A Stateful Inspection of FireWall-1

data protect

Thomas Lopatic, John McDonald
TÜV data protect GmbH

tl@dataprotect.com, jm@dataprotect.com

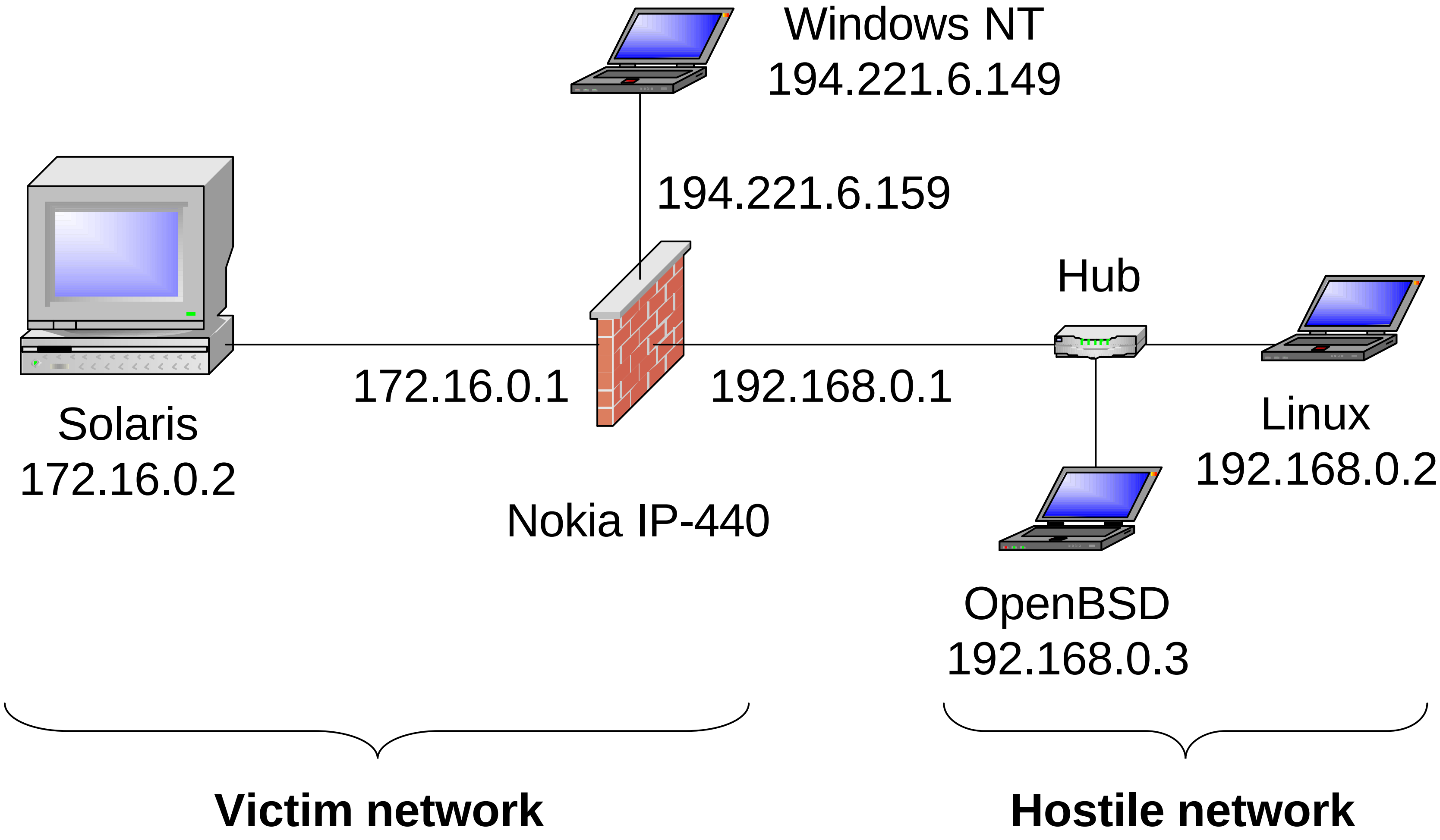
Dug Song
CITI at the University of Michigan

dugsong@umich.edu

Overview

- Architecture of FireWall-1
- Attacking the firewall’s state I
- FWZ encapsulation
- Attacking the firewall’s state II
- Attacking authentication between firewall modules
- Hardening FireWall-1
- The big picture

Topology


Windows NT
194.221.6.149
194.221.6.159
Hub
Solaris
172.16.0.2
172.16.0.1
192.168.0.1
Linux
192.168.0.2
Nokia IP-440
OpenBSD
192.168.0.3
Victim network
Hostile network

Problems in Inspection

- Unreliable / unauthenticated input
- Layering restrictions on inspection
- Layering violations in inspection
- Ambiguous end-to-end semantics

Example: Airport Security

- Unreliable / unauthenticated input

  Examining baggage tags

- Layering restrictions on inspection

  Examining shape, size, weight

- Layering violations in inspection

  Parallelizing bag content inspection

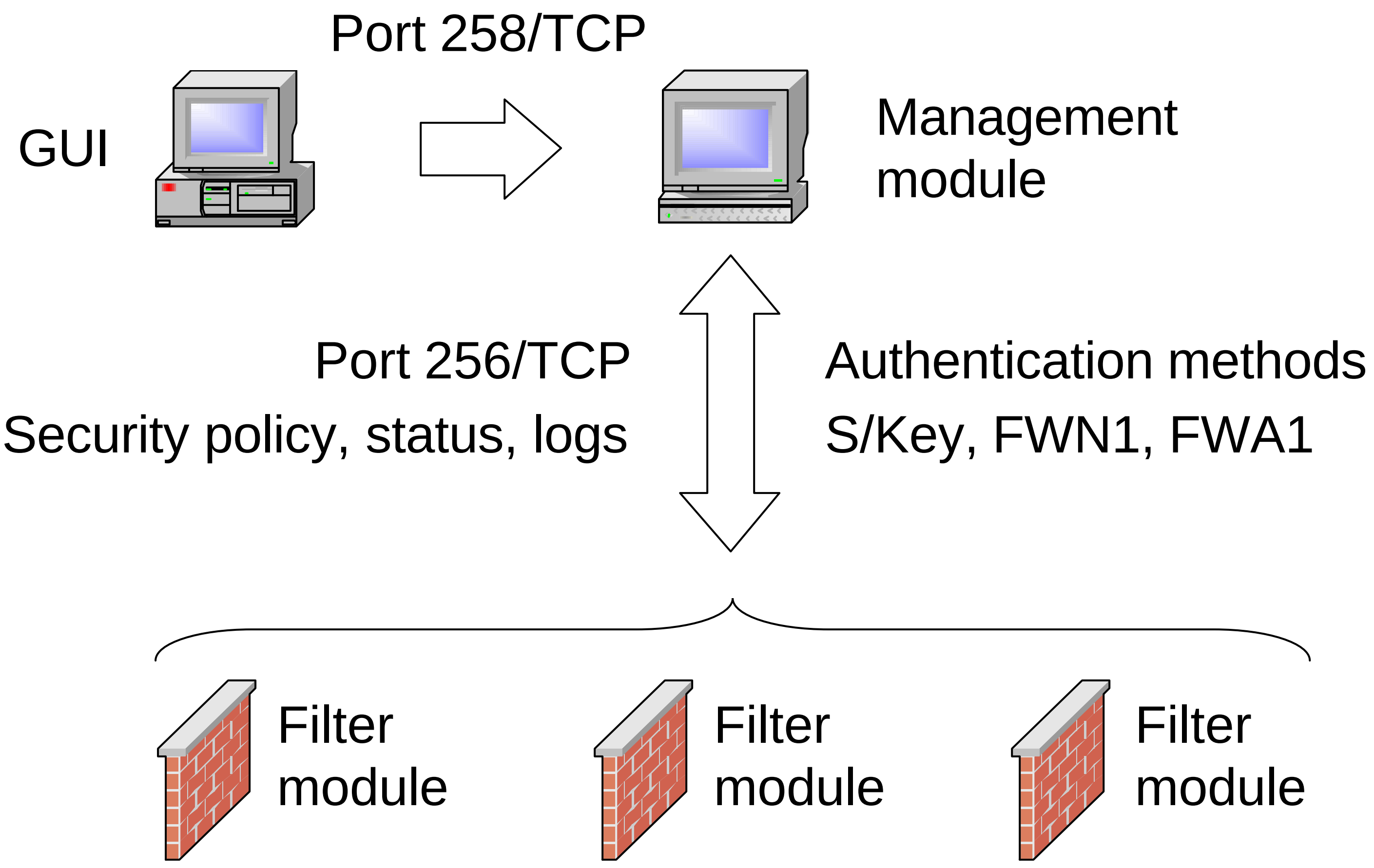
- Ambiguous end-to-end semantics

  Checking for known contraband

Classification of the Attacks

- Unreliable / unauthenticated input
  - **TCP fastmode**
- Layering restrictions on inspection
  - **FWZ VPN encapsulation**
- Layering violations in inspection
  - **FTP data connection handling**
  - **unidirectional TCP data flow**
  - **RSH error connection handling**
- Ambiguous end-to-end semantics
  - **Parsing of FTP “PORT” commands**

FireWall-1 Modules



GUI

Port 258/TCP

Management module

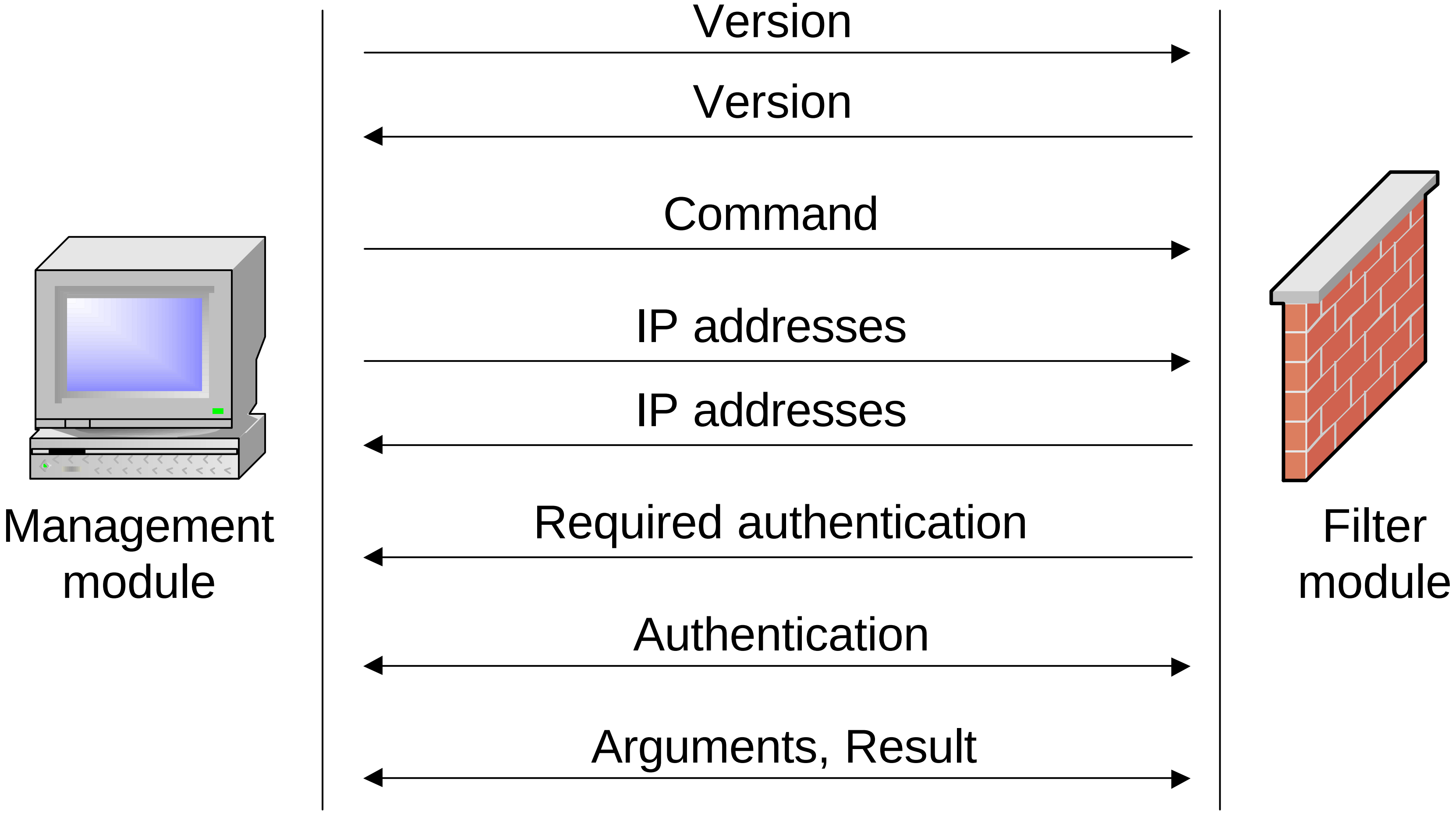
Port 256/TCP

Security policy, status, logs

Authentication methods

S/Key, FWN1, FWA1

Inter-Module Protocol



Version

Version

Command

IP addresses

IP addresses

Required authentication

Authentication

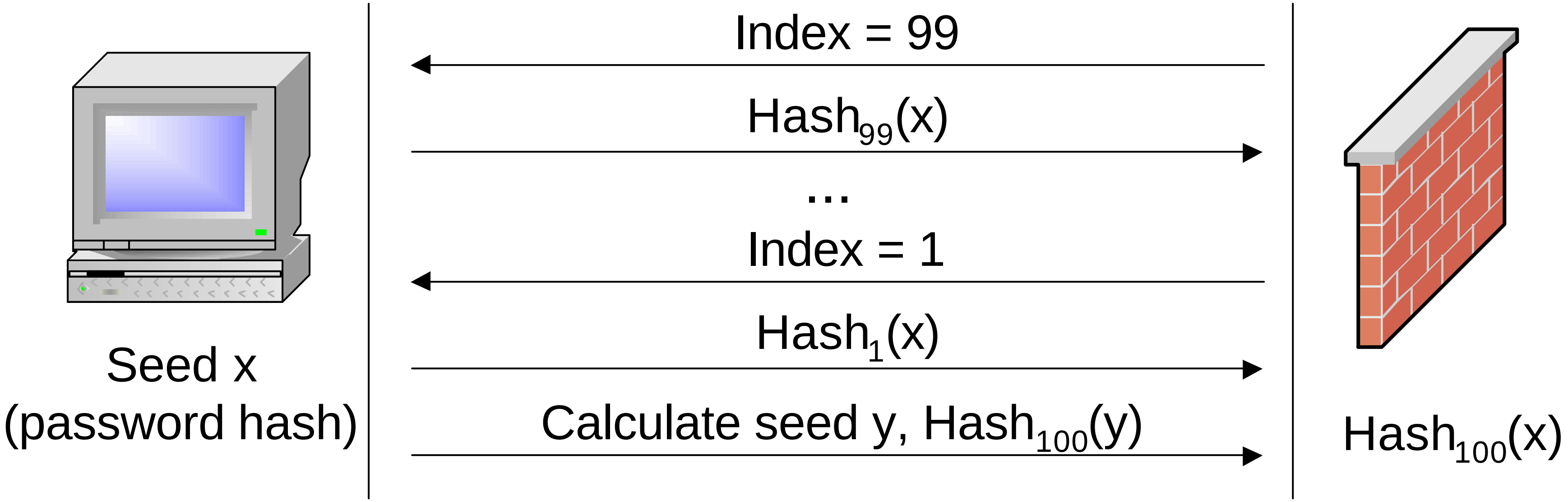
Arguments, Result

Management module

Filter module

S/Key Authentication

$$\text{Hash}_n(x) = \underbrace{\text{Hash}(\text{Hash}(\ldots\ \text{Hash}(x)))}_{\text{n times}} = \text{Hash}(\text{Hash}_{n-1}(x))$$



Index = 99

$\text{Hash}_{99}(x)$

...

Index = 1

$\text{Hash}_1(x)$

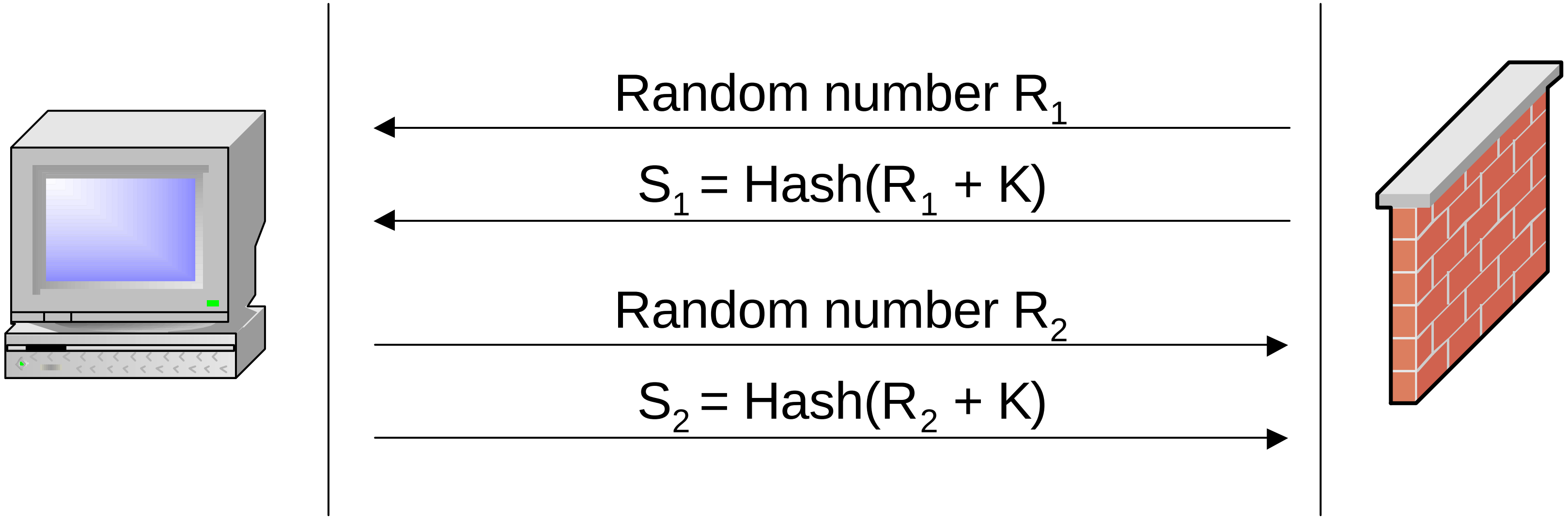
Calculate seed y, $\text{Hash}_{100}(y)$

Seed x
(password hash)

$\text{Hash}_{100}(x)$

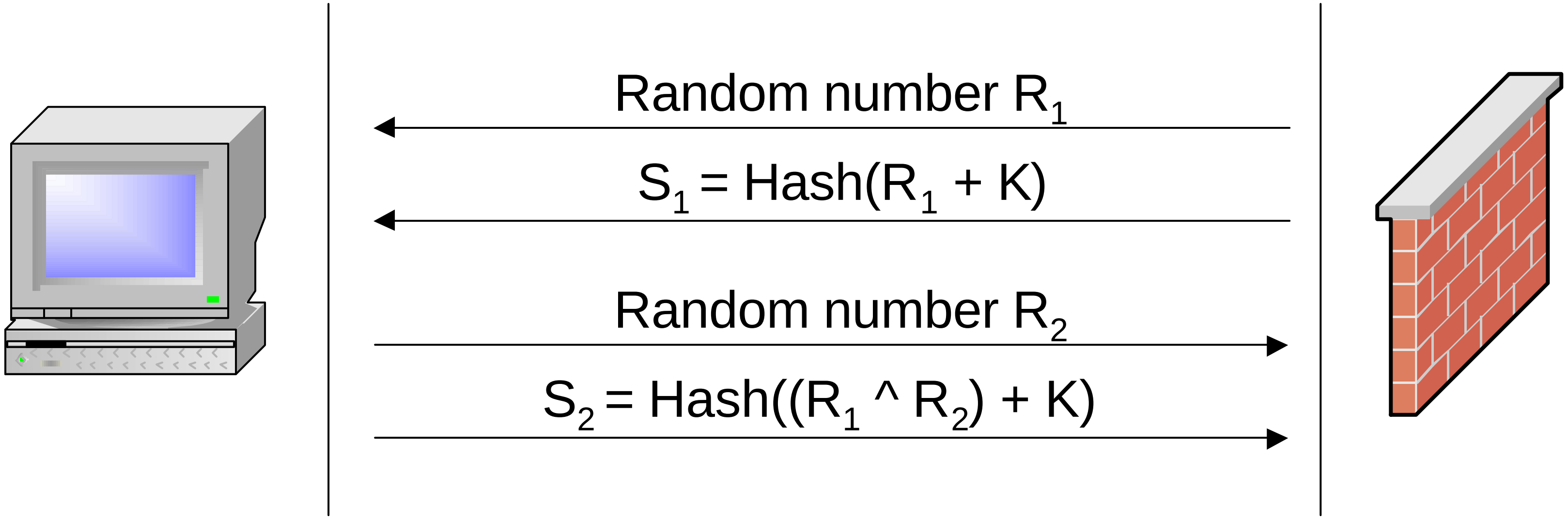
- “y = MakeSeed(time(NULL))”
- Attack: brute force

FWN1 Authentication



- Shared key *K* ("fw putkey")
- Attack: choose $R_2 = R_1$, so that $S_2 = S_1$

FWA1 Authentication



Random number R_1

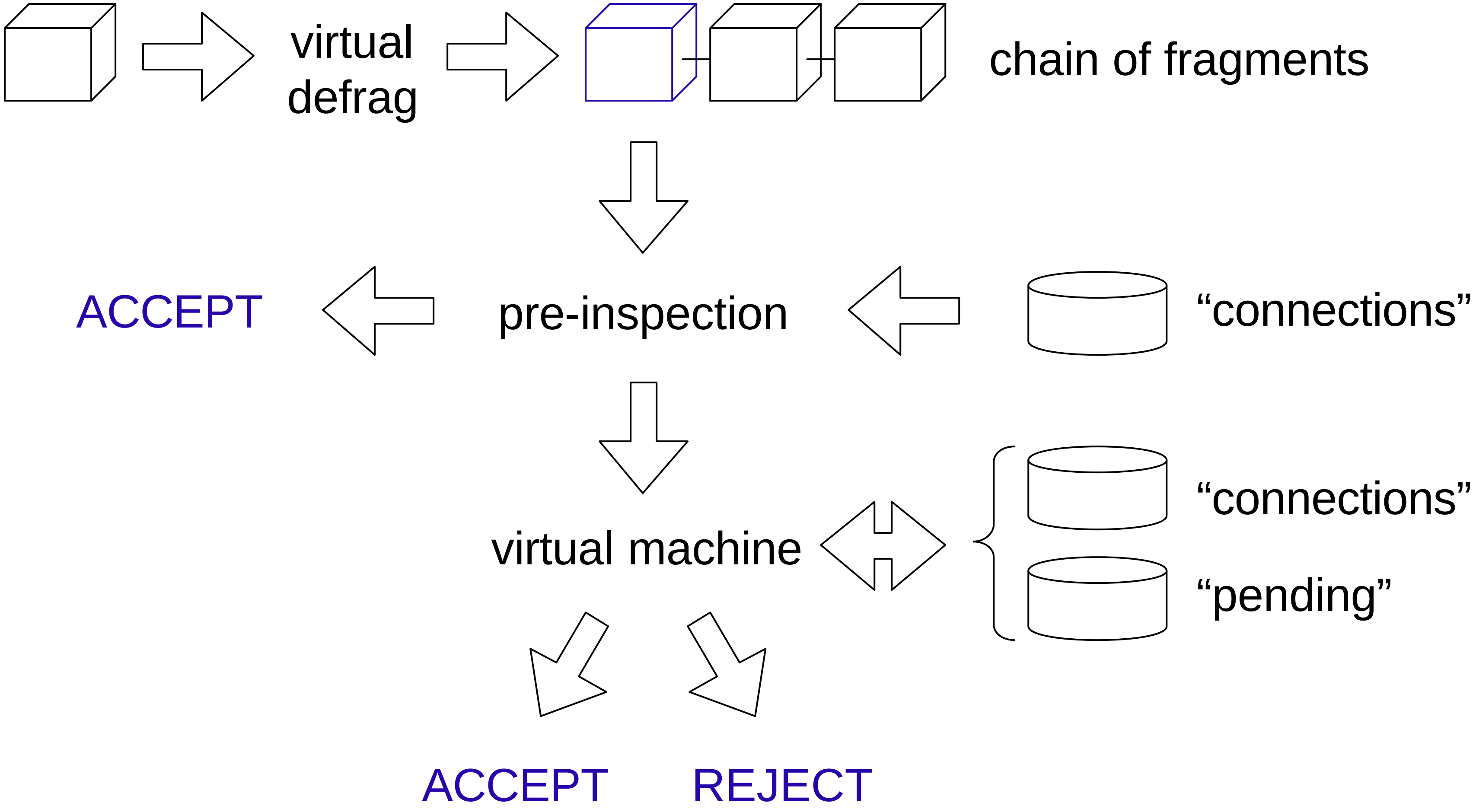
$S_1 = \text{Hash}(R_1 + K)$

Random number R_2

$S_2 = \text{Hash}((R_1 \wedge R_2) + K)$

- Shared key K (“fw putkey”)
- Attack: choose $R_2 = 0$, so that
  - $R_1 \wedge R_2 = R_1$ and
  - $S_2 = \text{Hash}((R_1 \wedge R_2) + K) = \text{Hash}(R_1 + K) = S_1$
- To be solved: encryption

Stateful Inspection I



virtual defrag

chain of fragments

ACCEPT

pre-inspection

“connections”

virtual machine

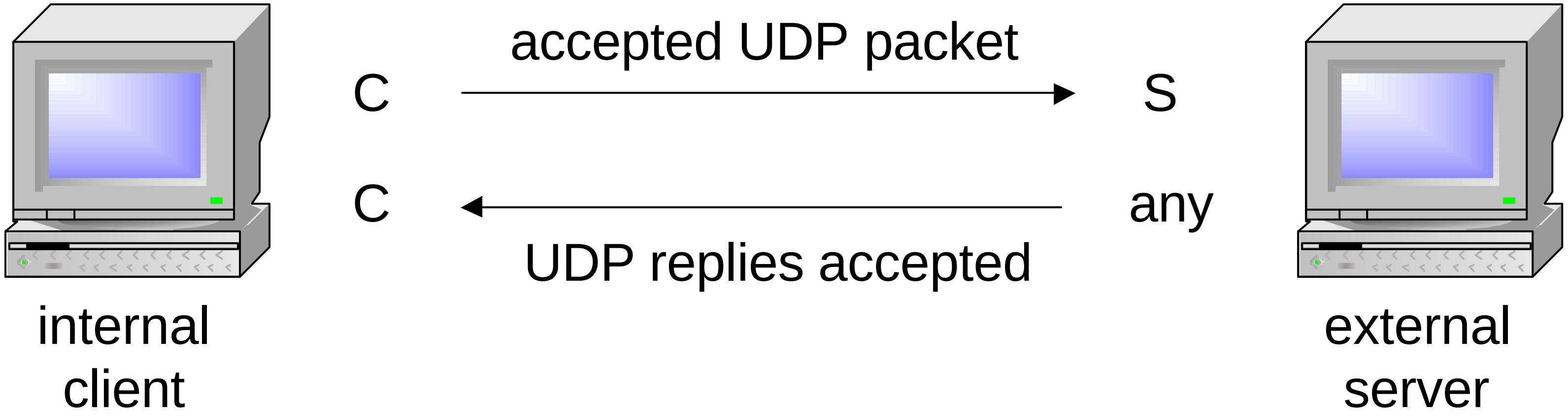
“connections”

“pending”

ACCEPT

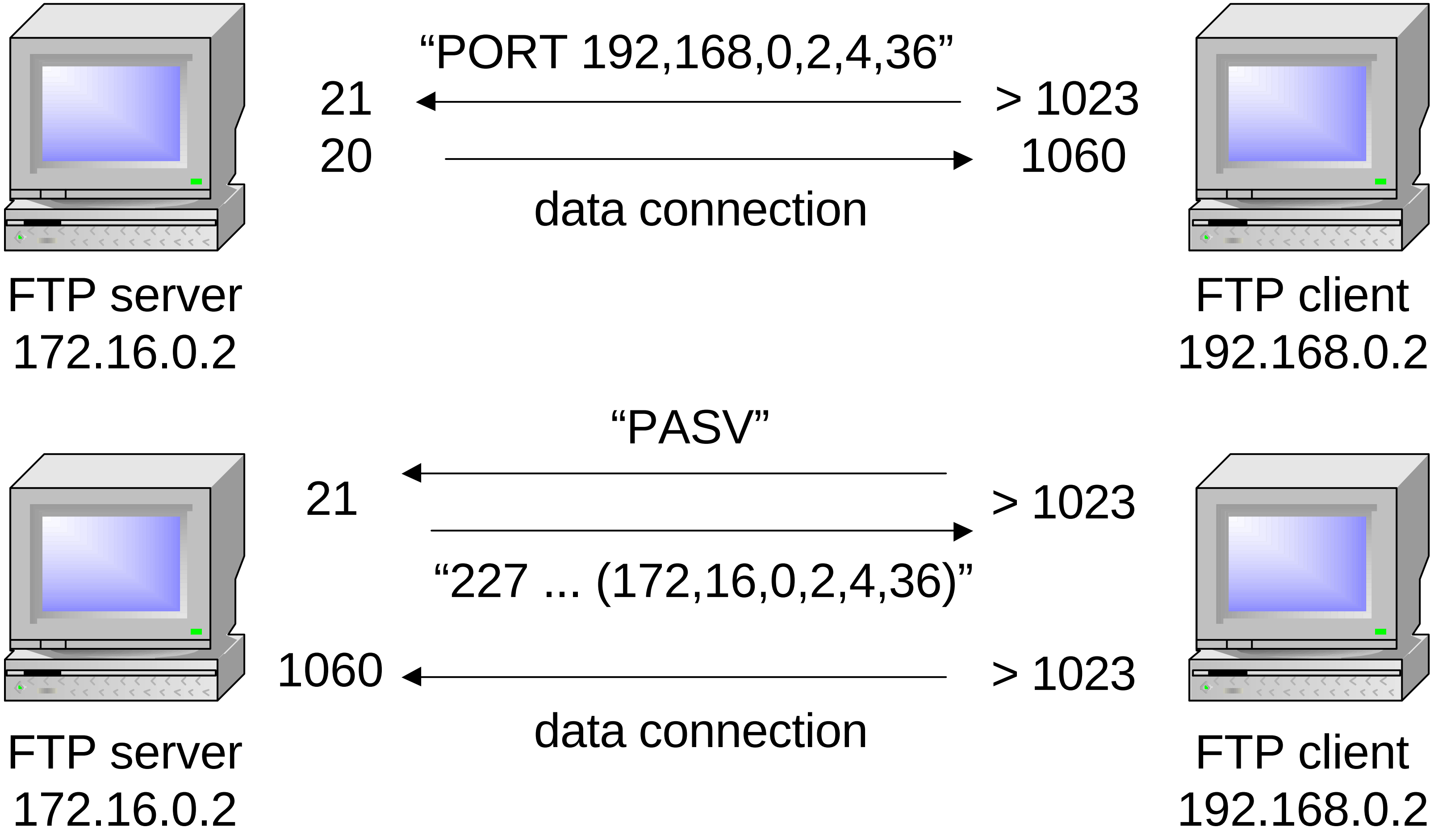
REJECT

Stateful Inspection II



- UDP “connections”
  - from a client, port C
  - to a server, port S + wildcard port
- <s-address, s-port, d-address, d-port, protocol>

Stateful Inspection III



"PORT 192,168,0,2,4,36"

21

20

> 1023

1060

data connection

FTP server
172.16.0.2

FTP client
192.168.0.2

"PASV"

21

> 1023

"227 ... (172,16,0,2,4,36)"

1060

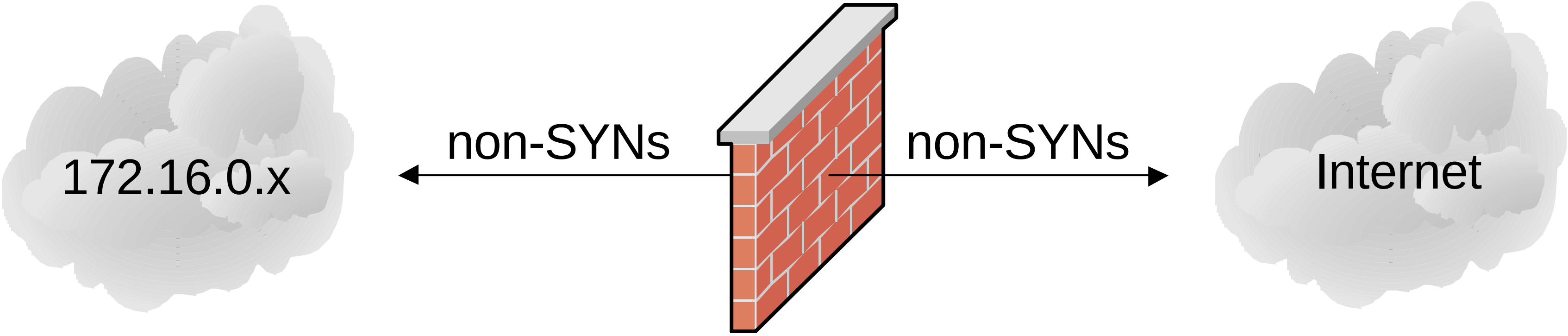
> 1023

data connection

FTP server
172.16.0.2

FTP client
192.168.0.2

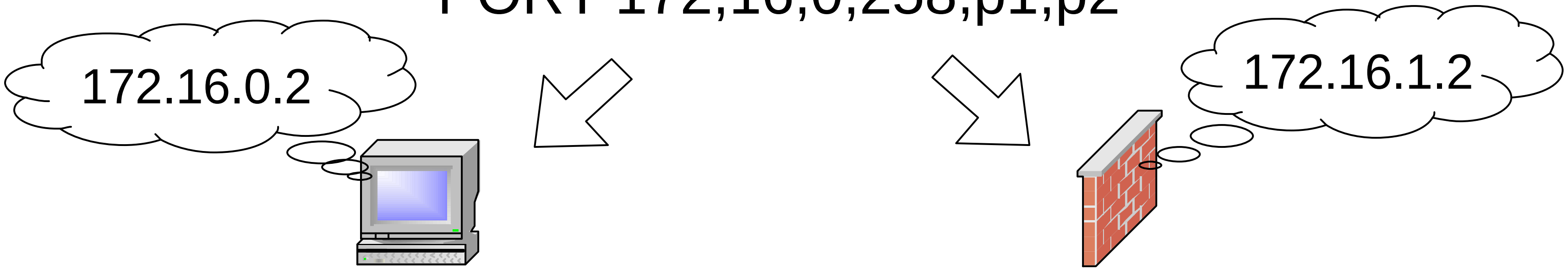
Fastmode Services



- non-SYN packets accepted
    - Source port = fastmode service
    - Destination port = fastmode service
- Stealth scanning (FINs, ...)

FTP “PORT” Parsing

“PORT 172,16,0,258,p1,p2”

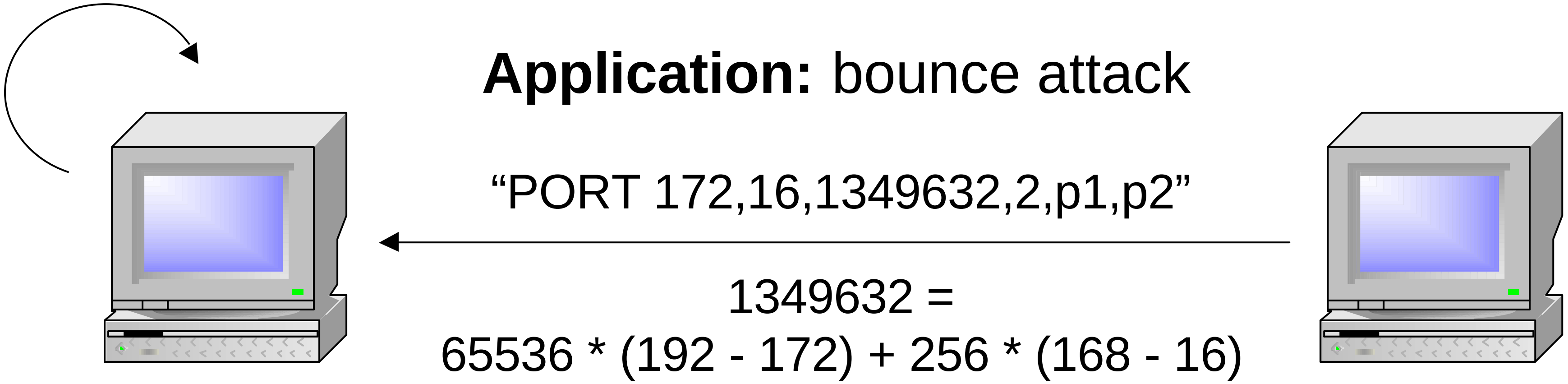


172.16.0.2

172.16.1.2

data connection

Application: bounce attack



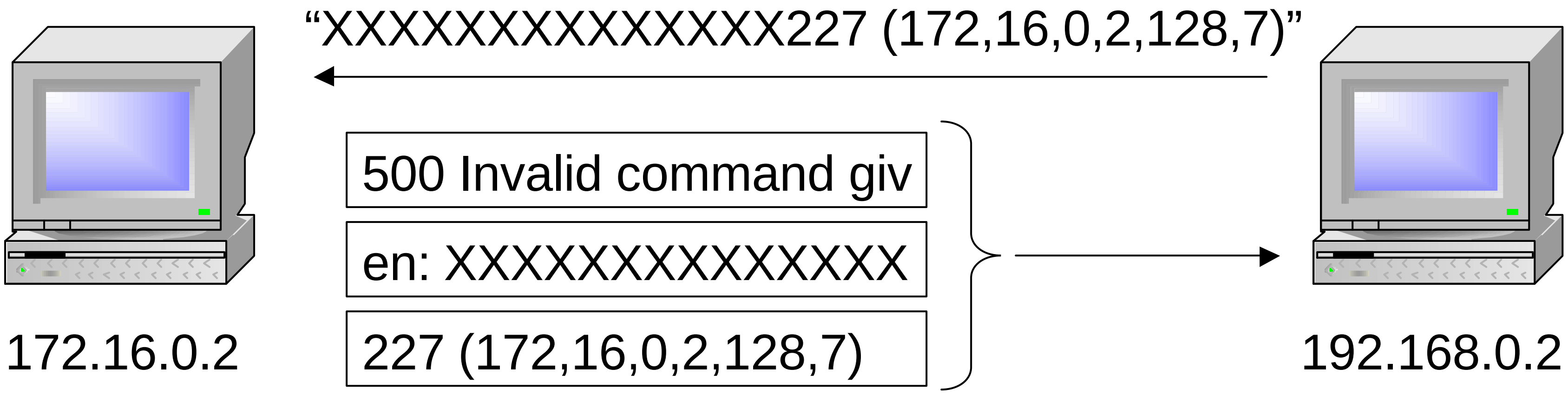
“PORT 172,16,1349632,2,p1,p2”

1349632 =
65536 * (192 - 172) + 256 * (168 - 16)

172.16.0.2

192.168.0.2

FTP “PASV” Handling



“XXXXXXXXXXXXXXX227 (172,16,0,2,128,7)”

500 Invalid command giv

en: XXXXXXXXXXXXXX

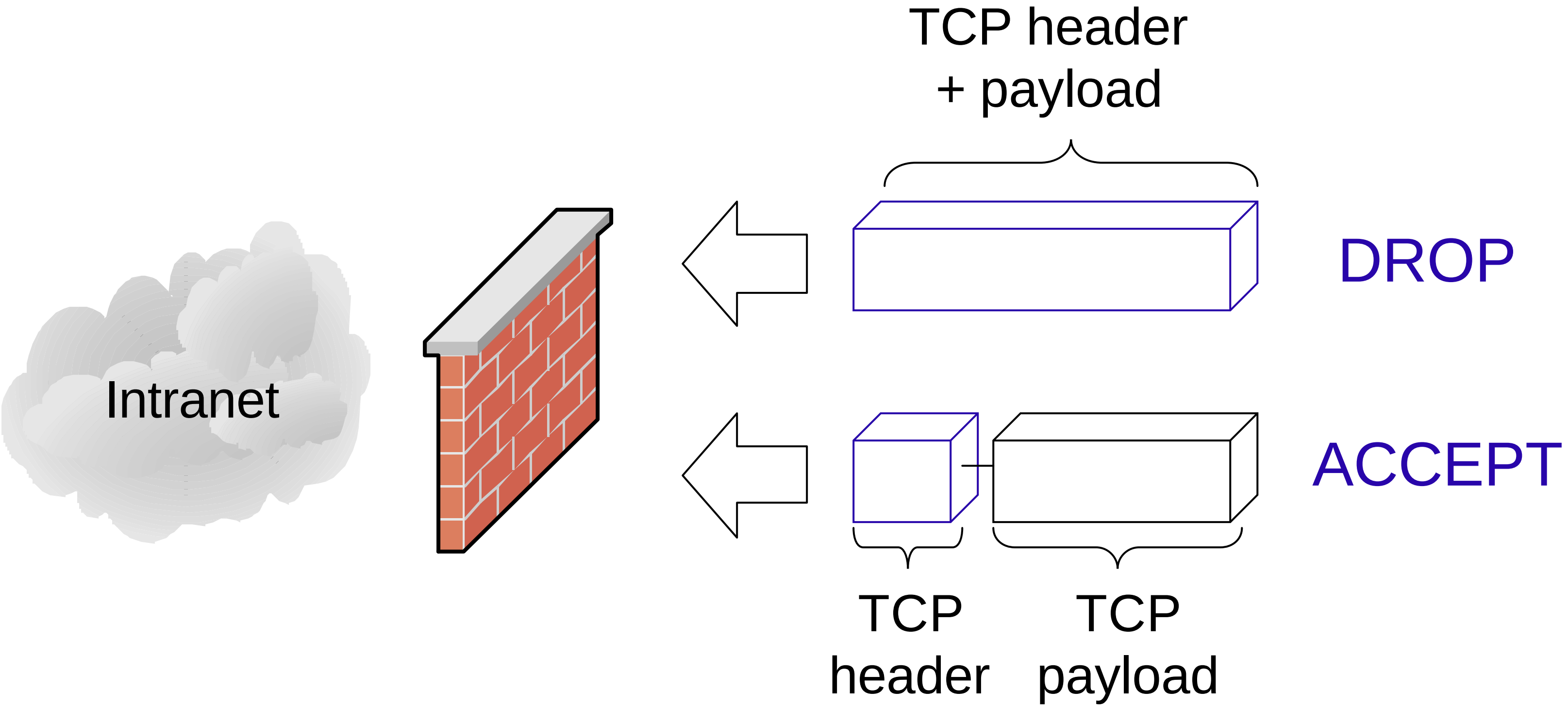
227 (172,16,0,2,128,7)

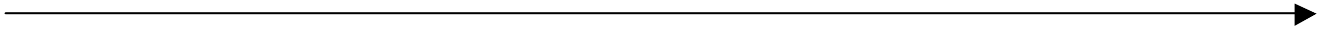
172.16.0.2

192.168.0.2

- Advertise small Maximal Segment Size
- Server replies split

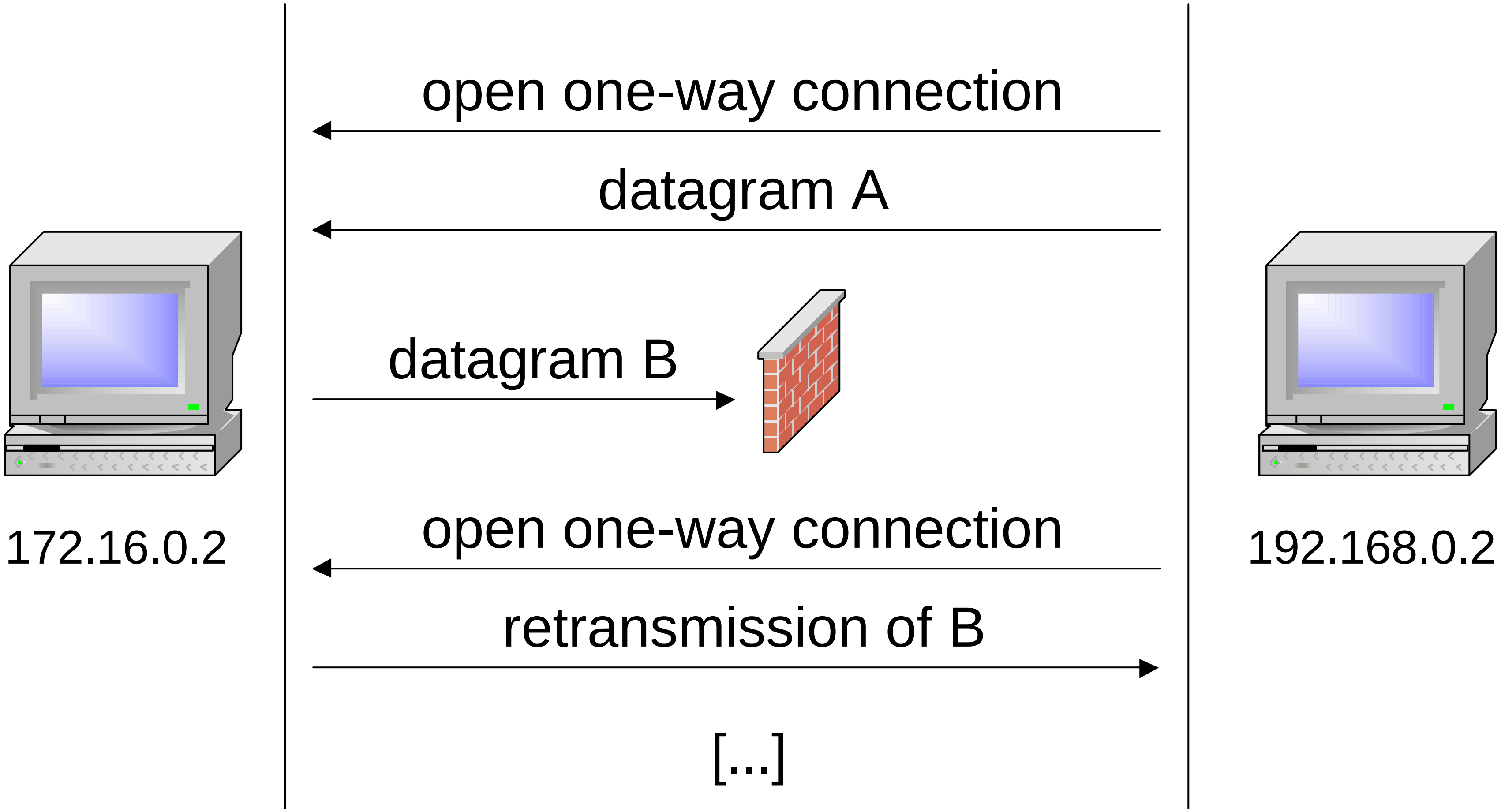
One-way Connections I





established one-way connection

One-way Connections II


open one-way connection

datagram A

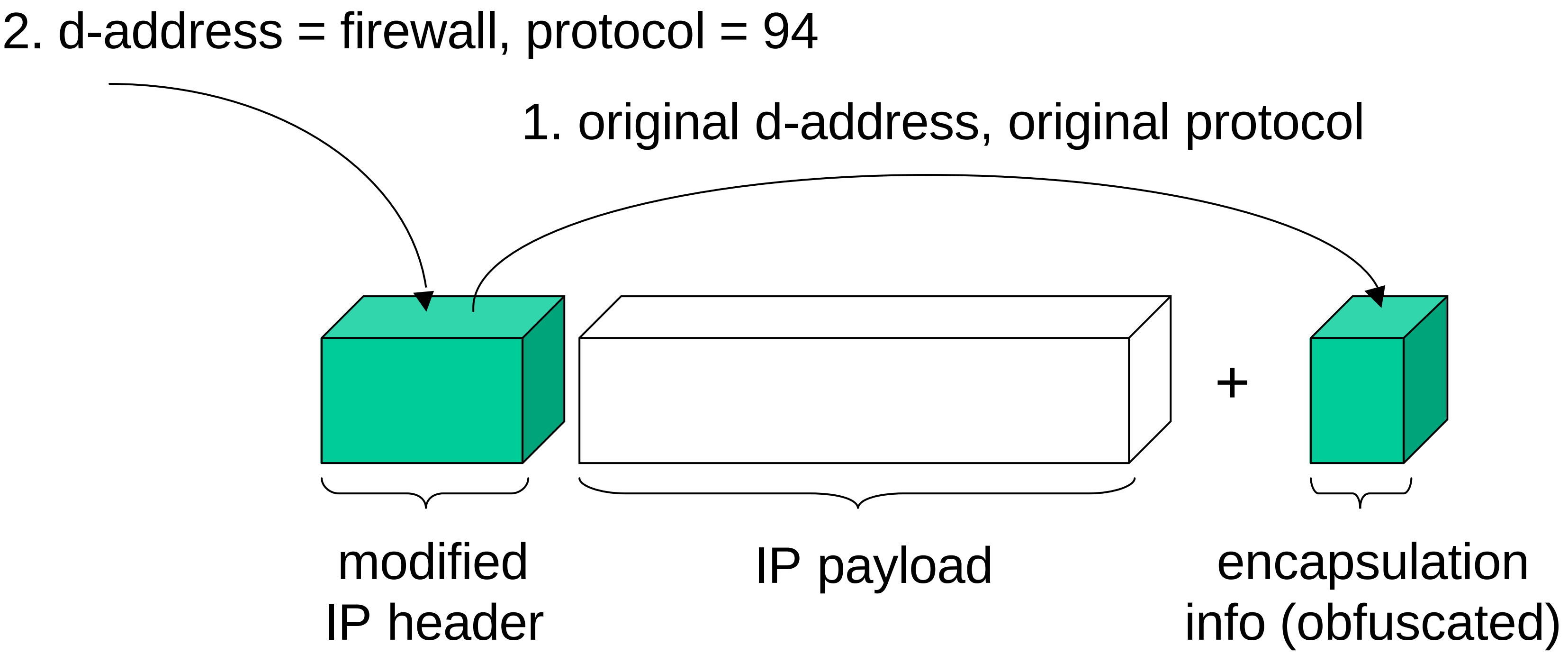
datagram B

open one-way connection

retransmission of B

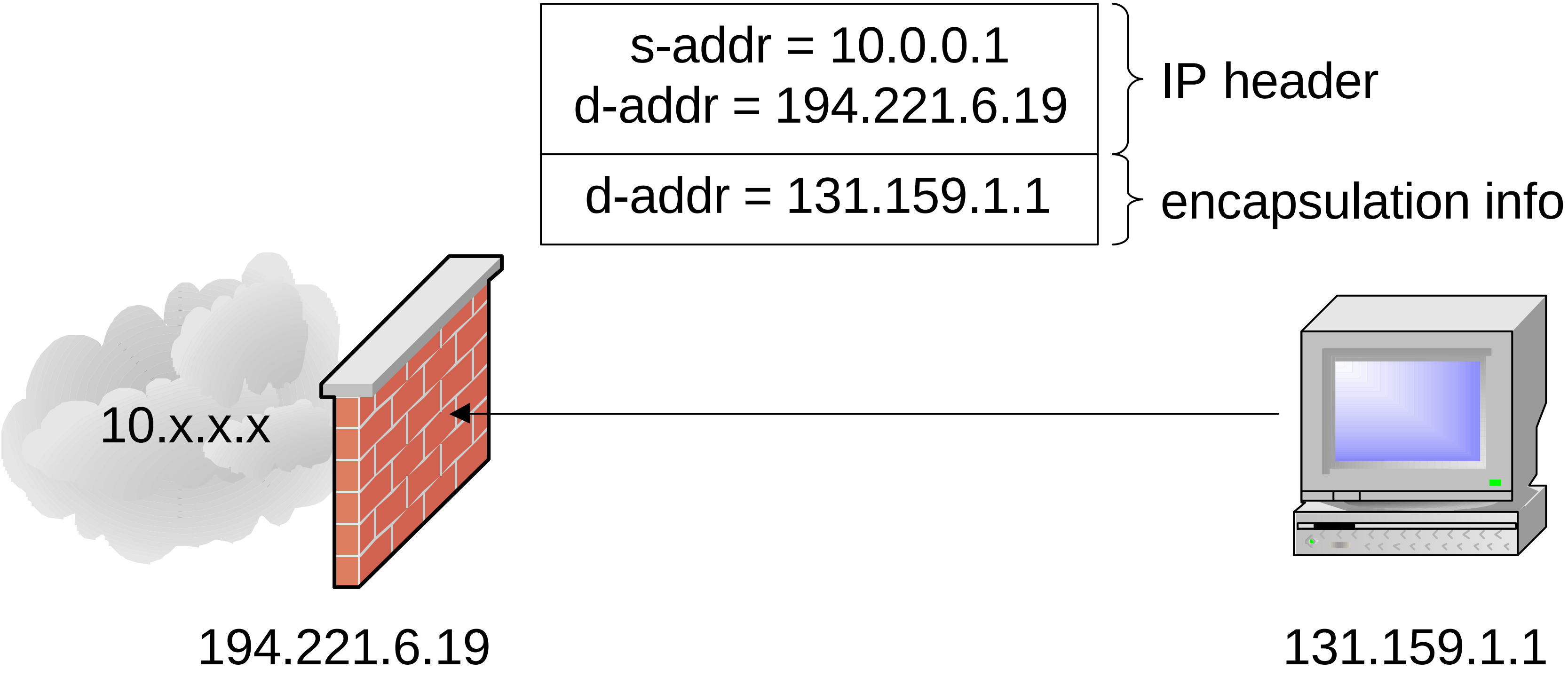
[...]

FWZ Encapsulation I


2. d-address = firewall, protocol = 94
1. original d-address, original protocol
+
modified IP header
IP payload
encapsulation info (obfuscated)

- VPN tunneling protocol
- Decapsulation without decryption or authentication
- Cannot be disabled

FWZ Encapsulation II



s-addr = 10.0.0.1
d-addr = 194.221.6.19

d-addr = 131.159.1.1

IP header

encapsulation info

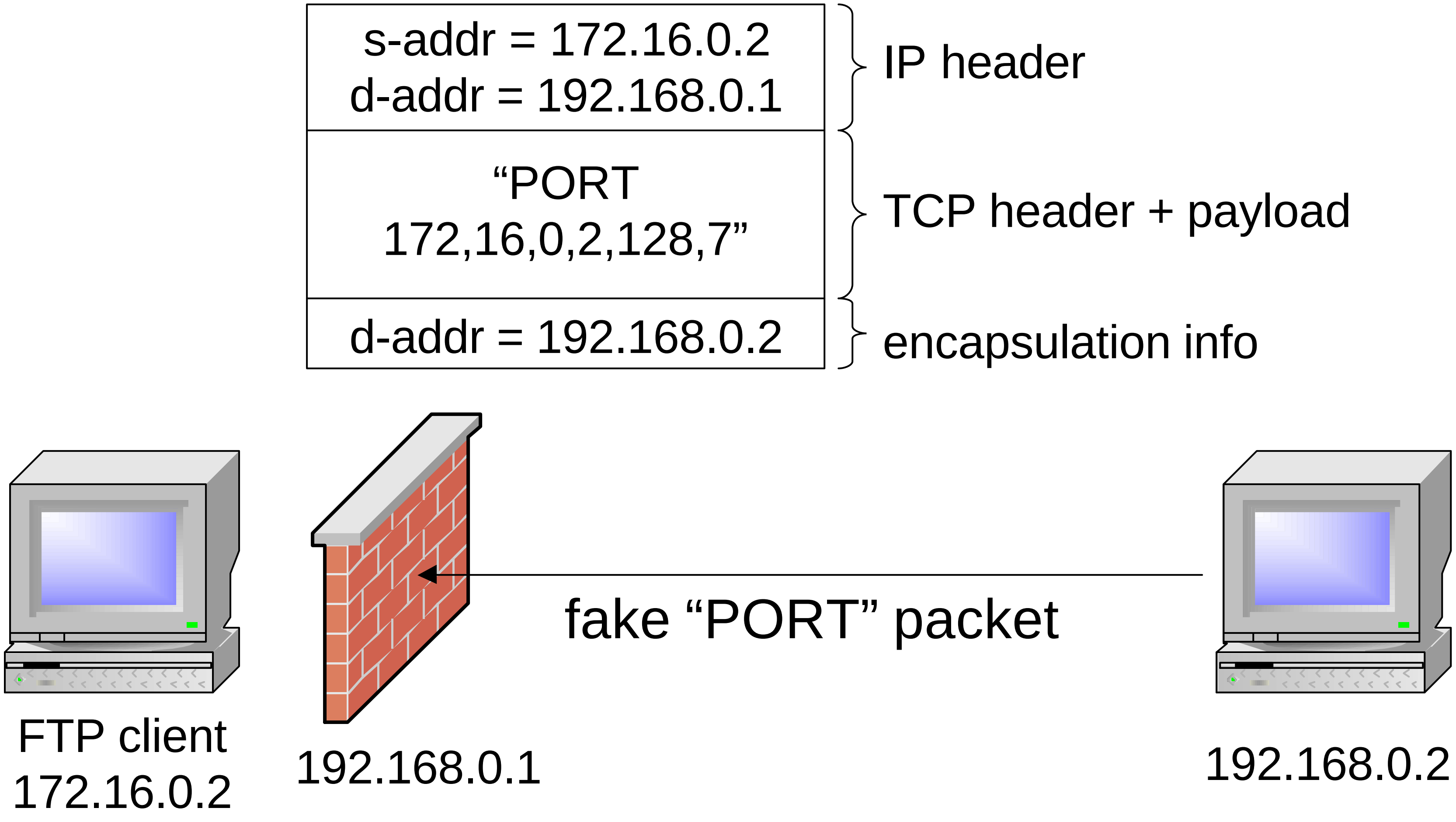
10.x.x.x

194.221.6.19

131.159.1.1

Key to spoofing attacks

Fake “PORT” Commands



s-addr = 172.16.0.2
d-addr = 192.168.0.1

“PORT
172,16,0,2,128,7”

d-addr = 192.168.0.2

IP header

TCP header + payload

encapsulation info

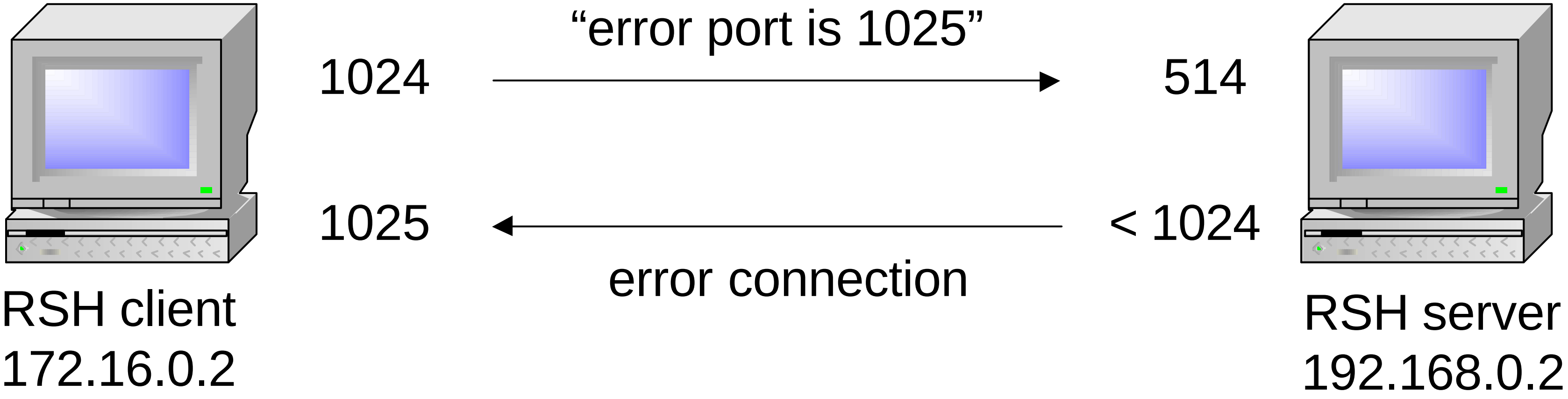
fake “PORT” packet

FTP client
172.16.0.2

192.168.0.1

192.168.0.2

RSH Error Connections I



“error port is 1025”

1024 → 514

1025 ← < 1024

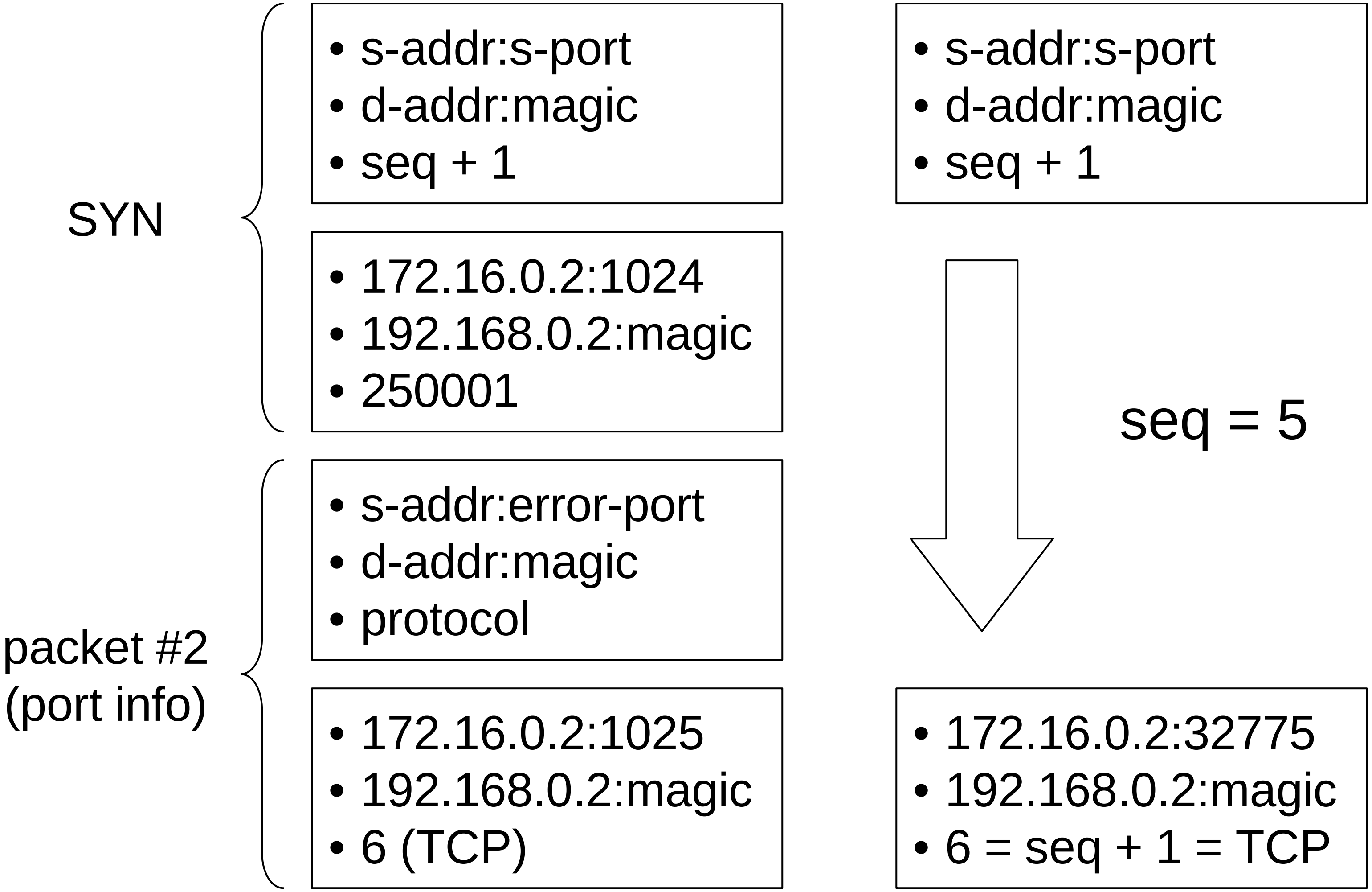
error connection

RSH client
172.16.0.2

RSH server
192.168.0.2

- <172.16.0.2, 1024, 192.168.0.2, 514, 6> in “connections”
- <172.16.0.2, 1025, 192.168.0.2, magic, 6> in “pending”
- Reversed matching

RSH Error Connections II



SYN

- s-addr:s-port
- d-addr:magic
- seq + 1

- 172.16.0.2:1024
- 192.168.0.2:magic
- 250001

packet #2
(port info)

- s-addr:error-port
- d-addr:magic
- protocol

- 172.16.0.2:1025
- 192.168.0.2:magic
- 6 (TCP)

- s-addr:s-port
- d-addr:magic
- seq + 1

seq = 5

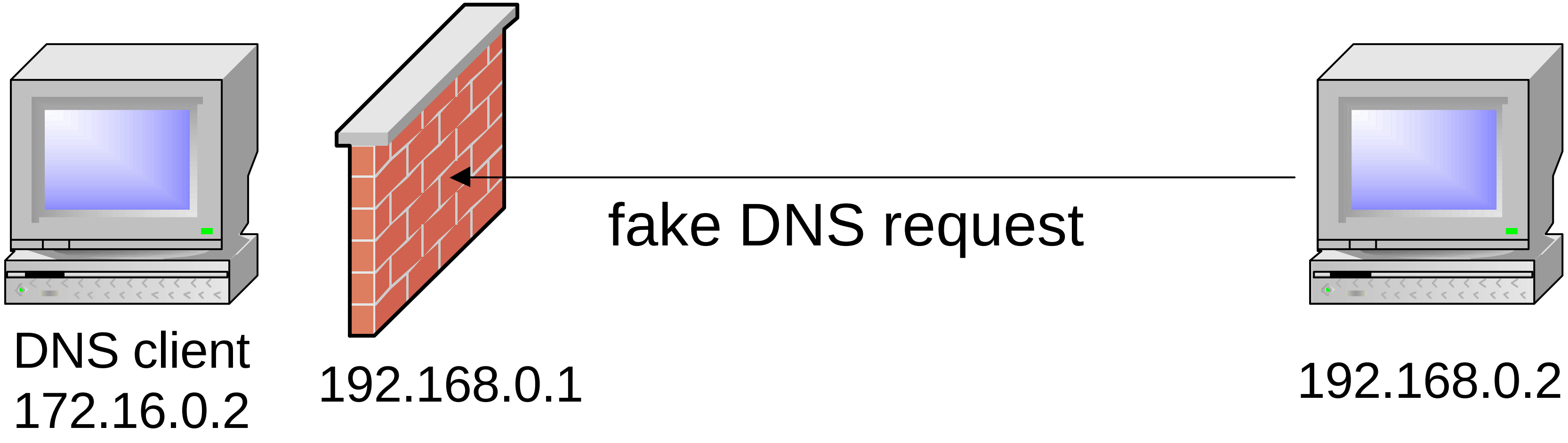
- 172.16.0.2:32775
- 192.168.0.2:magic
- 6 = seq + 1 = TCP

Fake UDP Requests

s-addr = 172.16.0.2
d-addr = 192.168.0.1
	IP header

s-port = 161
d-port = 53
	UDP header

d-addr = 192.168.0.2	encapsulation info



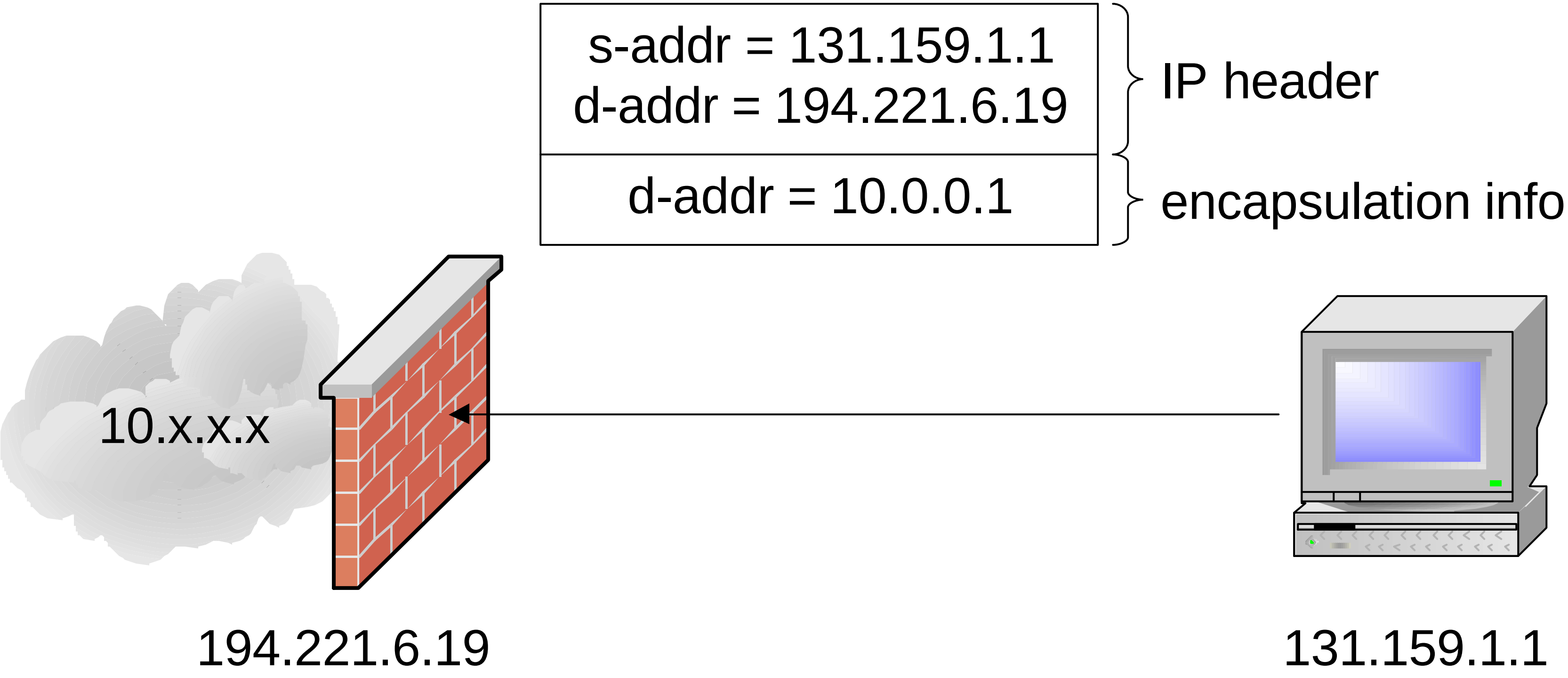
fake DNS request

DNS client
172.16.0.2

192.168.0.1

192.168.0.2

FWZ Encapsulation III



Key to non-routable addresses

Anti-Spoofing Protection I

1\.

s-addr = 192.168.0.1
d-addr = 192.168.0.1

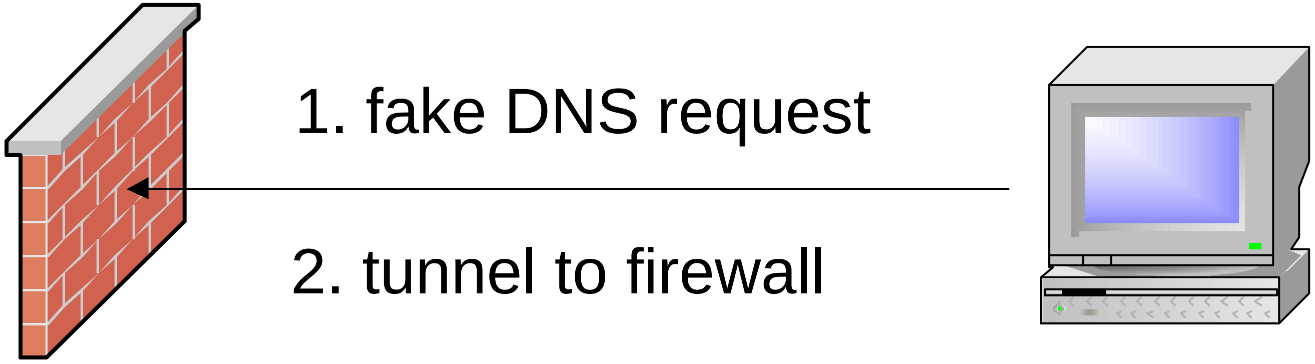
s-port = 161
d-port = 53

d-addr = 192.168.0.2

2\.

s-addr = 192.168.0.2
d-addr = 192.168.0.1

s-port = any
d-port = 161



1. fake DNS request

2. tunnel to firewall

192.168.0.1

192.168.0.2

Anti-Spoofing Protection II

1.

s-addr = 224.0.0.1
d-addr = 192.168.0.1

s-port = 161
d-port = 53

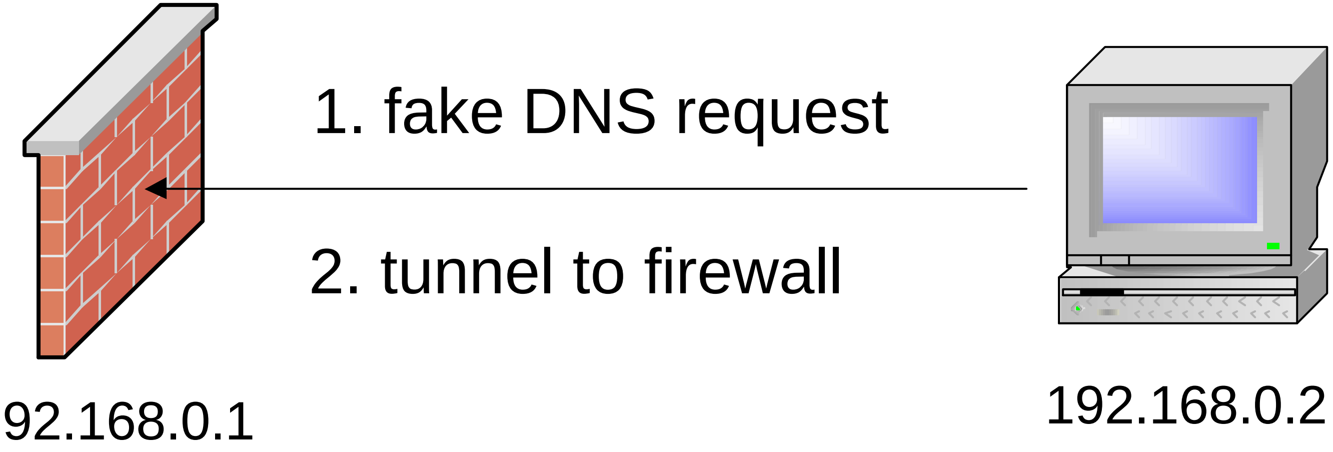
d-addr = 192.168.0.2

2.

s-addr = 192.168.0.2
d-addr = 192.168.0.1

s-port = 53
d-port = 161

d-addr = 224.0.0.1



1. fake DNS request

2. tunnel to firewall

192.168.0.1

192.168.0.2

Hardening I

- Disable implicit rules
  - DNS
  - control connections
  - ICMP
- Restrictive access rules
  - no “any” sources or destinations
  - deny broadcast / multicast addresses
  - “minimal privilege”
- Properly configure anti-spoofing mechanism
- Filter protocol 94 (e.g. IP Filter)

Hardening II

- Different (virtual) IP addresses for public services
- Restrict control connections
    - FWA1 authentication
    - VPN technology
    - **never** use “127.0.0.1: */none”
- More than one line of defense!

Fixes by Check Point

Solutions by Check Point available today at

http://www.checkpoint.com/techsupport

Thanks.

Thomas Lopatic
tl@dataprotect.com

John McDonald
jm@dataprotect.com

Dug Song
dugsong@umich.edu