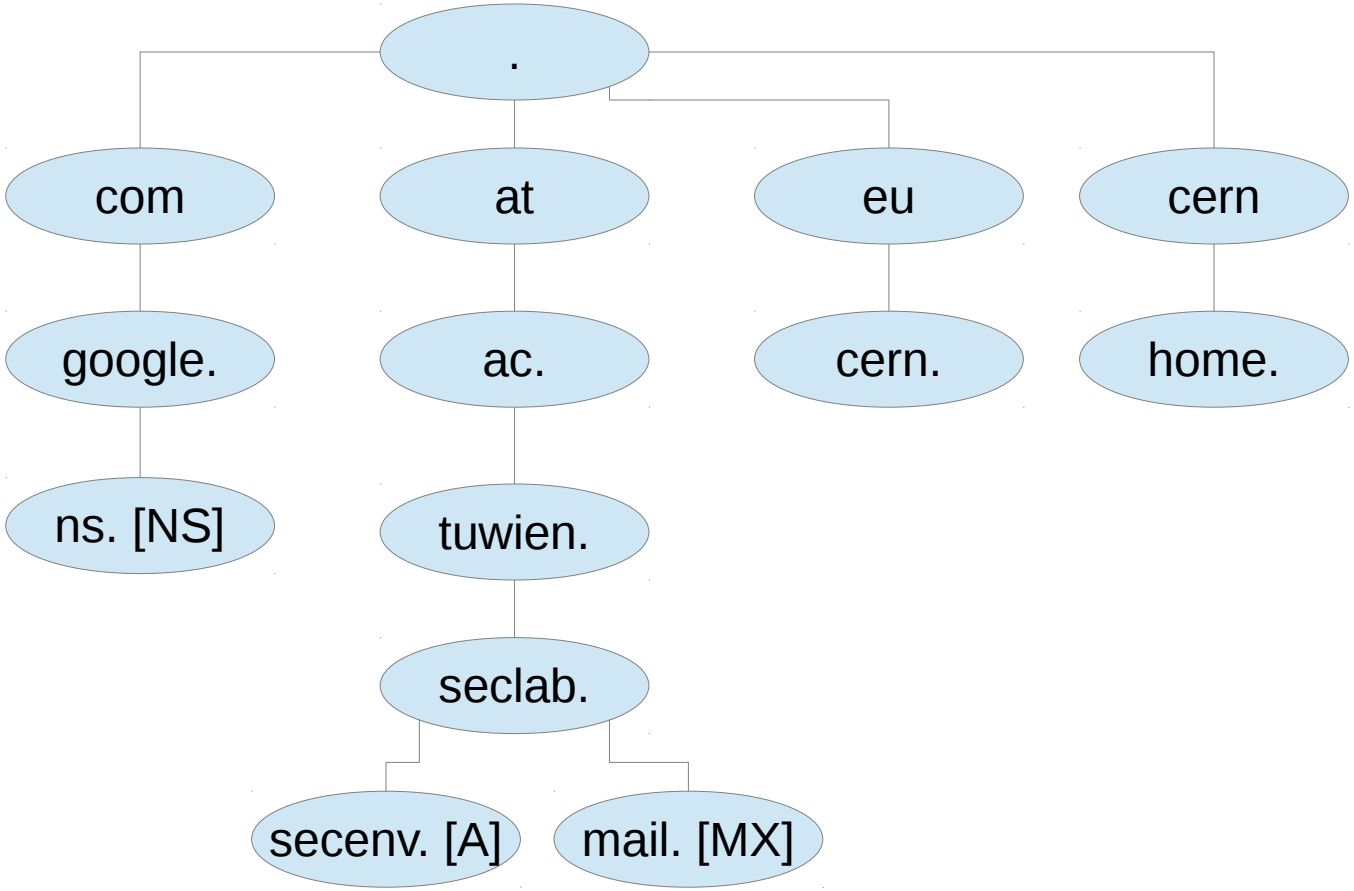
Part 1

DNS

DNS: Domain Name System

- initially specified in RFC 1034/1035
- Maps domain names to IP adresses
  - distributed database
  - name space is hierarchically divided
  - each domain is managed by a name server
- uses mostly UDP
  - sometimes TCP for long queries and for zone transfers between name servers

DNS Hierarchy



Root Zone

TLD

2nd Level

3rd Level / Resource Records / ...

Name Server

- Domain responsibilities are nested
  - odin.auto.tuwien.ac.at is responsible for resolving secenv.seclab.tuwien.ac.at
  - tunamec.tuwien.ac.at is responsible for resolving seclab.tuwienac.at
  - [d,j,u,…].ns.at is responsible for resolving tuwien.ac.at
  - Root DNS is responsible for resolving .at
- Root name servers
  - 13 machines distributed around the world
  - associated with the top level of the hierarchy (.org, .com, .at, etc..)
  - dispatch queries to the appropriate domains
- Server types
  - primary (authoritative for the domain, loads data from disk)
  - secondary (backup servers, get data through **zone transfers**)
  - caching-only (relies on other servers but caches results)
  - forwarding (simply forwards query to other servers)

Name Server

- A server that cannot answer a query, forwards the query up in the hierarchy
- Then, the search follows the correct branch in the hierarchy down to the authoritative server
- The results are usually maintained in a local cache
- Reverse lookup
  - mapping from IP addresses to names
  - also called pointer queries
  - use dedicated branch in name space starting with ARPA.IN-ADDR or ARPA.IP6
  - example
  - if 128.131.172.79 is resolved, this is mapped into 79.172.131.128.in-addr.arpa

DNS Clients

- At least one name server has to be specified
  - e.g., Linux uses /etc/resolv.conf
- Queries can be
  - recursive
    - require a name server to find the answer to the query itself
  - iterative
    - instead of the resolved name another server‘s address is returned, which can be asked
      - ANSWER: I don't know, but ask 1.1.1.1
- Lookup can be performed with
  - nslookup, host, dig

DNS Data

- same message format for requests and replies (binary)
- contains questions, answers, authoritative information
- DNS data is structured in Resource Records, which store the information.
- Different types of RR exist:

A/AAAA	defines an IP address for domain name
HINFO	host information (CPU, OS)
NS	authoritative name server for domain
MX	mail server for domain
TXT	human-readable information ((ab)used for SPF)

DNS Data

```
$  dig +trace www.google.com
; <<>> DiG 9.7.3-P3 <<>> +trace www.google.com
;; global options: +cmd
.         82458     IN        NS        c.root-servers.net.
…
.         82458     IN        NS        a.root-servers.net.
;; Received 449 bytes from 192.168.178.1#53(192.168.178.1) in 56 ms

com.      172800    IN        NS        l.gtld-servers.net.
…
com.      172800    IN        NS        i.gtld-servers.net.
;; Received 504 bytes from 198.41.0.4#53(a.root-servers.net) in 181 ms

google.com.         172800    IN        NS        ns2.google.com.
google.com.         172800    IN        NS        ns1.google.com.
google.com.         172800    IN        NS        ns3.google.com.
google.com.         172800    IN        NS        ns4.google.com.
;; Received 168 bytes from 192.43.172.30#53(i.gtld-servers.net) in 75 ms

www.google.com.     604800    IN        CNAME     www.l.google.com.
www.l.google.com.   300       IN        A         173.194.35.178
www.l.google.com.   300       IN        A         173.194.35.180
www.l.google.com.   300       IN        A         173.194.35.176
www.l.google.com.   300       IN        A         173.194.35.177
www.l.google.com.   300       IN        A         173.194.35.179
;; Received 132 bytes from 216.239.32.10#53(ns1.google.com) in 73 ms
```

DNS Security Issues

- Daemon vulnerabilities
  - BIND named has a bad security history
- DNS often provides rich information (IETF 89)
  - IP addresses
  - HINFO records
  - WKS (Well Known Servers)
  - can be gathered via exhaustive queries or via zone transfers if configured incorrectly
  - IP scanning is not necessary in many cases
- Simple DNS spoofing
- DNS cache poisoning

Simple DNS Spoofing

- Used when authentication is performed based on DNS names with reverse lookup
  - e.g., trusted.example.com may login using rlogin without specifying a username/password
  - or, only trusted.example.com may login at all

- Concept
  - domain name is obtained through reverse DNS query
  - a DNS query is forwarded to the authoritative DNS server for the IP address that logs in (under control of the attacker)
  - this DNS server replies with the (faked) trusted name

Simple DNS Spoofing

66.66.66.3
Auth for 66.66.66.0/24

123.1.2.22
resolver.example.com

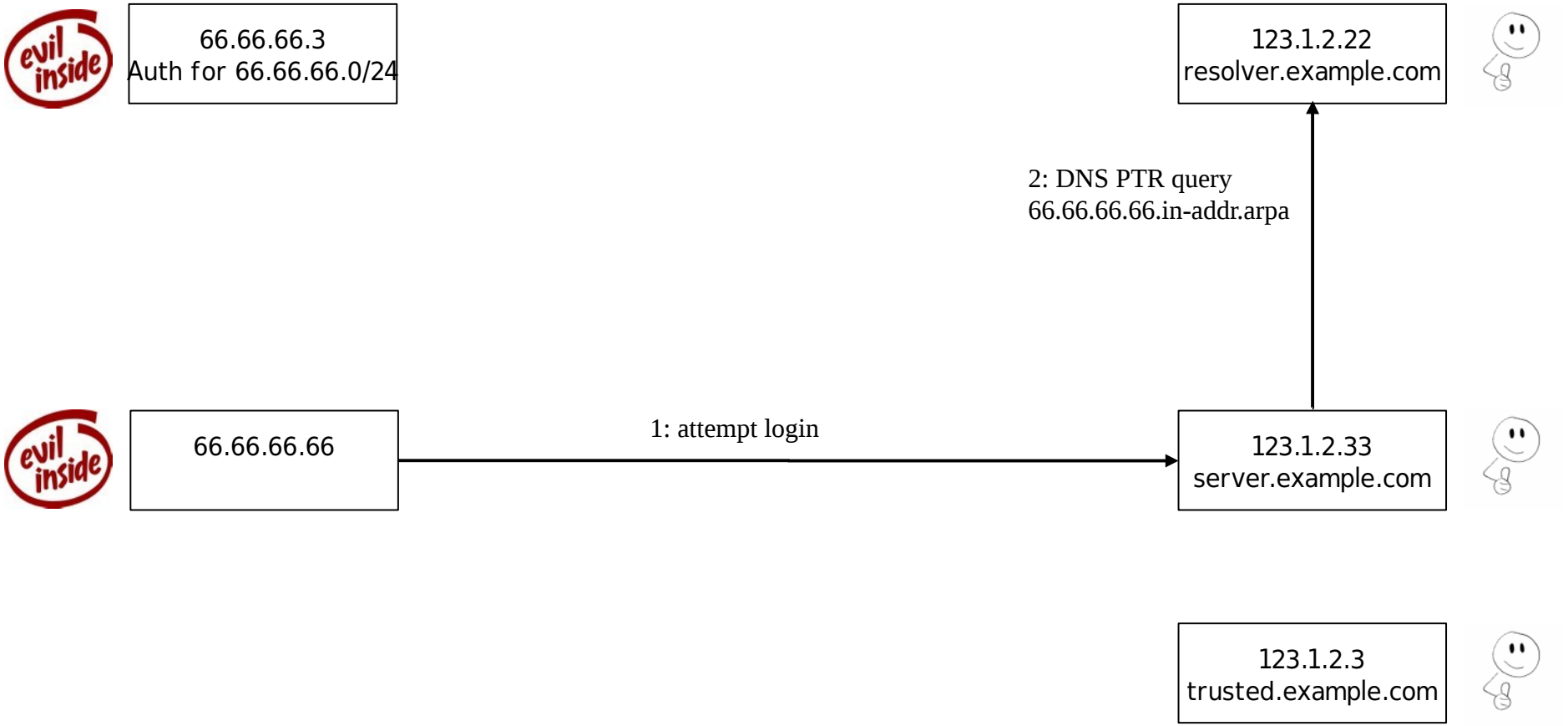
66.66.66.66

1: attempt login

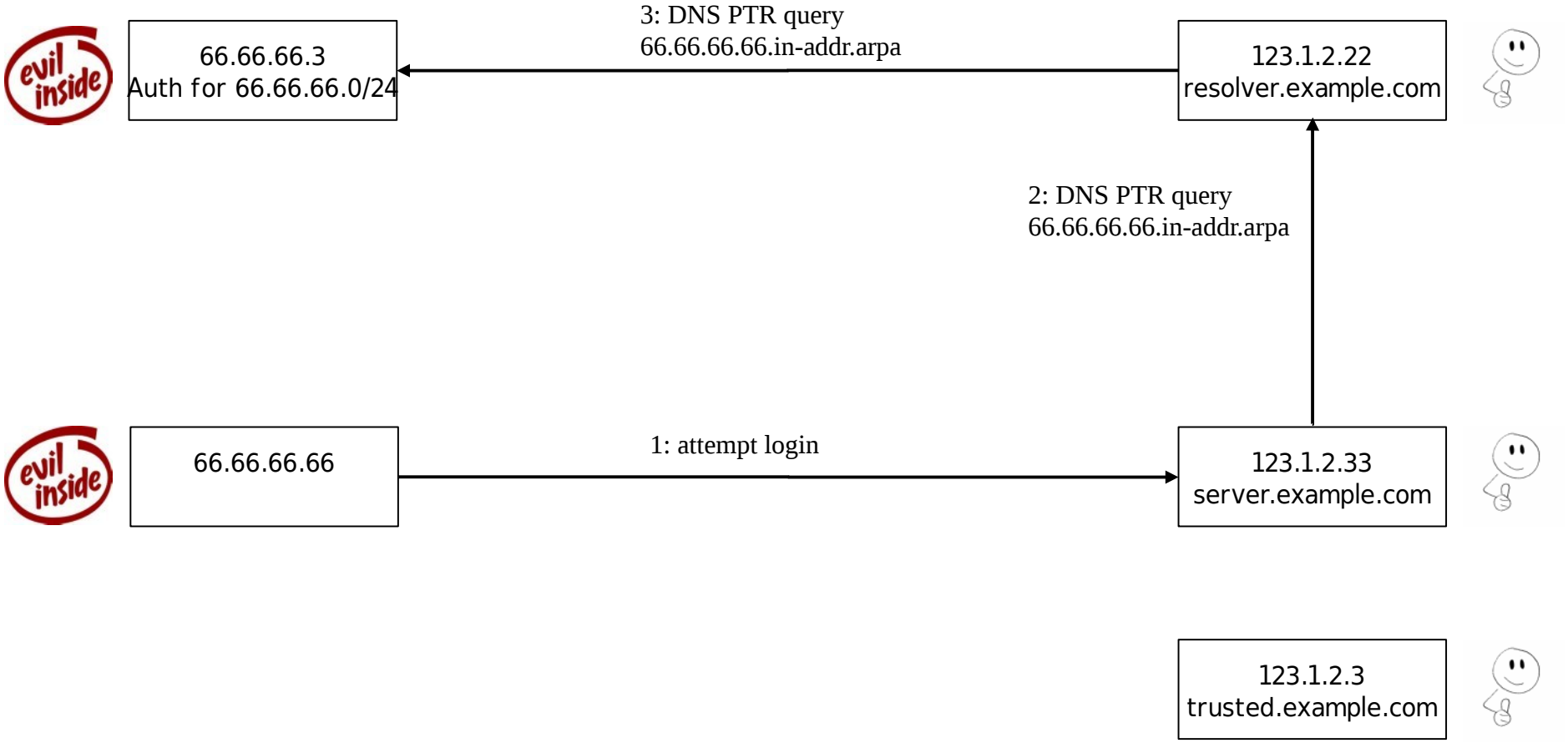
123.1.2.33
server.example.com

123.1.2.3
trusted.example.com

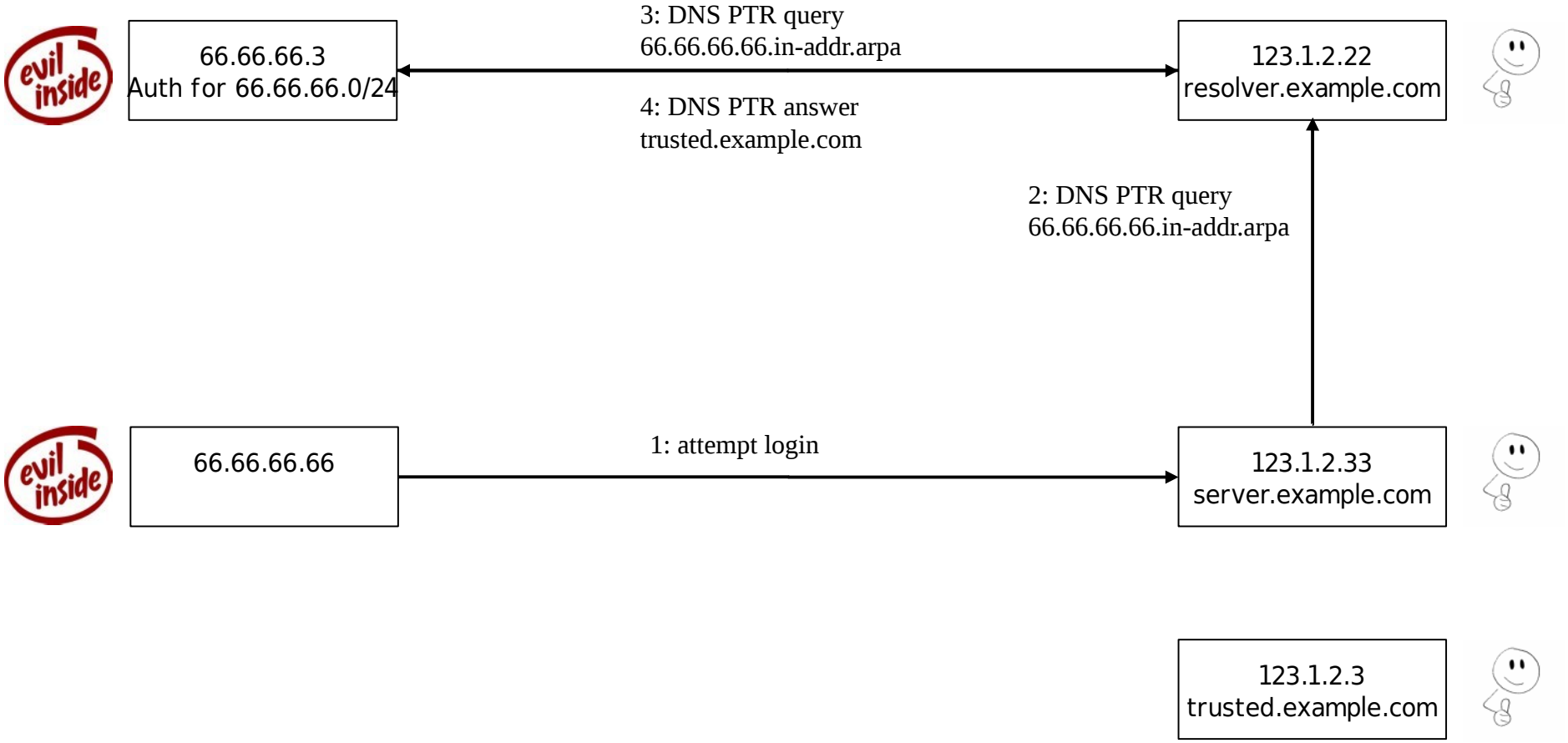
Simple DNS Spoofing



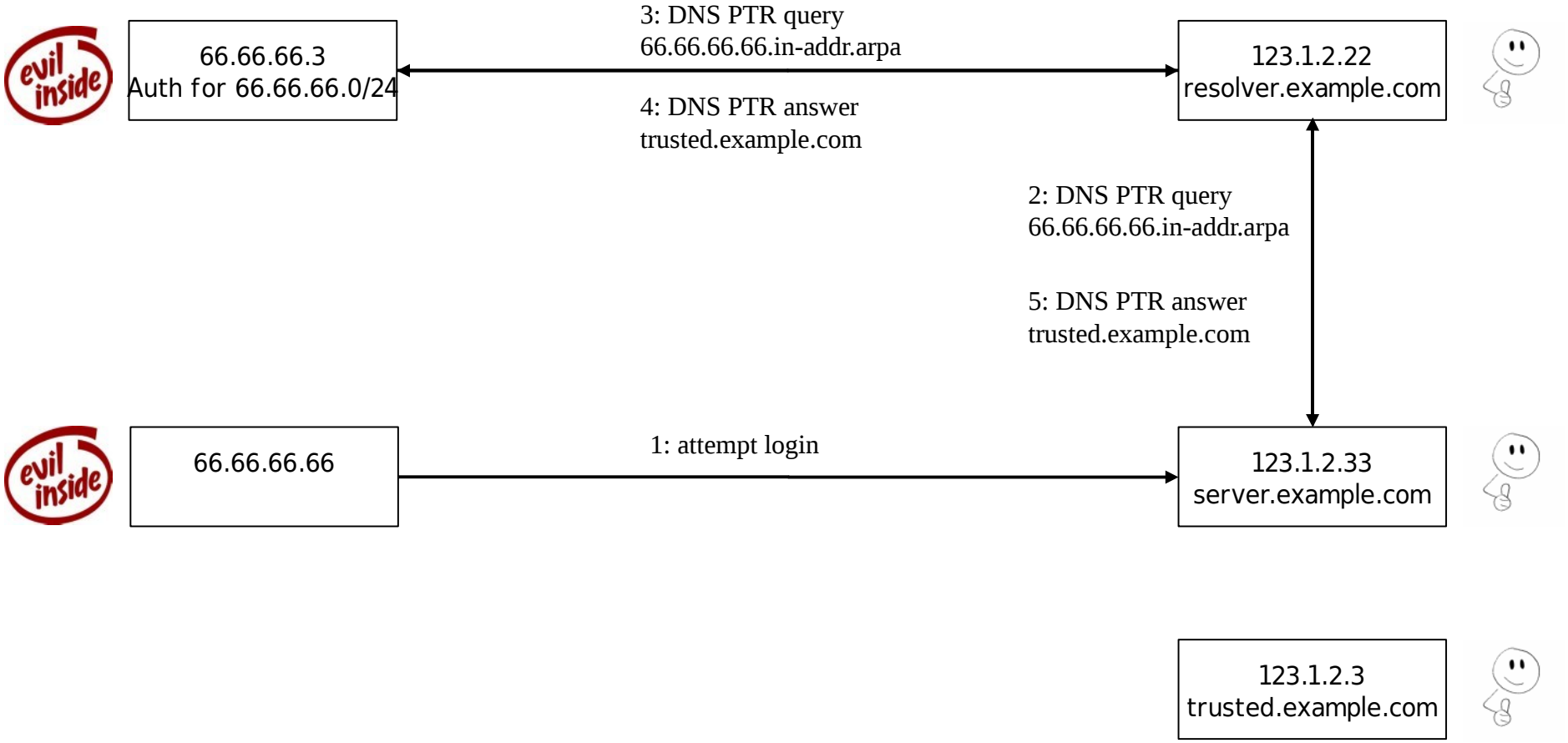
Simple DNS Spoofing



Simple DNS Spoofing



Simple DNS Spoofing



Simple DNS Spoofing

- Countermeasure
  - use double reverse lookup
  - reverse lookup 66.66.66.66 => trusted.example.com
  - now do forward (normal) lookup for trusted.example.com => 123.1.2.3
  - refuse login!

DNS Cache Poisoning

- DNS requests/replies normally sent over **UDP**
  - Reply not authenticated
  - Remember UDP?
  - Spoof reply
  - Race against legitimate reply

- DNS Hijacking possibilities
  - racing with the server with respect to a client
  - racing with a server with respect to another server

Spoofed DNS reply

- Must match a query!
- Use correct (spoofed) source IP address of the real server
- Use correct destination UDP port (the source port from which the query was sent)
- Answer correct query
- Correct value of DNS nonce field
  - 16 bit, randomly selected request id

DNS Cache Poisoning

- Attack a caching-only server
- Send a request for host.example.com
  - Server will send request to authoritative NS for example.com
- Immediately send many spoofed replies
  - Source IP is one of the NS for example.com (~4 options)
  - Guess destination UDP port (16 bit)
  - Guess DNS nonce (16 bit)
- Number of replies needed on average:
  - 2^16 * 2^16 * 4 /2 = 2^33 ~= 8 billion
- Need multi-terabit/s pipe to do it in 1 second (before real reply)

Improving the Chances

- Src port known
  - many servers always send queries from same UDP port
  - bind up to ~9.4.2 chose port at startup
  - attacker can find it out by setting up authoritative server for his own domain, and querying for that domain
  - now only need about 2^16 * 4 / 2 ~= 130000 attempts
- Birthday attack
  - some servers allow multiple outstanding requests for same domain
  - send 100 queries+100 answers~=10000 chances of guessing
- But still, If you lose the race, can't retry until cache expires (~1 week)

DNS Cache Poisoning: Effects

- Redirect traffic
- Denial of Service
- MITM attack with no physical access
- Redirect email (MX record…)
- Exploit auto-update
  - java updater uses no crypto: just need to poison java.sun.com
    - metasploit evilgrade module
    - video demo: http://www.infobyte.com.ar/demo/evilgrade.htm

- Example: WebView for mobile devices
  - Android, iPhone, Windows Phone, Blackberry

DNS Cache Poisoning: Countermeasures

- Check the DNS server(s) you use: use random src ports
  - sniff outgoing query traffic (often not possible)
  - Tool at https://www.dns-oarc.net/oarc/services/dnsentropy
  - run a NS for your own domain, make a recursive query and sniff incoming packets
- Block queries to your recursive resolvers from outside your network
- DNSSEC:
  - authoritative replies are cryptographically signed
  - deployed on DNS root zone in 2010
  - deployed on most TLDs now (.AT Domain: 29.02.2012)
  - http://www.internetsociety.org/deploy360/dnssec/statistics/

DNS Level Poisoning

- Not only done by the “bad”-guys
- Very common on ISP-Level with the DNS servers that will get pushed to your modem/router
  - Ads: don't return RFC ERR NXDOMAIN, instead redirect to own search engine with ads
  - Censorship: Blocking “URLs”
  - Law enforcement: Redirect and inject malicious traffic
- Use a trustworthy, “working” DNS server or hosts file
- Keep in mind, DNS also exposes which sites you visit

Part 2

FTP

FTP: File Transfer Protocol

- initially specified in RFC 542
- provides file transfer service
- based on TCP
- client / server architecture
  - client (ftp) sends a connection request to the server (ftpd)
  - server is listening on port 21
  - client provides username and password to authenticate
  - client uses the GET and PUT commands to transfer files

FTP

- 2 TCP connections are used
  - control stream for commands
  - data stream for the actual data that is transmitted

- Active Mode:
  - Client tells the server to connect to one of its local ports using the PORT command
  - Server opens a connection from port 20 to the port specified by the client
  - Transfer is executed and the connection is closed

- Passive Mode:
  - Client issues the PASV command
  - Server opens a port and sends port number to the client
  - Client connects to the port specified by the server
  - Transfer is executed and the connection is closed

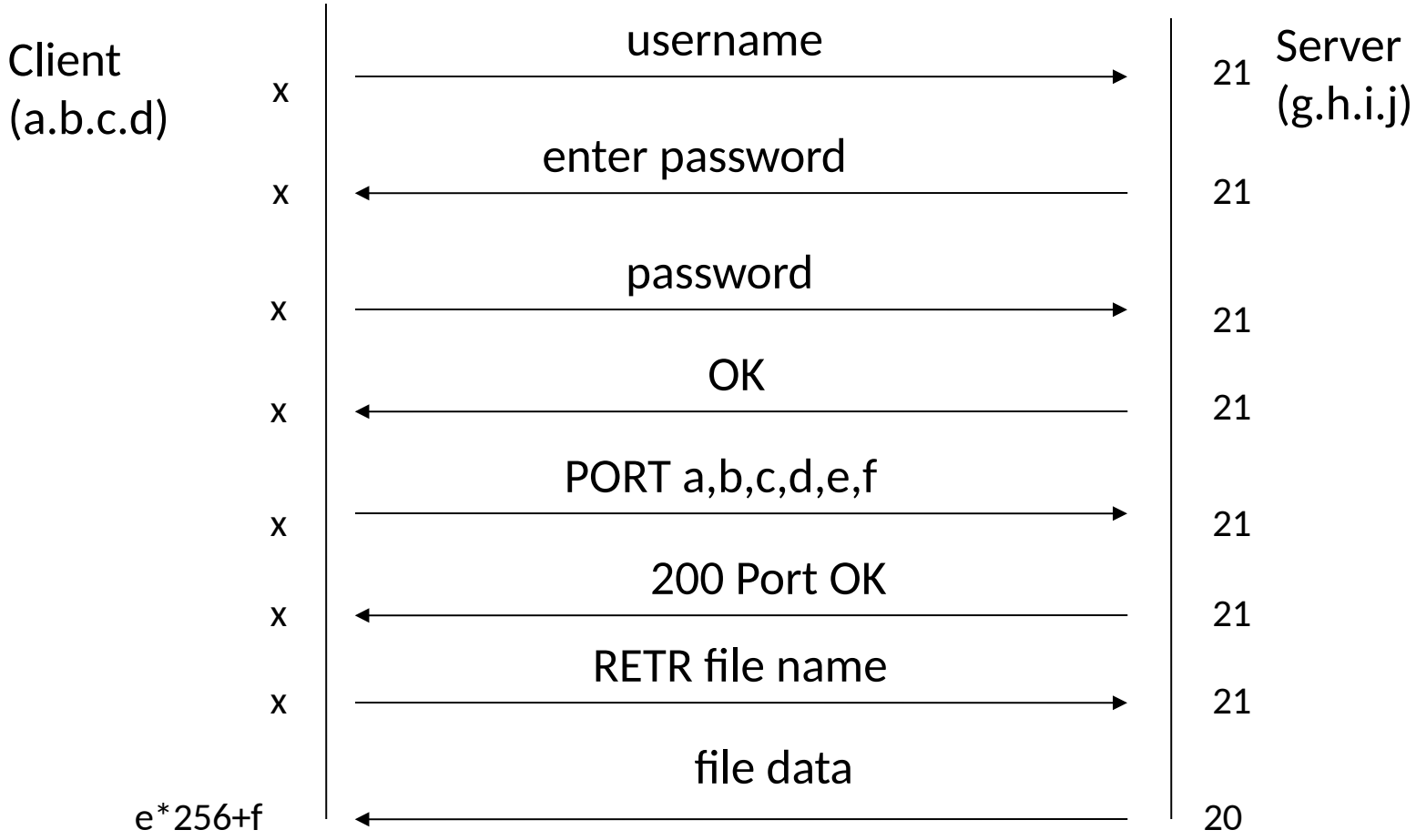
FTP

- 2 TCP connections are used
  - control stream for commands
  - data stream for the actual data that is transmitted

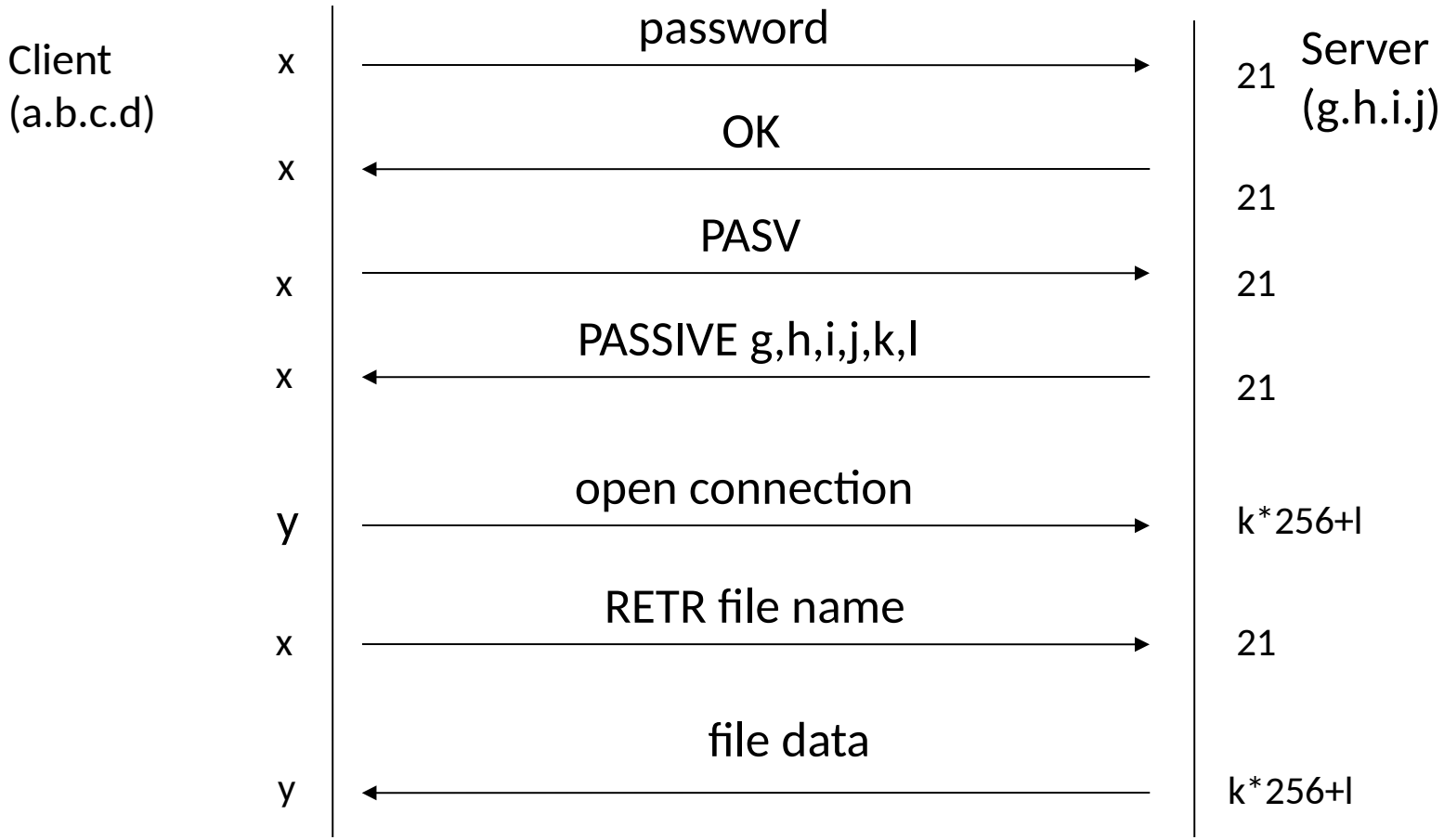
- Active Mode:
  - Client tells the server to connect to one of its local ports using the PORT command
  - Server opens a connection from port 20 to the port specified by the client
  - Transfer is executed and the connection is closed

- Passive Mode:
  - Client issues the PASV command
  - Server opens a port and sends port number to the client
  - Client connects to the port specified by the server
  - Transfer is executed and the connection is closed

FTP Protocol



Passive FTP


Client
(a.b.c.d)
Server
(g.h.i.j)
x
21
password
OK
x
21
PASV
x
21
PASSIVE g,h,i,j,k,l
x
21
y
open connection
k*256+l
x
RETR file name
21
y
file data
k*256+l

FTP Security

- Server implementation vulnerabilities

- Configuration errors
  - allow "anonymous" user to write files
  - write to user home directory
    - Can be abused to write files into home directories that are normally used for authentication (e.g. .ssh/authorized_keys)
    - If an anonymous user is allowed to put such a file in the home directory he can get access to the computer, using private key authentication

PASV Connection Theft

- Attacker can connect to port that was opened by server before the legitimate client does

- Since the command connection is still established, client commands lead to file transfers between attacker and server

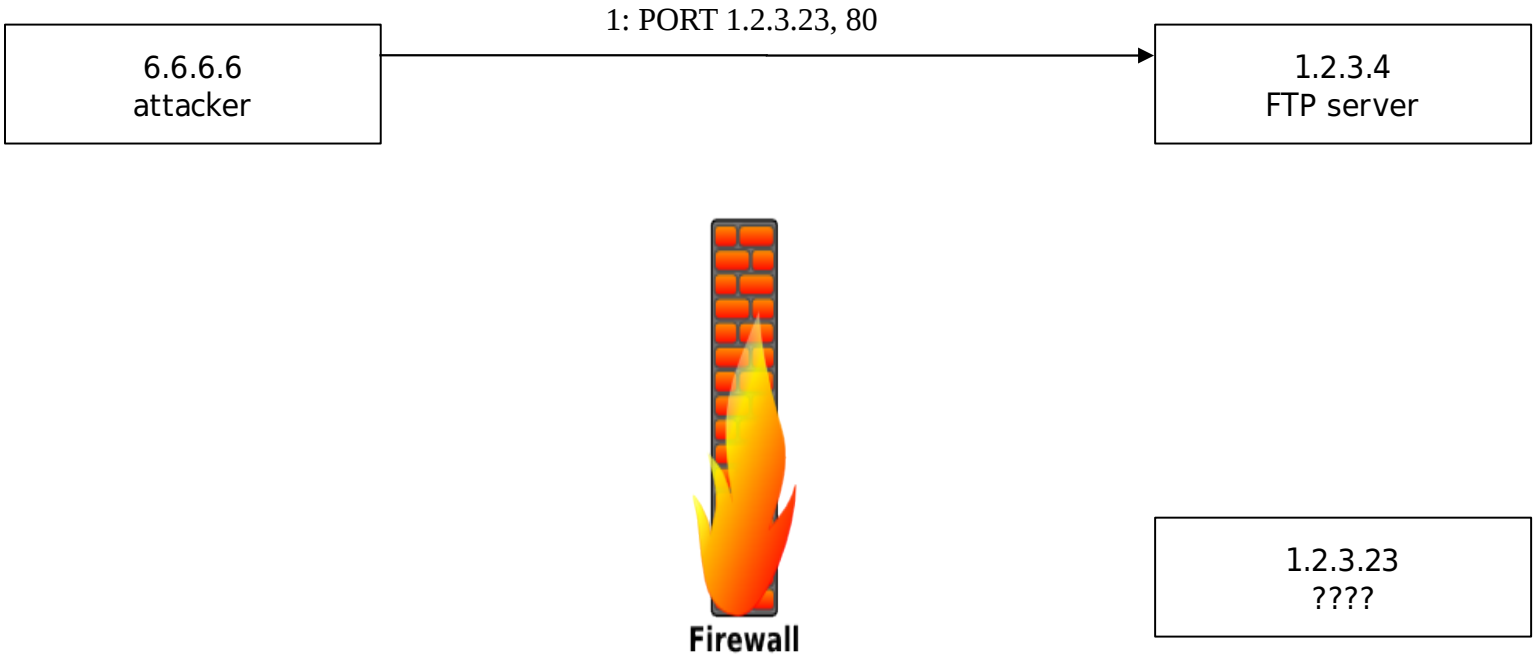
FTP Bounce

- The PORT command
  - Tells the server the address and port to be used when opening a data connection
- According to the RFC 959 the address does not have to be the same as the one the client has
- Therefore it is possible to instruct a server to open a connection to a third host

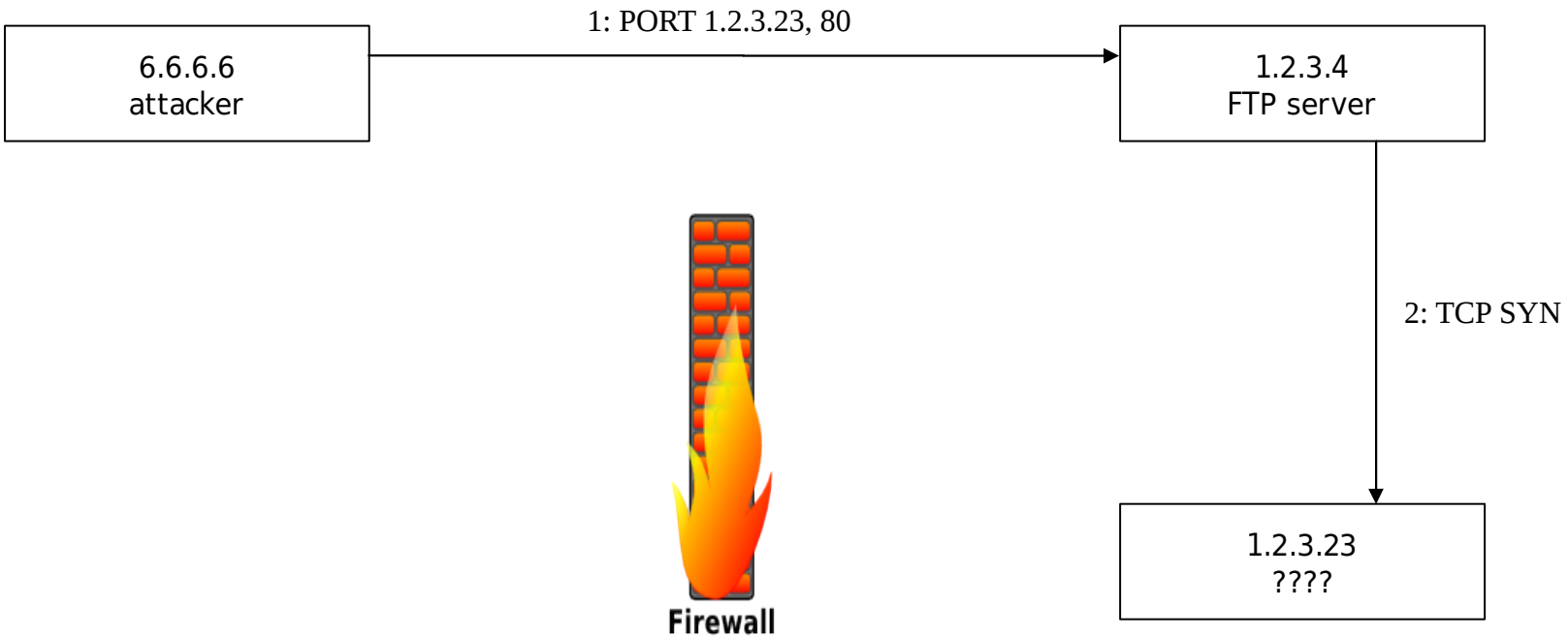
FTP Bounce

- Can be used to perform a TCP portscan
- The host running ftpd appears to be the source of the scan
- It is possible to scan „behind“ a firewall
  - suppose that only port 21 and 20 are open at the firewall

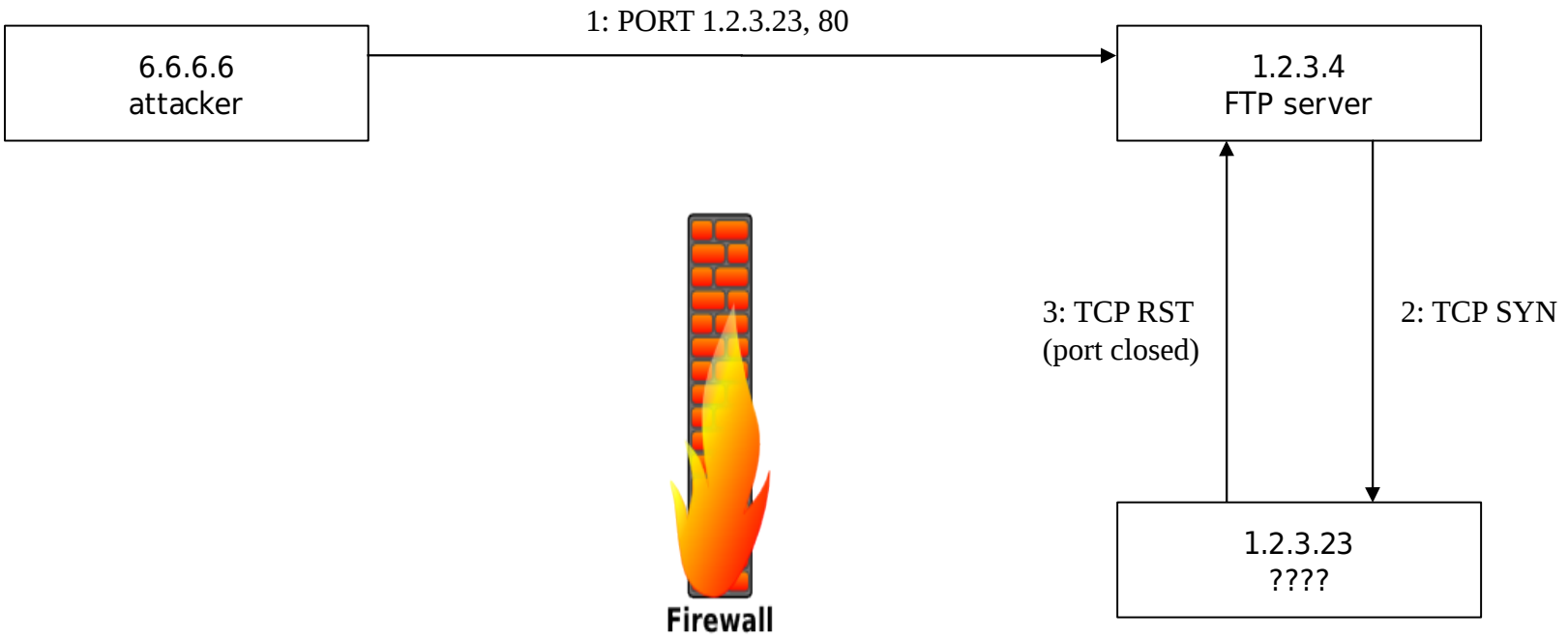
FTP Bounce: Port Scan



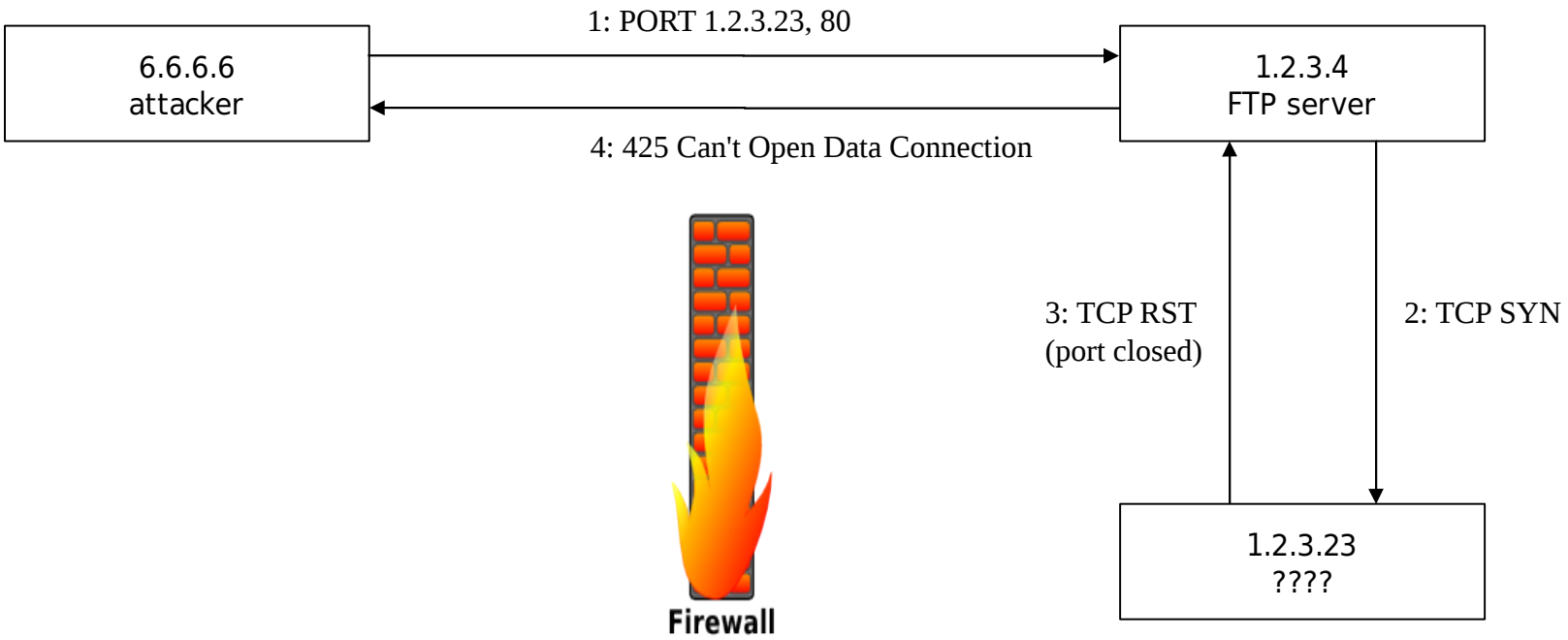
FTP Bounce: Port Scan



FTP Bounce: Port Scan

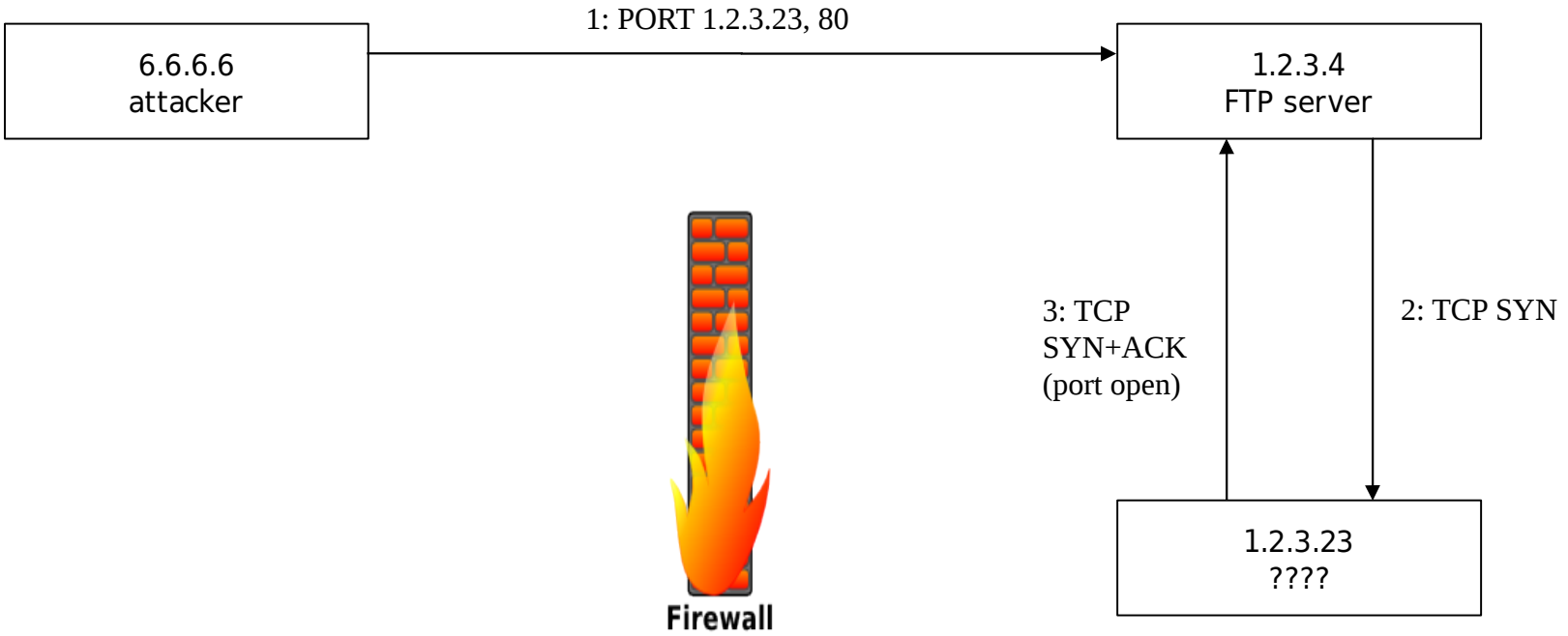


FTP Bounce: Port Scan

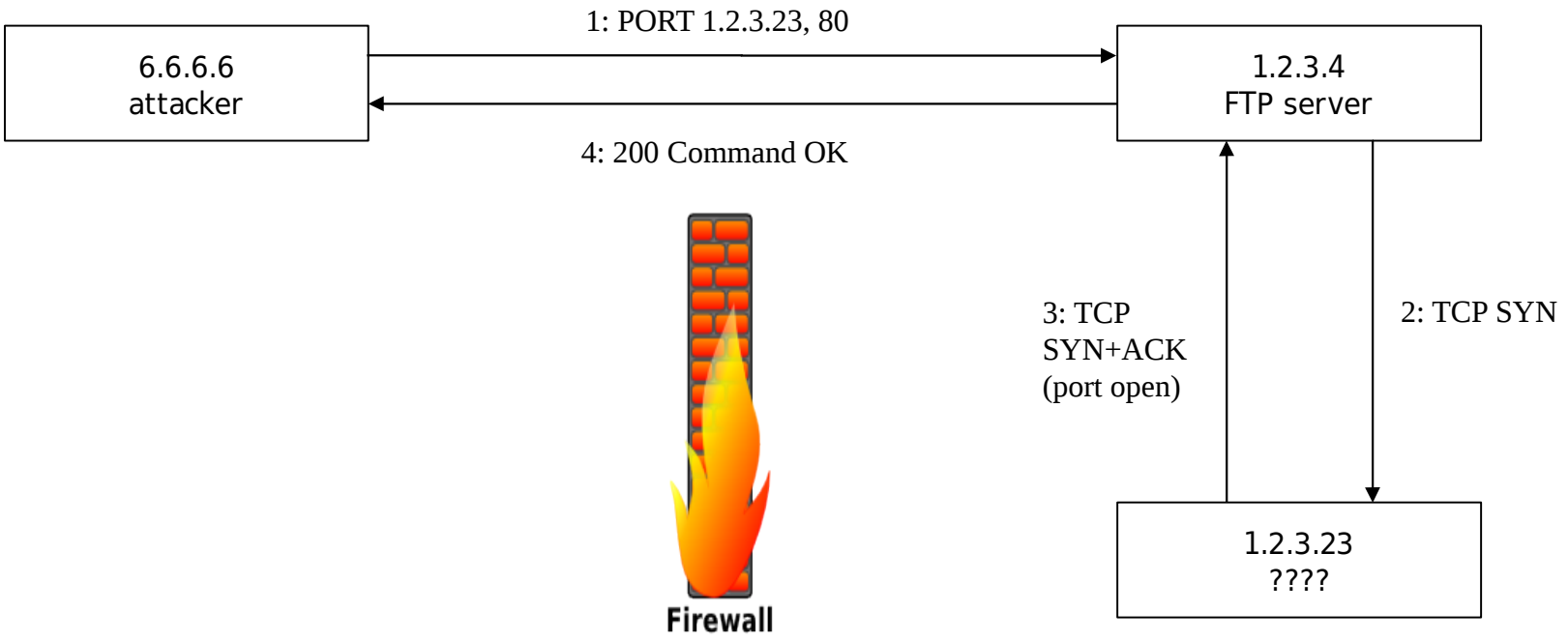


- 1.2.3.23 does not have an HTTP server running on port 80 !!

FTP Bounce: Port Scan



FTP Bounce: Port Scan



1.2.3.23 has a server running on port 80!

FTP Bounce

- Not only useful for port scans: can be used to send data to arbitrary ports
  - if an FTP writable directory exists, arbitrary data can be sent to a third host
  - can be used to bypass restrictions (IP based authentication)
  - connection laundry

- Step 1:
  - upload data to the ftp server (PUT mydata)
- Step 2:
  - PORT destination IP, destination port
- Step 3:
  - GET mydata

Part 3

SMTP

SMTP: Simple Mail Transfer Protocol

- initially specified in RFC 821
  - de facto standard for email transmission
  - simple, text-based protocol
  - MIME used to encode binary files (attachments)
  - listens on port 25

- push protocol:
  - used to send email
  - used to exchange emails between servers
  - clients have to retrieve emails via other protocols such as IMAP or POP

SMTP Session

```
S: 220 www.example.com ESMTP Postfix
C: HELO mydomain.com
S: 250 Hello mydomain.com
C: MAIL FROM: sender@mydomain.com
S: 250 Ok
C: RCPT TO: friend@example.com
S: 250 Ok
C: DATA
S: 354 End data with <CR><LF>.<CR><LF>
C: Subject: test message
C: From: sender@mydomain.com
C: To: friend@example.com
C:
C: Hello,
C: This is a test.
C: Goodbye.
C: .
S: 250 Ok: queued as 12345
C: QUIT
S: 221 Bye
```

No Authentication

SMTP

- Security Issues
  - mail servers have wide distribution base and are publicly accessible
    - software vulnerabilities
    - configuration errors
  - *sendmail*
    - one of the first SMTP implementations (MTAs)
    - long history of vulnerabilities
    - complicated configuration (M4 macro language)
    - e.g., buffer overflow in Sendmail 8.12.9 and before (2003)
  - *postfix*, *qmail*
    - secure replacements

SMTP

- No Authentication
  - everyone can connect to a SMTP server and transmit a message
  - server cannot check sender identity (besides IP address/domain name)

- Fundamental reason: open, distributed system
  - you can receive email from anyone on the Internet
  - there is no central authority
  - this is why email was so successful!
  - …also the root cause of SPAM

Open Mail Relay

- The mail server for example.com should deliver:
  - messages from email accounts of example.com
  - messages to email accounts of example.com
- It should NOT relay messages:
  - from untrusted sources, to destinations outside example.com
  - Open Mail Relay: will be used to deliver SPAM

SMTP Authentication

- IP address
  - check if user is inside the example.com network
- Extensions
  - SMTP-AUTH
    - access control list with explicit login
    - clients must be aware of SMTP-AUTH
  - POP-before-SMTP
    - logins are simulated by POP request (which require a login)
    - when a client performs a POP request, its IP address is authenticated with the SMTP server for some time (e.g., 30 minutes)

MTA Encryption

- POPS / IMAPS / SMTPS (SSL/TLS)
  - Like HTTPS but for POP/IMAP/SMTP
- Extensions
  - STARTTLS
    - Using plaintext standard Port (POP 110, IMAP 143, SMTP 25)
    - But can start an encrypted session unsing STARTTLS command
    - SSL stripping attack
  - Dedicated SSL Ports (explicit SSL)
    - Encryption is mandatory on the following ports:
    - POP 995, IMAP 465, SMTP 993
    - No stripping attack possible

Address Spoofing

- Authentication by IP:
  - Anyone in example.com network can send email from ceo@example.com
- Sender can forge source address
  - pretend to be relaying email from mybank.com

Address Spoofing (countermeasures)

- SPF: Sender Policy Framework
  - leverage DNS infrastructure
  - owner of mydomain.com specifies which IPs are authorized to send emails from *@mydomain.com
  - uses TXT records in DNS server (no changes required to DNS implementation)
  - SMTP server can check sender IP/domain name against authorized senders

SPF Example

```
$host -t TXT gmx.net

gmx.net descriptive text "v=spf1 ip4:213.165.64.0/23 ip4:74.208.5.64/26
    -all"
```

- An SPF compliant SMTP server receiving mail from *@gmx.net:
  - will check if client IP is in the list. If not, it will reject mail claiming to be from gmx.net

Part 3.5

SPAM/ PHISHING

Spam

- Unsolicited email messages
- Gather destination email addresses
  - brute force guessing
  - harvesting (web pages, mailing lists, news groups, ...)
  - malware (steal user's address book)
  - verified address are more valuable
- Delivering spam messages
  - own machine (not very smart)
  - other machines
    - open mail relays
    - open proxies
    - web forms
    - zombie nets (compromised machines)

Spam

- Countermeasures
  - spam filtering tools (e.g., SpamAssassin)
  - blacklists: identify origins of spam messages and quickly distribute this information
  - greylisting: temporarily reject email from unknown senders
    - legitimate senders will retry
    - spammers often don't
  - infrastructure
    - SPF (sender policy framework)

Spam

- Reasons for spam
  - legitimate businesses advertise products and services
  - attempts to get money from victims
    - actually quite old idea, was done with letters decades ago
    - victims sometimes even travel to remote places
  - offer of porn or other interesting material to lure people on sites where Malware can be installed

- Statistics
  - Symantec Internet Security Threat Report 2014
    - 29 billion spam mails a day
    - 66% of all mails in 2013 (89% in 2010, still around 75% in 2011)
    - most spam mails produced by botnets
    - 25% of spam mails contained malware as URL

Phishing

- Exploits openness/weakness of SMTP protocol and Humans (social engineering aspects)
- Tricks people into providing sensitive information
  - create a situation that asks receiver to act on (urgent) problem
  - provide a link to site to solve problem
  - site prepared by attacker
    - appearance of site is spoofed
    - asks for personal information
- Interesting side note
  - scammers typically require people to launder money
  - additional spam mails that invite people to “earn money with their bank account”
- Nowerdays the “Presidential attack” is very common

Phishing

- Camouflage techniques
  - use images, look&feel from original site
  - sender name and email addresses can be faked easily
  - attempt to avoid obvious spelling and grammar mistakes :-)

  - link to phishing site must be obfuscated
  - URL and port redirection
    http://www.bank.com@evil.com:80/index.html

- Examples:
  - http://www.fraudwatchinternational.com/phishing/
  - http://www.phishtank.com/

NTP

- Network Time Protocol
  - Strata
  - sync time from own and layer above
  - tell time to lower layer
- Not a lot of security attention
  - mainly MITM (can mess with Kerberos)
- So why am I talking about it???
  - monlist command in older clients sends a list of the last 600 ntp clients connected (NTP reflection attack, recon, DDoS)
  - iOS 9 – 1970s Bug, fixed in 9.3
  - All time based authentification systems, tokens, cookies, etc…

DOS

- Targets
  - Hosts
  - Services
  - Infrastructure – Think “wire capacity”
  - …
- Taxonomy
  - R-U-Dead-Yet?
  - PDoS
  - DoS Level II
  - APDoS
  - …

DOS

- Distributed Denial Of Service
  - Botnets
  - Reflection Attacks
  - Hard to defend against
  - Hard to suppress
- Layer
  - WebApp
  - WebServer
  - TCP
  - IPv6 ND, ...

DOS


MAIN MENU
MY STORIES: 20
RISK ASSESSMENT
100 G
Matthew Prince
Oles
@olesovhcom
Follow
KrebsonSecurity
In-depth security news and investigation
14 The New Normal: 200-400 Gbps DDoS Attacks
FEB 14
Over the past four years, KrebsOnSecurity has been targeted by countless denial-of-service attacks intended to knock it offline. Earlier this week, KrebsOnSecurity was hit by easily the most massive and intense such attack yet — a nearly 200 Gbps assault leveraging a simple attack method that industry experts say is becoming alarmingly common.

Reflected DOS

- Victim only sees packets from reflectors
- Reflectors only see spoofed packets
- Huge amplification factors
- http://blog.cloudflare.com/understanding-and-mitigating-ntp-based-ddos-attacks

Reflected DOS

Protocol	BAF *all*	BAF 50%	BAF 10%	PAF *all*	Scenario
SNMP v2	6.3	8.6	11.3	1.00	*GetBulk* request
NTP	556.9	1083.2	4670.0	3.84	Request client statistics
DNS_{NS}	54.6	76.7	98.3	2.0	
DNS_{OR}	28.7	41.2	64.1	1.3	
NetBios	3.8	4.5	4.9	1.0	
SSDP	30.8	40.4	75.9	9.9	
CharGen	358.8	n/a	n/a	1.0	
QOTD	140.3	n/a	n/a	1.0	
BitTorrent	3.8	5.3	10.3	1.5	
Kad	16.3	21.5	22.7	1.0	
Quake 3	63.9	74.9	82.8	1.0	
Steam	5.5	6.9	14.7	1.1	
ZAv2	36.0	36.6	41.1	1.0	
Sality	37.3	37.9	38.4	1.0	
Gameover	45.4	45.9	46.2	5.3	

Protocol	Amplifiers	Tech.	t_{1k}	t_{100k}
SNMP v2	4,832,000	Scan	1.5s	148.9s
NTP	1,451,000	Scan	2.0s	195.1s
DNS_{NS}	255,819	Crawl	35.3s	3530.0s
DNS_{OR}	7,782,000	Scan	0.9s	92.5s
NetBios	2,108,000	Scan	3.4s	341.5s
SSDP	3,704,000	Scan	1.9s	193.5s
CharGen	89,000	Scan	80.6s	n/a
QOTD	32,000	Scan	228.2s	n/a
BitTorrent	5,066,635	Crawl	0.9s	63.6s
Kad	232,012	Crawl	0.9s	108.0s
Quake 3	1,059	Master	0.6s	n/a
Steam	167,886	Master	1.3s	137.1s
ZAv2	27,939	Crawl	1.5s	n/a
Sality	12,714	Crawl	4.7s	n/a
Gameover	2,023	Crawl	168.5s	n/a

Source: Christian Rossow. "Amplification Hell: Revisiting Network Protocols for DDoS Abuse".
Network and Distributed System Security Symposium, NDSS 2014, San Diego, USA. // Bandwidth/Packet Apmlification Factor (BAF, PAF)

Conclusions

- Traditional Internet applications
  - not built with security in mind
  - some could be easily replaced (telnet, rservices)
  - others cause significant problems
- Remote Access
- DNS
  - simple UDP-based request / reply structure
- FTP
  - different command and data channel, FTP bounce
- SMTP
  - sender authentication lacking, spam, phishing
- NTP
  - lets get DOSing