OSY.SSI[2015][5]

Internet part I

Question.

Table of Contents

A high-level view

High-level experience :

A high-level view

High-level experience :

- “Connect” to a network

A high-level view

High-level experience :

- “Connect” to a network
- (wait a bit)

A high-level view

High-level experience :

- “Connect” to a network
- (wait a bit)
- Type in “`www.ecp.xxx`”

A high-level view

High-level experience :

- “Connect” to a network
- (wait a bit)
- Type in “`www.ecp.xxx`”
- (wait a bit)

A high-level view

High-level experience :

- "Connect" to a network
- (wait a bit)
- Type in "`www.ecp.xxx`"
- (wait a bit)
- Magic! Cat videos!

A high-level view

High-level experience :

- "Connect" to a network
- (wait a bit)
- Type in "`www.ecp.xxx`"
- (wait a bit)
- Magic! Cat videos!

Metaphor:

A high-level view

High-level experience :

- "Connect" to a network
- (wait a bit)
- Type in "`www.ecp.xxx`"
- (wait a bit)
- Magic! Cat videos!

Metaphor: "cyberspace"

A high-level view

High-level experience :

- "Connect" to a network
- (wait a bit)
- Type in "`www.ecp.xxx`"
- (wait a bit)
- Magic! Cat videos!

Metaphor: "cyberspace"(cf. "noosphere")

A high-level view

High-level experience :

- "Connect" to a network
- (wait a bit)
- Type in "`www.ecp.xxx`"
- (wait a bit)
- Magic! Cat videos!

Metaphor: "cyberspace"(cf. "noosphere")

Fact:

A high-level view

High-level experience :

- "Connect" to a network
- (wait a bit)
- Type in "`www.ecp.xxx`"
- (wait a bit)
- Magic! Cat videos!

Metaphor: "cyberspace"(cf. "noosphere")

Fact: Internet is *not a place*.

An intermediary-level view

Internet is

An intermediary-level view

Internet is a stack of spaghetti protocols:

An intermediary-level view

Internet is a stack of spaghetti protocols:

- http, ftp, smtp, pop, imap, Ethernet...

An intermediary-level view

Internet is a stack of spaghetti protocols:

- http, ftp, smtp, pop, imap, Ethernet...
- ssh, ldap, dhcp, dns, sip, ospf, bgp...

An intermediary-level view

Internet is a stack of spaghetti protocols:

- http, ftp, smtp, pop, imap, Ethernet...
- ssh, ldap, dhcp, dns, sip, ospf, bgp...
- tcp, udp, dccp, sctp, rsvp...

An intermediary-level view

Internet is a stack of spaghetti protocols:

- http, ftp, smtp, pop, imap, Ethernet...
- ssh, ldap, dhcp, dns, sip, ospf, bgp...
- tcp, udp, dccp, sctp, rsvp...
- ip, ipv6, icmp, icmpv6, ecn, igmp..

An intermediary-level view

Internet is a stack of spaghetti protocols:

- http, ftp, smtp, pop, imap, Ethernet...
- ssh, ldap, dhcp, dns, sip, ospf, bgp...
- tcp, udp, dccp, sctp, rsvp...
- ip, ipv6, icmp, icmpv6, ecn, igmp...
- arp, ndp, l2tp, ppp, isdn...

An intermediary-level view

Internet is a stack of spaghetti protocols:

- http, ftp, smtp, pop, imap, Ethernet...
- ssh, ldap, dhcp, dns, sip, ospf, bgp...
- tcp, udp, dccp, sctp, rsvp...
- ip, ipv6, icmp, icmpv6, ecn, igmp...
- arp, ndp, l2tp, ppp, isdn...

Plus many *proprietary and obscure* protocols: Cisco, eXtreme, Juniper, Microsoft...

An intermediary-level view

Internet is a stack of spaghetti protocols:

- http, ftp, smtp, pop, imap, Ethernet...
- ssh, ldap, dhcp, dns, sip, ospf, bgp...
- tcp, udp, dccp, sctp, rsvp...
- ip, ipv6, icmp, icmpv6, ecn, igmp...
- arp, ndp, l2tp, ppp, isdn...

Plus many *proprietary and obscure* protocols: Cisco, eXtreme, Juniper, Microsoft...

How can it even work at all?

An intermediary-level view

A nice approach by nice 1970's guys:

An intermediary-level view

A nice approach by nice 1970's guys:

- Separation of concerns

An intermediary-level view

A nice approach by nice 1970's guys:

- Separation of concerns
- Abstraction (arpanet, cyclades...)

An intermediary-level view

A nice approach by nice 1970's guys:

- Separation of concerns
- Abstraction (arpanet, cyclades...)
- Layered cake

An intermediary-level view

A nice approach by nice 1970's guys:

- Separation of concerns
- Abstraction (arpanet, cyclades...)
- Layered cake
- Open and free protocols

An intermediary-level view

A nice approach by nice 1970's guys:

- Separation of concerns
- Abstraction (arpanet, cyclades...)
- Layered cake
- Open and free protocols

ISO/IEC 7498-1 (1974–1984, 1994) – "Open Systems Interconnection" (OSI)

An intermediary-level view

A nice approach by nice 1970's guys:

- Separation of concerns
- Abstraction (arpanet, cyclades...)
- Layered cake
- Open and free protocols

ISO/IEC 7498-1 (1974–1984, 1994) – "Open Systems Interconnection" (OSI)

Then someone invented TCP/IP and all of it was forgotten.

An intermediary-level view

"Internet protocol suite"

An intermediary-level view

"Internet protocol suite" (DARPA US DoD, IETF, RFC 1122)

An intermediary-level view

"Internet protocol suite" (DARPA US DoD, IETF, RFC 1122)

- Simple (4 layers), inspired by CYCLADES

An intermediary-level view

"Internet protocol suite" (DARPA US DoD, IETF, RFC 1122)

- Simple (4 layers), inspired by CYCLADES
- Few protocols, mostly TCP/IP

An intermediary-level view

"Internet protocol suite" (DARPA US DoD, IETF, RFC 1122)

- Simple (4 layers), inspired by CYCLADES
- Few protocols, mostly TCP/IP
- Heavy promotion

An intermediary-level view

"Internet protocol suite" (DARPA US DoD, IETF, RFC 1122)

- Simple (4 layers), inspired by CYCLADES
- Few protocols, mostly TCP/IP
- Heavy promotion
- In June 1989, AT&T published the TCP/IP code developed for UNIX

An intermediary-level view

"Internet protocol suite" (DARPA US DoD, IETF, RFC 1122)

- Simple (4 layers), inspired by CYCLADES
- Few protocols, mostly TCP/IP
- Heavy promotion
- In June 1989, AT&T published the TCP/IP code developed for UNIX

This last one effectively killed all other projects (Xerox, IBM, Microsoft...)

An intermediary-level view

"Internet protocol suite" (DARPA US DoD, IETF, RFC 1122)

- Simple (4 layers), inspired by CYCLADES
- Few protocols, mostly TCP/IP
- Heavy promotion
- In June 1989, AT&T published the TCP/IP code developed for UNIX

This last one effectively killed all other projects (Xerox, IBM, Microsoft...)

Debate question:

An intermediary-level view

"Internet protocol suite" (DARPA US DoD, IETF, RFC 1122)

- Simple (4 layers), inspired by CYCLADES
- Few protocols, mostly TCP/IP
- Heavy promotion
- In June 1989, AT&T published the TCP/IP code developed for UNIX

This last one effectively killed all other projects (Xerox, IBM, Microsoft...)

Debate question: More and more communication happens only over HTTP/TCP/IP from a few hosts...

An intermediary-level view

"Internet protocol suite" (DARPA US DoD, IETF, RFC 1122)

- Simple (4 layers), inspired by CYCLADES
- Few protocols, mostly TCP/IP
- Heavy promotion
- In June 1989, AT&T published the TCP/IP code developed for UNIX

This last one effectively killed all other projects (Xerox, IBM, Microsoft...)

Debate question: More and more communication happens only over HTTP/TCP/IP from a few hosts...($\Rightarrow$ back to the Minitel?)

An intermediary-level view

TCP/IP stack:

An intermediary-level view

TCP/IP stack:

- **Link:**

An intermediary-level view

TCP/IP stack:

- **Link:** local area network

An intermediary-level view

TCP/IP stack:

- **Link:** local area network
- **Internet:**

An intermediary-level view

TCP/IP stack:

- **Link:** local area network
- **Internet:** host addressing, identification, packet routing

An intermediary-level view

TCP/IP stack:

- **Link:** local area network
- **Internet:** host addressing, identification, packet routing
- **Transport:**

An intermediary-level view

TCP/IP stack:

- **Link:** local area network
- **Internet:** host addressing, identification, packet routing
- **Transport:** process-to-process connectivity, control, application addressing

An intermediary-level view

TCP/IP stack:

- **Link:** local area network
- **Internet:** host addressing, identification, packet routing
- **Transport:** process-to-process connectivity, control, application addressing
- **Application:**

An intermediary-level view

TCP/IP stack:

- **Link:** local area network
- **Internet:** host addressing, identification, packet routing
- **Transport:** process-to-process connectivity, control, application addressing
- **Application:** application-to-application, user contact

An intermediary-level view

TCP/IP stack:

- **Link:** local area network
- **Internet:** host addressing, identification, packet routing
- **Transport:** process-to-process connectivity, control, application addressing
- **Application:** application-to-application, user contact

Example:

An intermediary-level view

TCP/IP stack:

- **Link:** local area network
- **Internet:** host addressing, identification, packet routing
- **Transport:** process-to-process connectivity, control, application addressing
- **Application:** application-to-application, user contact

Example: HTTP/TCP/IP/Ethernet.

An intermediary-level view

TCP/IP stack:

- **Link:** local area network
- **Internet:** host addressing, identification, packet routing
- **Transport:** process-to-process connectivity, control, application addressing
- **Application:** application-to-application, user contact

Example: HTTP/TCP/IP/Ethernet.

Note:

An intermediary-level view

TCP/IP stack:

- **Link:** local area network
- **Internet:** host addressing, identification, packet routing
- **Transport:** process-to-process connectivity, control, application addressing
- **Application:** application-to-application, user contact

Example: HTTP/TCP/IP/Ethernet.

Note: Application-to-Application communication.

An intermediary-level view

TCP/IP stack:

- **Link:** local area network
- **Internet:** host addressing, identification, packet routing
- **Transport:** process-to-process connectivity, control, application addressing
- **Application:** application-to-application, user contact

Example: HTTP/TCP/IP/Ethernet.

Note: Application-to-Application communication.

But there's only one cable! How works!?

A low-level view

Wow. Such networks. Much protocol.

Protocols are *encapsulated*:

A low-level view

Wow. Such networks. Much protocol.

Protocols are *encapsulated*:

A low-level view

Wow. Such networks. Much protocol.

Protocols are *encapsulated*:

Network equipment may only peel a few layers.

A low-level view

Wow. Such networks. Much protocol.

Protocols are *encapsulated*:

Network equipment may only peel a few layers.

A switch only has to consider:

A low-level view

Wow. Such networks. Much protocol.

Protocols are *encapsulated*:

Network equipment may only peel a few layers.

A switch only has to consider:

A low-level view

Can we see that in action?

A low-level view

Can we see that in action?Yes!

A low-level view

Can we see that in action?Yes!

Demo:

A low-level view

Can we see that in action?Yes!

Demo: `tcpdump`, `wireshark`, or Network Lab (next time)!

That's not enough...

For our purpose, we actually need a few more things on:

That's not enough...

For our purpose, we actually need a few more things on:

- Addressing (link, IP and domain name)

That's not enough...

For our purpose, we actually need a few more things on:

- Addressing (link, IP and domain name)
- TCP (handshake, transport...)

That's not enough...

For our purpose, we actually need a few more things on:

- Addressing (link, IP and domain name)
- TCP (handshake, transport...)
- Management protocols (NTP, ARP, etc.)

That's not enough...

For our purpose, we actually need a few more things on:

- Addressing (link, IP and domain name)
- TCP (handshake, transport...)
- Management protocols (NTP, ARP, etc.)

Spoiler:

That's not enough...

For our purpose, we actually need a few more things on:

- Addressing (link, IP and domain name)
- TCP (handshake, transport...)
- Management protocols (NTP, ARP, etc.)

Spoiler: all of this was devised against $\perp$.

Table of Contents

Link addressing

Question:

Link addressing

Question: How does the link layer manages addressing?

Link addressing

Question: How does the link layer manages addressing?

Answer: It does not.

Link addressing

Question: How does the link layer manages addressing?

Answer: It does not.

No CIA guarantee at the link level

Link addressing

Question: How does the link layer manages addressing?

Answer: It does not.

No CIA guarantee at the link level

Demo:

Link addressing

Question: How does the link layer manages addressing?

Answer: It does not.

No CIA guarantee at the link level

Demo: `mode promiscuous/monitor`, `airodump-ng`

Link addressing

Question: How does the link layer manages addressing?

Answer: It does not.

No CIA guarantee at the link level

Demo: `mode promiscuous/monitor`, `airodump-ng`

Note:

Link addressing

Question: How does the link layer manages addressing?

Answer: It does not.

No CIA guarantee at the link level

Demo: `mode promiscuous/monitor`, `airodump-ng`

Note: caffe-latte attack.

Internet addressing

Insight:

Internet addressing

Insight: It would be inefficient to transmit *all data to everyone*.

Internet addressing

Insight: It would be inefficient to transmit *all data to everyone*. We should only send it to its destination.

Internet addressing

Insight: It would be inefficient to transmit *all data to everyone*. We should only send it to its destination.

But how do we know where it is?

Internet addressing

Insight: It would be inefficient to transmit *all data to everyone*. We should only send it to its destination.

But how do we know where it is?

Solution:

Internet addressing

Insight: It would be inefficient to transmit *all data to everyone*. We should only send it to its destination.

But how do we know where it is?

Solution: hierarchical addresses + routing

Internet addressing

Hierarchical addressing

32 bit addresses = Four 8-bit bytes (little endian):

Internet addressing

Hierarchical addressing

32 bit addresses = Four 8-bit bytes (little endian):

`138.195.42.69`

Internet addressing

Hierarchical addressing

32 bit addresses = Four 8-bit bytes (little endian):

`138.195.42.69`

IP address ranges : `xxx.xxx.xxx.0` to `xxx.xxx.xxx.255`

Internet addressing

Hierarchical addressing

32 bit addresses = Four 8-bit bytes (little endian):

`138.195.42.69`

IP address ranges : `xxx.xxx.xxx.0` to `xxx.xxx.xxx.255`

`138.195.42.0/24`

(“fixed 24 first bits”)

Internet addressing

Hierarchical addressing

Huge inequalities in IP address estates...

Internet addressing

Hierarchical addressing

Huge inequalities in IP address estates...

- USA biggest owner

Internet addressing

Hierarchical addressing

Huge inequalities in IP address estates...

- USA biggest owner
- Some countries have 1 IP address!

Internet addressing

Hierarchical addressing

Huge inequalities in IP address estates...

- USA biggest owner
- Some countries have 1 IP address!

Workarounds:

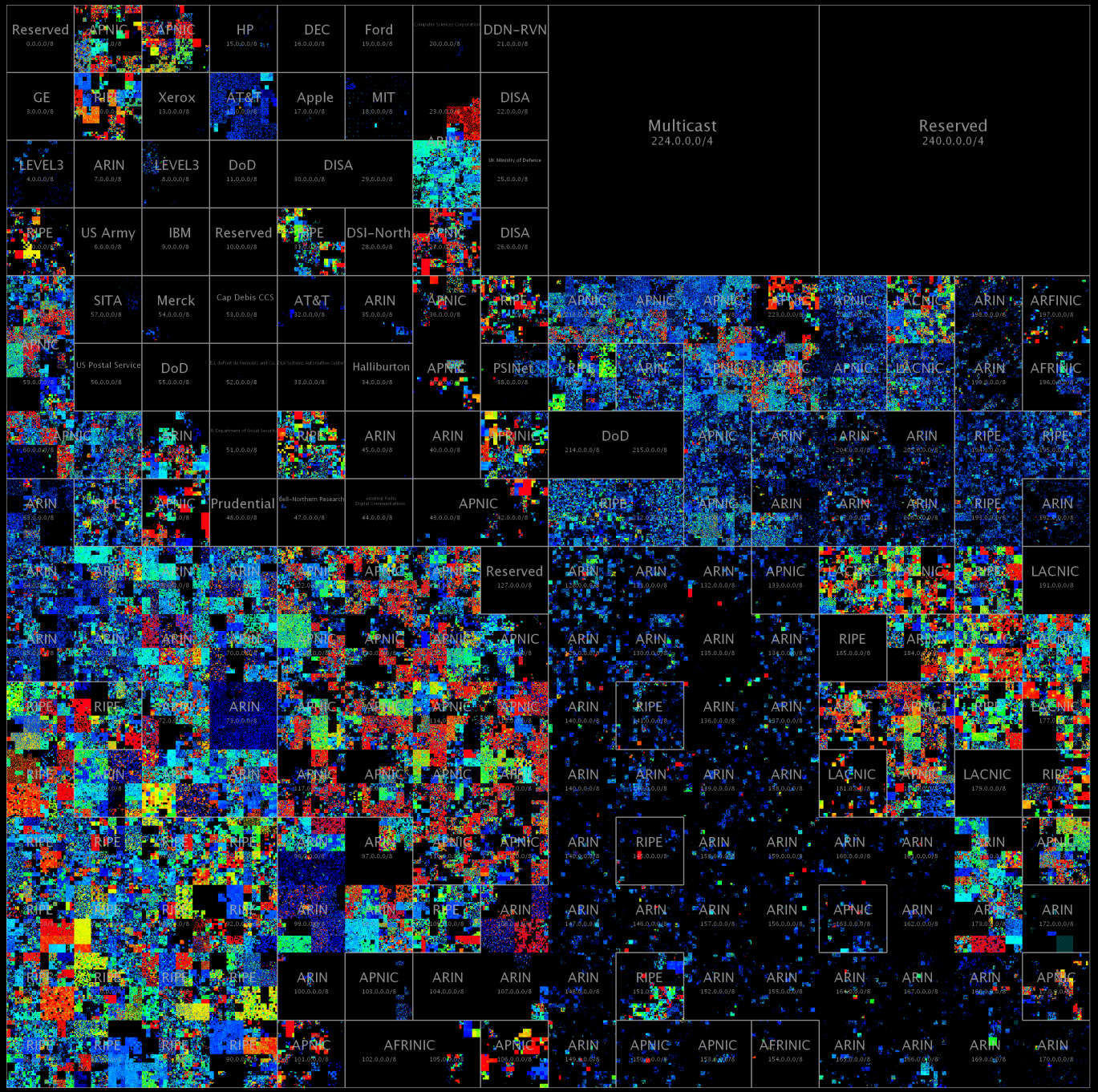
Internet addressing

Hierarchical addressing

Huge inequalities in IP address estates...

- USA biggest owner
- Some countries have 1 IP address!

Workarounds: NAT, encapsulation, IPv6...



IPv4 Census Map

June – October 2012

Utilization

100%
90%
80%
70%
60%
50%
40%
30%
20%
10%
0%

Prefix Sizes

= /8

= /12

= /16

= /20

= /24

420 Million hosts that responded to ICMP Ping at least 2 times between June and October 2012
Source: Carna Botnet

Internet addressing

Routing

Routing...

Internet addressing

Routing

Routing...

- The task of a *router*.

Internet addressing

Routing

Routing...

- The task of a *router*.
- Routing tables can be hardwired or learnt (OSPF, BGP...)

Internet addressing

Routing

Routing...

- The task of a *router*.
- Routing tables can be hardwired or learnt (OSPF, BGP...)
- Not immune to tampering (Pakistan vs. YouTube, China vs. the World...)

Internet addressing

Routing

Routing...

- The task of a *router*.
- Routing tables can be hardwired or learnt (OSPF, BGP...)
- Not immune to tampering (Pakistan vs. YouTube, China vs. the World...)
- Relies in the end on the BackBone's authority (7 people).

Internet addressing

Routing

Routing...

- The task of a *router*.
- Routing tables can be hardwired or learnt (OSPF, BGP...)
- Not immune to tampering (Pakistan vs. YouTube, China vs. the World...)
- Relies in the end on the BackBone's authority (7 people).

Take over the BackBone, take over the Internet.

Internet addressing

Routing

Routing...

- The task of a *router*.
- Routing tables can be hardwired or learnt (OSPF, BGP...)
- Not immune to tampering (Pakistan vs. YouTube, China vs. the World...)
- Relies in the end on the BackBone's authority (7 people).

Take over the BackBone, take over the Internet.

Demo:

Internet addressing

Routing

Routing...

- The task of a *router*.
- Routing tables can be hardwired or learnt (OSPF, BGP...)
- Not immune to tampering (Pakistan vs. YouTube, China vs. the World...)
- Relies in the end on the BackBone's authority (7 people).

Take over the BackBone, take over the Internet.

Demo: `route -nev`

Internet addressing

The missing links

I'm Bob. I want to send Alice a message.

Internet addressing

The missing links

I'm Bob. I want to send Alice a message.

- Figure out Alice's IP address `1.2.3.4` (**How???**)

Internet addressing

The missing links

I'm Bob. I want to send Alice a message.

- Figure out Alice's IP address `1.2.3.4` (**How???**)
- Send an IP packet with `destination = 1.2.3.4` to my local switch

Internet addressing

The missing links

I'm Bob. I want to send Alice a message.

- Figure out Alice's IP address `1.2.3.4` (**How???**)
- Send an IP packet with `destination = 1.2.3.4` to my local switch
- The switch figures it's not a local address, sends it to its local router

Internet addressing

The missing links

I'm Bob. I want to send Alice a message.

- Figure out Alice's IP address `1.2.3.4` (**How???**)
- Send an IP packet with `destination = 1.2.3.4` to my local switch
- The switch figures it's not a local address, sends it to its local router
- The router sends it to his friend router in Bangladesh

Internet addressing

The missing links

I'm Bob. I want to send Alice a message.

- Figure out Alice's IP address `1.2.3.4` (**How???**)
- Send an IP packet with `destination = 1.2.3.4` to my local switch
- The switch figures it's not a local address, sends it to its local router
- The router sends it to his friend router in Bangladesh
- The message hops from place to place (hopefully not too much)

Internet addressing

The missing links

I'm Bob. I want to send Alice a message.

- Figure out Alice's IP address `1.2.3.4` (**How???**)
- Send an IP packet with `destination = 1.2.3.4` to my local switch
- The switch figures it's not a local address, sends it to its local router
- The router sends it to his friend router in Bangladesh
- The message hops from place to place (hopefully not too much)
- The message arrives to the router in charge of `1.2.3.0/24`

Internet addressing

The missing links

I'm Bob. I want to send Alice a message.

- Figure out Alice's IP address `1.2.3.4` (**How???**)
- Send an IP packet with `destination = 1.2.3.4` to my local switch
- The switch figures it's not a local address, sends it to its local router
- The router sends it to his friend router in Bangladesh
- The message hops from place to place (hopefully not too much)
- The message arrives to the router in charge of `1.2.3.0/24`
- The router sends it to the appropriate LAN

Internet addressing

The missing links

I'm Bob. I want to send Alice a message.

- Figure out Alice's IP address `1.2.3.4` (**How???**)
- Send an IP packet with `destination = 1.2.3.4` to my local switch
- The switch figures it's not a local address, sends it to its local router
- The router sends it to his friend router in Bangladesh
- The message hops from place to place (hopefully not too much)
- The message arrives to the router in charge of `1.2.3.0/24`
- The router sends it to the appropriate LAN
- The switch figures out which cable to use for Alice (**How???**)

Internet addressing

The missing links

I'm Bob. I want to send Alice a message.

- Figure out Alice's IP address `1.2.3.4` (**How???**)
- Send an IP packet with `destination = 1.2.3.4` to my local switch
- The switch figures it's not a local address, sends it to its local router
- The router sends it to his friend router in Bangladesh
- The message hops from place to place (hopefully not too much)
- The message arrives to the router in charge of `1.2.3.0/24`
- The router sends it to the appropriate LAN
- The switch figures out which cable to use for Alice (**How???**)

The missing links are

Internet addressing

The missing links

I'm Bob. I want to send Alice a message.

- Figure out Alice's IP address `1.2.3.4` (**How???**)
- Send an IP packet with `destination = 1.2.3.4` to my local switch
- The switch figures it's not a local address, sends it to its local router
- The router sends it to his friend router in Bangladesh
- The message hops from place to place (hopefully not too much)
- The message arrives to the router in charge of `1.2.3.0/24`
- The router sends it to the appropriate LAN
- The switch figures out which cable to use for Alice (**How???**)

The missing links are *name resolution* and *address resolution*.

Name resolution

I give you “Alice”, give me `1.2.3.4`

Mostly the work of the *Domain Name Service* (DNS) protocol.

Name resolution

I give you “Alice”, give me `1.2.3.4`

Mostly the work of the *Domain Name Service* (DNS) protocol.

Demo:

Name resolution

I give you “Alice”, give me `1.2.3.4`

Mostly the work of the *Domain Name Service* (DNS) protocol.

Demo: `dig`, `host`, `dig -x`

Name resolution

I give you “Alice”, give me `1.2.3.4`

Mostly the work of the *Domain Name Service* (DNS) protocol.

Demo: `dig`, `host`, `dig -x`

Who answers?

Name resolution

I give you “Alice”, give me `1.2.3.4`

Mostly the work of the *Domain Name Service* (DNS) protocol.

Demo: `dig`, `host`, `dig -x`

Who answers?

- My browser

Name resolution

I give you “Alice”, give me `1.2.3.4`

Mostly the work of the *Domain Name Service* (DNS) protocol.

Demo: `dig`, `host`, `dig -x`

Who answers?

- My browser
- My computer

Name resolution

I give you “Alice”, give me `1.2.3.4`

Mostly the work of the *Domain Name Service* (DNS) protocol.

Demo: `dig`, `host`, `dig -x`

Who answers?

- My browser
- My computer
- DNS servers in the organisation (e.g. ECP)

Name resolution

I give you “Alice”, give me `1.2.3.4`

Mostly the work of the *Domain Name Service* (DNS) protocol.

Demo: `dig`, `host`, `dig -x`

Who answers?

- My browser
- My computer
- DNS servers in the organisation (e.g. ECP)
- DNS servers of the ISP

Name resolution

I give you “Alice”, give me `1.2.3.4`

Mostly the work of the *Domain Name Service* (DNS) protocol.

Demo: `dig`, `host`, `dig -x`

Who answers?

- My browser
- My computer
- DNS servers in the organisation (e.g. ECP)
- DNS servers of the ISP
- 3rd party DNS servers (e.g. Google)

Name resolution

I give you “Alice”, give me `1.2.3.4`

Mostly the work of the *Domain Name Service* (DNS) protocol.

Demo: `dig`, `host`, `dig -x`

Who answers?

- My browser
- My computer
- DNS servers in the organisation (e.g. ECP)
- DNS servers of the ISP
- 3rd party DNS servers (e.g. Google)
- Local DNS authority

Name resolution

I give you “Alice”, give me `1.2.3.4`

Mostly the work of the *Domain Name Service* (DNS) protocol.

Demo: `dig`, `host`, `dig -x`

Who answers?

- My browser
- My computer
- DNS servers in the organisation (e.g. ECP)
- DNS servers of the ISP
- 3rd party DNS servers (e.g. Google)
- Local DNS authority

Cache hierarchy.

Name resolution

I give you “Alice”, give me `1.2.3.4`

Mostly the work of the *Domain Name Service* (DNS) protocol.

Demo: `dig`, `host`, `dig -x`

Who answers?

- My browser
- My computer
- DNS servers in the organisation (e.g. ECP)
- DNS servers of the ISP
- 3rd party DNS servers (e.g. Google)
- Local DNS authority

Cache hierarchy.

Question:

Name resolution

I give you “Alice”, give me `1.2.3.4`

Mostly the work of the *Domain Name Service* (DNS) protocol.

Demo: `dig`, `host`, `dig -x`

Who answers?

- My browser
- My computer
- DNS servers in the organisation (e.g. ECP)
- DNS servers of the ISP
- 3rd party DNS servers (e.g. Google)
- Local DNS authority

Cache hierarchy.

Question: hypotheses, weaknesses, vulnerabilities?

Name resolution

I give you “Alice”, give me `1.2.3.4`

Fact:

Name resolution

I give you “Alice”, give me `1.2.3.4`

Fact: DNS is *critical*,

Name resolution

I give you “Alice”, give me `1.2.3.4`

Fact: DNS is *critical*, *slow*

Name resolution

I give you “Alice”, give me `1.2.3.4`

Fact: DNS is *critical*, *slow* and *not heavily protected*.

Name resolution

I give you “Alice”, give me `1.2.3.4`

Fact: DNS is *critical*, *slow* and *not heavily protected*.
Indeed:

Name resolution

I give you “Alice”, give me `1.2.3.4`

Fact: DNS is *critical*, *slow* and *not heavily protected*.
Indeed:

- Vulnerable to cache poisoning (DNSChanger, China, DPRK, Starbucks)

Name resolution

I give you “Alice”, give me `1.2.3.4`

Fact: DNS is *critical*, *slow* and *not heavily protected*.
Indeed:

- Vulnerable to cache poisoning (DNSChanger, China, DPRK, Starbucks)
- Cleartext transactions (vulnerable to passive attacks)

Name resolution

I give you “Alice”, give me `1.2.3.4`

Fact: DNS is *critical*, *slow* and *not heavily protected*.
Indeed:

- Vulnerable to cache poisoning (DNSChanger, China, DPRK, Starbucks)
- Cleartext transactions (vulnerable to passive attacks)
- No authentication (vulnerable to active attacks)

Name resolution

I give you “Alice”, give me `1.2.3.4`

Fact: DNS is *critical*, *slow* and *not heavily protected*.
Indeed:

- Vulnerable to cache poisoning (DNSChanger, China, DPRK, Starbucks)
- Cleartext transactions (vulnerable to passive attacks)
- No authentication (vulnerable to active attacks)
- No whitelist (vulnerable to disinformation)

Name resolution

I give you “Alice”, give me `1.2.3.4`

Fact: DNS is *critical*, *slow* and *not heavily protected*.
Indeed:

- Vulnerable to cache poisoning (DNSChanger, China, DPRK, Starbucks)
- Cleartext transactions (vulnerable to passive attacks)
- No authentication (vulnerable to active attacks)
- No whitelist (vulnerable to disinformation)

Question:

Name resolution

I give you “Alice”, give me `1.2.3.4`

Fact: DNS is *critical*, *slow* and *not heavily protected*.
Indeed:

- Vulnerable to cache poisoning (DNSChanger, China, DPRK, Starbucks)
- Cleartext transactions (vulnerable to passive attacks)
- No authentication (vulnerable to active attacks)
- No whitelist (vulnerable to disinformation)

Question: How do I know I’m talking to Alice... ?

Address resolution

I give you `1.2.3.4`, send it to Alice

Problem:

Address resolution

I give you `1.2.3.4`, send it to Alice

Problem: IP addresses are allocated *dynamically*

Address resolution

I give you `1.2.3.4`, send it to Alice

Problem: IP addresses are allocated *dynamically*

Solution:

Address resolution

I give you `1.2.3.4`, send it to Alice

Problem: IP addresses are allocated *dynamically*

Solution: Identify a machine and bind its IP to its identity.

Address resolution

I give you `1.2.3.4`, send it to Alice

Problem: IP addresses are allocated *dynamically*

Solution: Identify a machine and bind its IP to its identity.

$\Rightarrow$ Address Resolution Protocol (ARP)

Demo:

Address resolution

I give you `1.2.3.4`, send it to Alice

Problem: IP addresses are allocated *dynamically*

Solution: Identify a machine and bind its IP to its identity.

$\Rightarrow$ Address Resolution Protocol (ARP)

Demo: `arp -nav`

Address resolution

I give you `1.2.3.4`, send it to Alice

Who answers?

Address resolution

I give you `1.2.3.4`, send it to Alice

Who answers? Yep, a cache hierarchy.

Address resolution

I give you `1.2.3.4`, send it to Alice

Who answers? Yep, a cache hierarchy.

So... ARP...

Address resolution

I give you `1.2.3.4`, send it to Alice

Who answers? Yep, a cache hierarchy.
So... ARP...

- Vulnerable to cache poisoning (ARPspoof, dsniff)

Address resolution

I give you `1.2.3.4`, send it to Alice

Who answers? Yep, a cache hierarchy.
So... ARP...

- Vulnerable to cache poisoning (ARPspoof, dsniff)
- Cleartext transactions (vulnerable to passive attacks)

Address resolution

I give you `1.2.3.4`, send it to Alice

Who answers? Yep, a cache hierarchy.

So... ARP...

- Vulnerable to cache poisoning (ARPspoof, dsniff)
- Cleartext transactions (vulnerable to passive attacks)
- No authentication (vulnerable to active attacks)

Address resolution

I give you `1.2.3.4`, send it to Alice

Who answers? Yep, a cache hierarchy.
So... ARP...

- Vulnerable to cache poisoning (ARPspoof, dsniff)
- Cleartext transactions (vulnerable to passive attacks)
- No authentication (vulnerable to active attacks)

Impact limited to a LAN.

IP addresses: one last thing

No authentication

IP addresses: one last thing

No authentication means *source IP can be anything*.

IP addresses: one last thing

No authentication means *source IP can be anything*.

Why would you lie?

IP addresses: one last thing

No authentication means *source IP can be anything*.

Why would you lie?

- You may not care about a reply

IP addresses: one last thing

No authentication means *source IP can be anything*.

Why would you lie?

- You may not care about a reply
- You may want the reply to go to someone else (see later)

IP addresses: one last thing

No authentication means *source IP can be anything*.

Why would you lie?

- You may not care about a reply
- You may want the reply to go to someone else (see later)
- You may want someone else to get the blame

IP addresses: one last thing

No authentication means *source IP can be anything*.

Why would you lie?

- You may not care about a reply
- You may want the reply to go to someone else (see later)
- You may want someone else to get the blame

$\Rightarrow$ IP address

IP addresses: one last thing

No authentication means *source IP can be anything*.

Why would you lie?

- You may not care about a reply
- You may want the reply to go to someone else (see later)
- You may want someone else to get the blame

$\Rightarrow$ IP address $\neq$

IP addresses: one last thing

No authentication means *source IP can be anything*.

Why would you lie?

- You may not care about a reply
- You may want the reply to go to someone else (see later)
- You may want someone else to get the blame

$\Rightarrow$ IP address $\neq$ identity !

IP addresses: one last thing

No authentication means *source IP can be anything*.

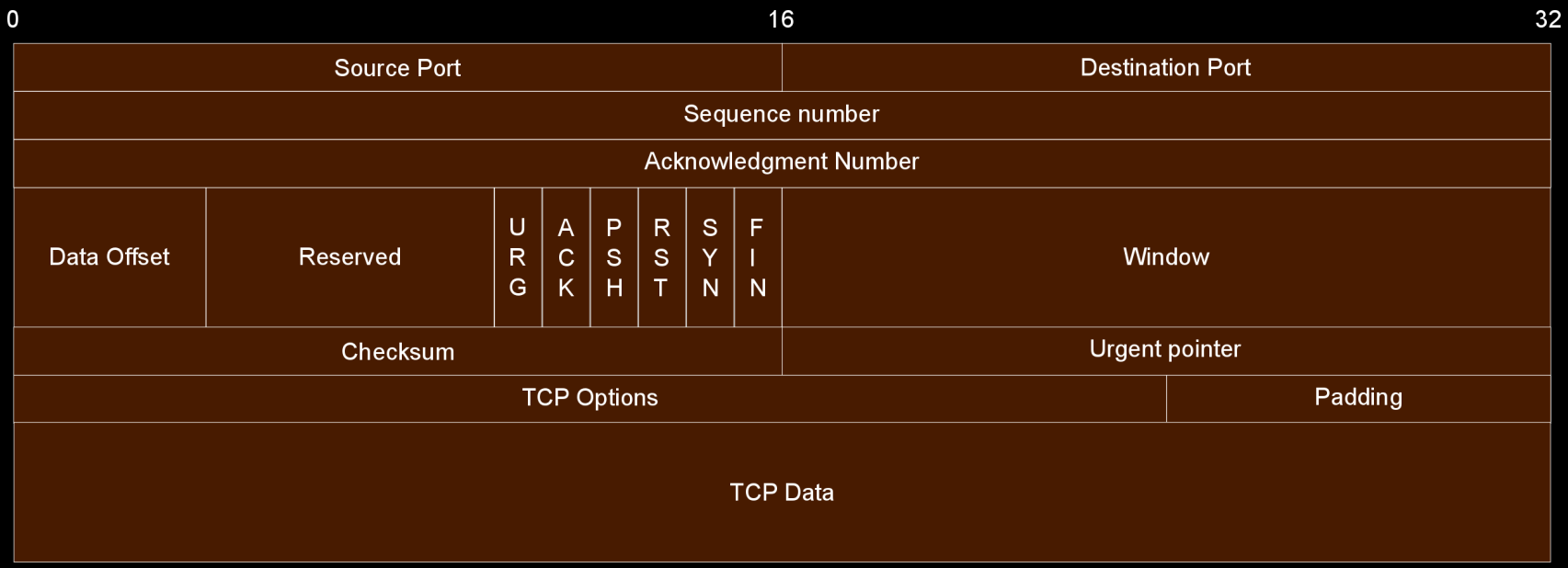
Why would you lie?

- You may not care about a reply
- You may want the reply to go to someone else (see later)
- You may want someone else to get the blame

$\Rightarrow$ IP address $\neq$ identity !
(sorry Hadopi...)

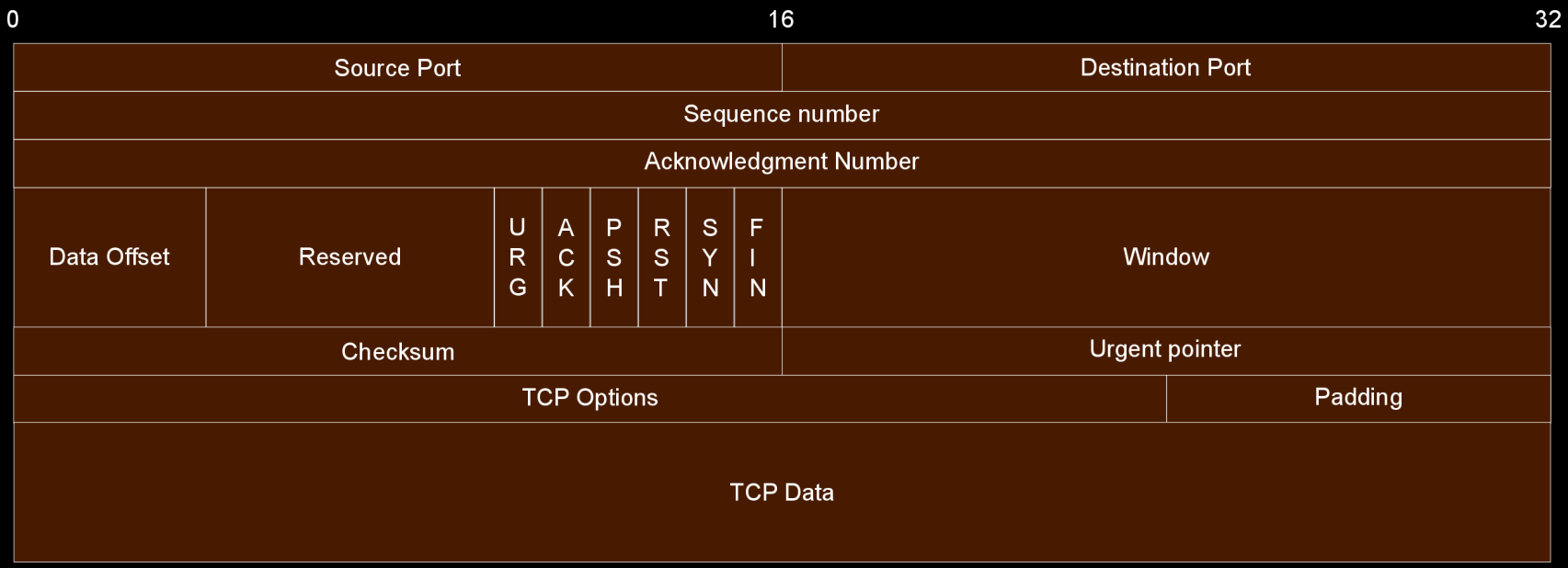
TCP: a few more notions

TCP packets



TCP: a few more notions

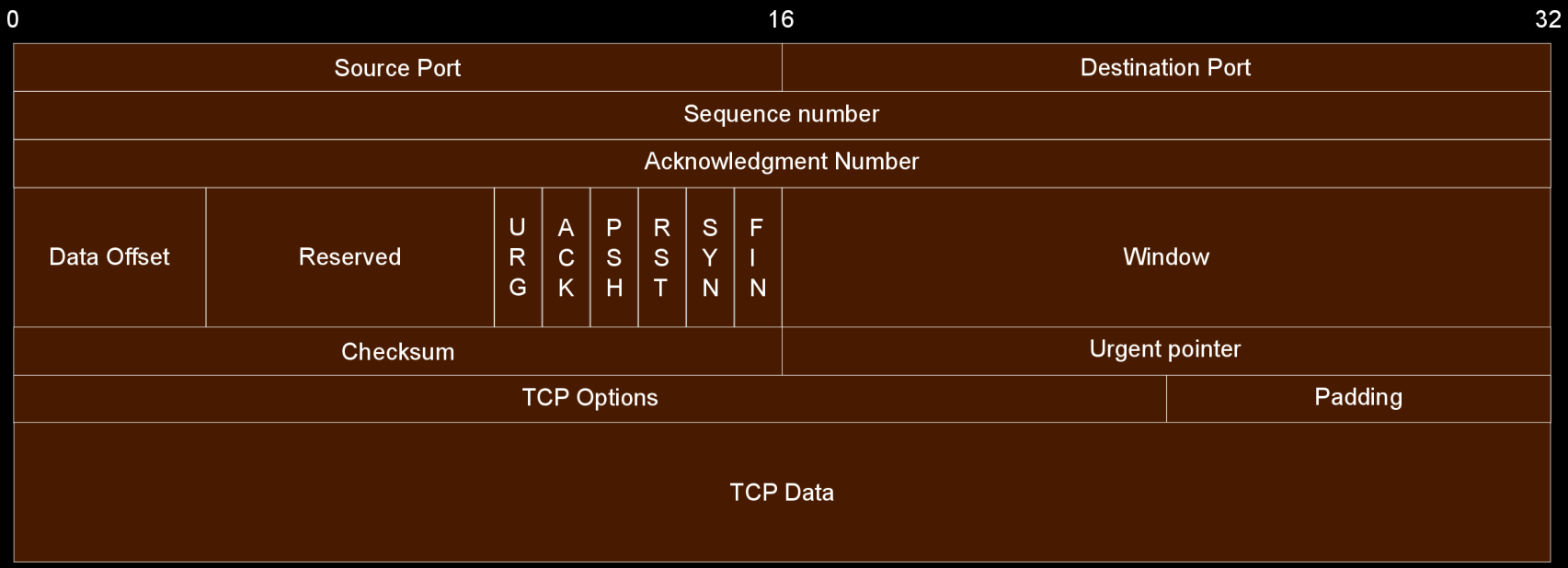
TCP packets



Interesting:

TCP: a few more notions

TCP packets

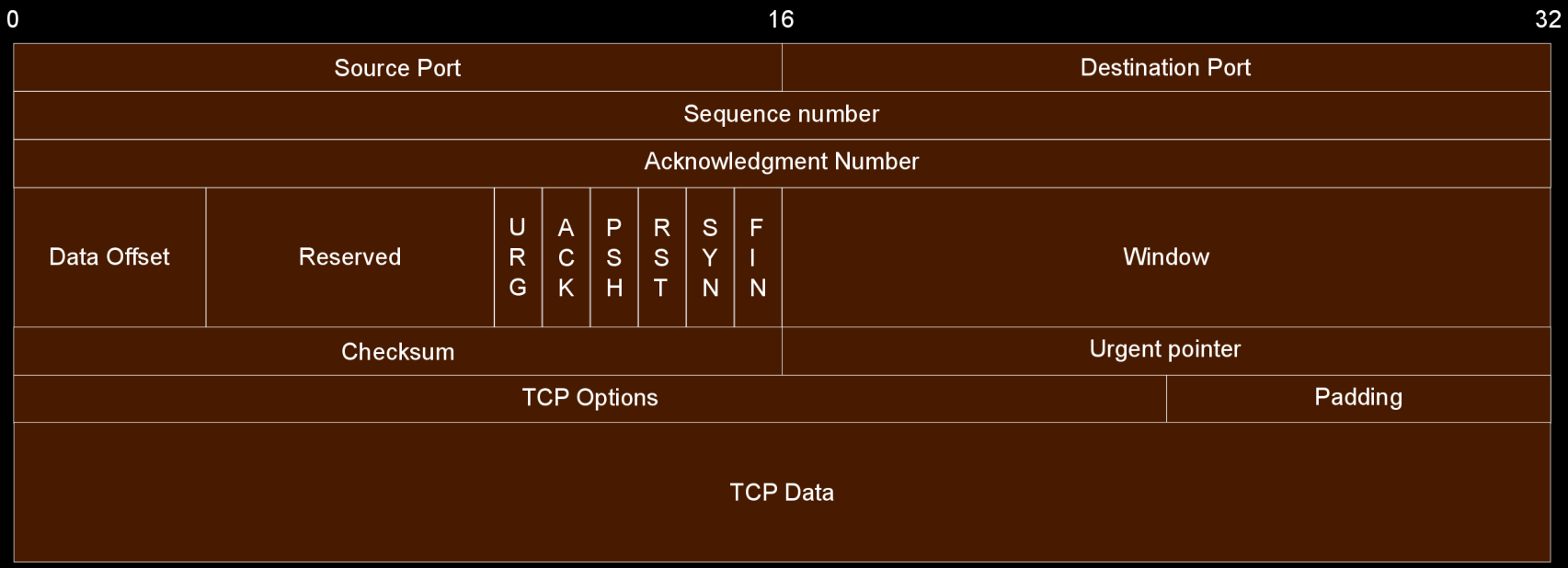


Interesting:

- Sequence and acknowledgement numbers

TCP: a few more notions

TCP packets

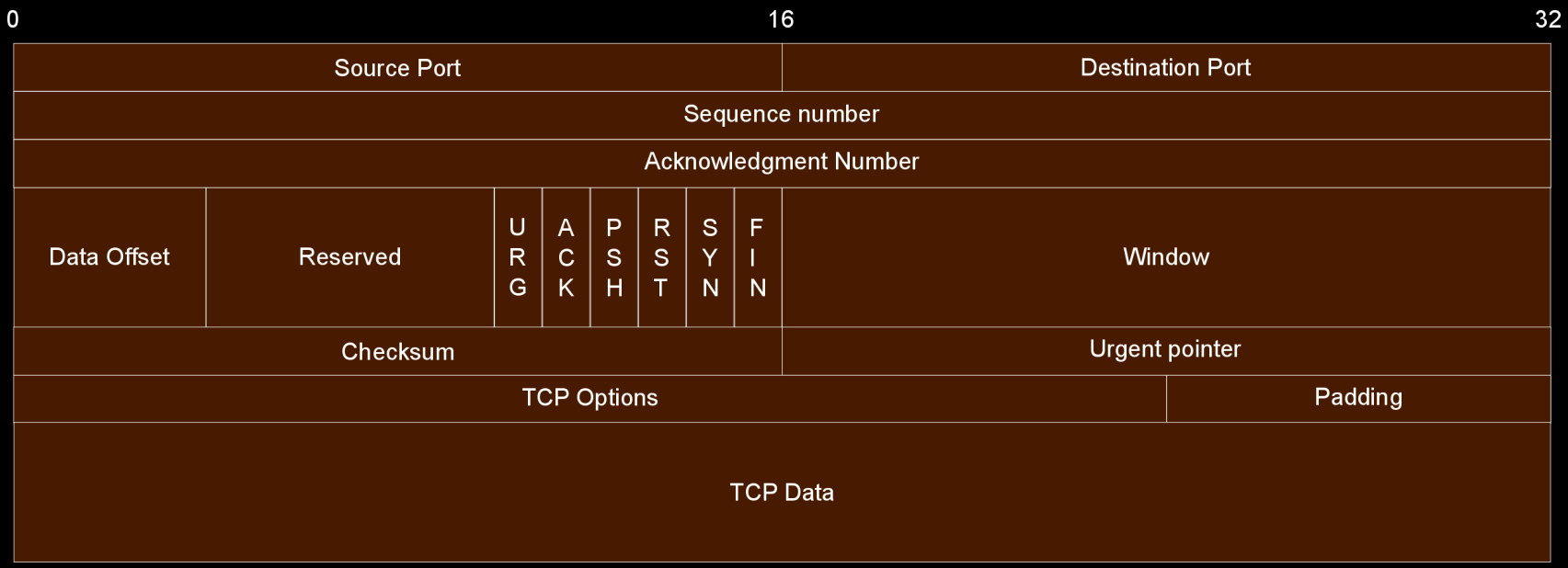


Interesting:

- Sequence and acknowledgement numbers
- Bit flags (URG, ACK, etc.)

TCP: a few more notions

TCP packets



Interesting:

- Sequence and acknowledgement numbers
- Bit flags (URG, ACK, etc.)
- No authentication either, just a checksum

TCP: a few more notions

The TCP handshake

TCP is connection-based .

TCP: a few more notions

The TCP handshake

TCP is connection-based .The 3-way handshake:

TCP: a few more notions

The TCP handshake

TCP is connection-based .The 3-way handshake:

- Client sends `SYN` packet to Server

TCP: a few more notions

The TCP handshake

TCP is connection-based .The 3-way handshake:

- Client sends `SYN` packet to Server
- Server accepts by sending `SYN+ACK` packet to Client

TCP: a few more notions

The TCP handshake

TCP is connection-based .The 3-way handshake:

- Client sends `SYN` packet to Server
- Server accepts by sending `SYN+ACK` packet to Client
- Client confirms by sending `ACK` packet to Server.

TCP: a few more notions

The TCP handshake

TCP is connection-based .The 3-way handshake:

- Client sends `SYN` packet to Server
- Server accepts by sending `SYN+ACK` packet to Client
- Client confirms by sending `ACK` packet to Server.

Question:

TCP: a few more notions

The TCP handshake

TCP is connection-based .The 3-way handshake:

- Client sends `SYN` packet to Server
- Server accepts by sending `SYN+ACK` packet to Client
- Client confirms by sending `ACK` packet to Server.

Question: what happens if the connection terminates early?

TCP: a few more notions

The TCP handshake – interrupted

At step 2, the Server has sent an `ACK` and opened a socket.

TCP: a few more notions

The TCP handshake – interrupted

At step 2, the Server has sent an `ACK` and opened a socket.
Why would the Client stop there?

TCP: a few more notions

The TCP handshake – interrupted

At step 2, the Server has sent an `ACK` and opened a socket.
Why would the Client stop there?

- Port scanning (more on that in a minute)

TCP: a few more notions

The TCP handshake – interrupted

At step 2, the Server has sent an `ACK` and opened a socket.
Why would the Client stop there?

- Port scanning (more on that in a minute)
- Denial of Service by resource exhaustion

TCP: a few more notions

The TCP handshake – interrupted

At step 2, the Server has sent an `ACK` and opened a socket.
Why would the Client stop there?

- Port scanning (more on that in a minute)
- Denial of Service by resource exhaustion
- Fingerprinting

TCP: a few more notions

The TCP handshake – interrupted

At step 2, the Server has sent an `ACK` and opened a socket.
Why would the Client stop there?

- Port scanning (more on that in a minute)
- Denial of Service by resource exhaustion
- Fingerprinting

More generally, fingerprinting by sending silly packets.

TCP: a few more notions

The TCP handshake – interrupted

At step 2, the Server has sent an `ACK` and opened a socket.
Why would the Client stop there?

- Port scanning (more on that in a minute)
- Denial of Service by resource exhaustion
- Fingerprinting

More generally, fingerprinting by sending silly packets. Why? (see next slide)

TCP: a few more notions

TCP sequence

About these sequence/acknowledgement numbers...

TCP: a few more notions

TCP sequence

About these sequence/acknowledgement numbers...

- What are they here for?

TCP: a few more notions

TCP sequence

About these sequence/acknowledgement numbers...

- What are they here for?
- What would happen if *someone* guessed them? $\Rightarrow$ TCP session hijack

TCP: a few more notions

TCP sequence

About these sequence/acknowledgement numbers...

- What are they here for?
- What would happen if *someone* guessed them? $\Rightarrow$ TCP session hijack
- Mitnick/Shimomura

TCP: a few more notions

TCP sequence

About these sequence/acknowledgement numbers...

- What are they here for?
- What would happen if *someone* guessed them? $\Rightarrow$ TCP session hijack
- Mitnick/Shimomura

Fact:

TCP: a few more notions

TCP sequence

About these sequence/acknowledgement numbers...

- What are they here for?
- What would happen if *someone* guessed them? $\Rightarrow$ TCP session hijack
- Mitnick/Shimomura

Fact: many early implementations have *predicable* sequence numbers.

TCP: a few more notions

TCP sequence

About these sequence/acknowledgement numbers...

- What are they here for?
- What would happen if *someone* guessed them? $\Rightarrow$ TCP session hijack
- Mitnick/Shimomura

Fact: many early implementations have *predicable* sequence numbers.

Fact2:

TCP: a few more notions

TCP sequence

About these sequence/acknowledgement numbers...

- What are they here for?
- What would happen if *someone* guessed them? $\Rightarrow$ TCP session hijack
- Mitnick/Shimomura

Fact: many early implementations have *predicable* sequence numbers.
Fact2: many embedded systems have early implementations.

TCP: a few more notions

More details on the handshake

According to the RFC:

TCP: a few more notions

More details on the handshake

According to the RFC:

- A machine that accepts sends `ACK`, otherwise `RST`

TCP: a few more notions

More details on the handshake

According to the RFC:

- A machine that accepts sends `ACK`, otherwise `RST`
- A machine receiving an unsollicited `SYN/ACK` responds with `RST`

TCP: a few more notions

More details on the handshake

According to the RFC:

- A machine that accepts sends `ACK`, otherwise `RST`
- A machine receiving an unsollicited `SYN/ACK` responds with `RST`
- An unsollicited `RST` is ignored

Zombie scanning

Hacker fridges!

Assume there is an idle device F

Zombie scanning

Hacker fridges!

Assume there is an idle device F whose sequence number you can predict.

Zombie scanning

Hacker fridges!

Assume there is an idle device F whose sequence number you can predict.

- Contact F and record the current sequence number

Zombie scanning

Hacker fridges!

Assume there is an idle device F whose sequence number you can predict.

- Contact F and record the current sequence number
- Send a forged `SYN` packet to the victim C,

Zombie scanning

Hacker fridges!

Assume there is an idle device F whose sequence number you can predict.

- Contact F and record the current sequence number
- Send a forged `SYN` packet to the victim C, with IP source that of F

Zombie scanning

Hacker fridges!

Assume there is an idle device F whose sequence number you can predict.

- Contact F and record the current sequence number
- Send a forged `SYN` packet to the victim C, with IP source that of F
- The victim contacts F, thus updating its sequence number

Zombie scanning

Hacker fridges!

Assume there is an idle device F whose sequence number you can predict.

- Contact F and record the current sequence number
- Send a forged `SYN` packet to the victim C, with IP source that of F
- The victim contacts F, thus updating its sequence number
- Contact F again, record the new sequence number

Zombie scanning

Hacker fridges!

Assume there is an idle device F whose sequence number you can predict.

- Contact F and record the current sequence number
- Send a forged `SYN` packet to the victim C, with IP source that of F
- The victim contacts F, thus updating its sequence number
- Contact F again, record the new sequence number

Why would we do that?

Zombie scanning

Hacker fridges!

Assume there is an idle device F whose sequence number you can predict.

- Contact F and record the current sequence number
- Send a forged `SYN` packet to the victim C, with IP source that of F
- The victim contacts F, thus updating its sequence number
- Contact F again, record the new sequence number

Why would we do that?

- Logs (if any) indicate an attack from F.

Zombie scanning

Hacker fridges!

Assume there is an idle device F whose sequence number you can predict.

- Contact F and record the current sequence number
- Send a forged `SYN` packet to the victim C, with IP source that of F
- The victim contacts F, thus updating its sequence number
- Contact F again, record the new sequence number

Why would we do that?

- Logs (if any) indicate an attack from F.
- Blind attack: the attackers' real IP is never sent.

Zombie scanning

Hacker fridges!

Assume there is an idle device F whose sequence number you can predict.

- Contact F and record the current sequence number
- Send a forged `SYN` packet to the victim C, with IP source that of F
- The victim contacts F, thus updating its sequence number
- Contact F again, record the new sequence number

Why would we do that?

- Logs (if any) indicate an attack from F.
- Blind attack: the attackers' real IP is never sent.
- If F is trusted, may easily bypass firewalls and NIDS.

Zombie scanning

Hacker fridges!

Assume there is an idle device F whose sequence number you can predict.

- Contact F and record the current sequence number
- Send a forged `SYN` packet to the victim C, with IP source that of F
- The victim contacts F, thus updating its sequence number
- Contact F again, record the new sequence number

Why would we do that?

- Logs (if any) indicate an attack from F.
- Blind attack: the attackers' real IP is never sent.
- If F is trusted, may easily bypass firewalls and NIDS.

Demo:

Zombie scanning

Hacker fridges!

Assume there is an idle device F whose sequence number you can predict.

- Contact F and record the current sequence number
- Send a forged `SYN` packet to the victim C, with IP source that of F
- The victim contacts F, thus updating its sequence number
- Contact F again, record the new sequence number

Why would we do that?

- Logs (if any) indicate an attack from F.
- Blind attack: the attackers' real IP is never sent.
- If F is trusted, may easily bypass firewalls and NIDS.

Demo: `nmap -sI`.

Zombie scanning

Hacker fridges!

Assume there is an idle device F whose sequence number you can predict.

- Contact F and record the current sequence number
- Send a forged `SYN` packet to the victim C, with IP source that of F
- The victim contacts F, thus updating its sequence number
- Contact F again, record the new sequence number

Why would we do that?

- Logs (if any) indicate an attack from F.
- Blind attack: the attackers' real IP is never sent.
- If F is trusted, may easily bypass firewalls and NIDS.

Demo: `nmap -sI`.
Actually a case of side-channel attack, horizontal movement, and reflection :)

Reflection attack

Key idea:

Reflection attack

Key idea: some people are talkative.

Reflection attack

Key idea: some people are talkative.

Demo:

- `dig DNSKEY isc.org @8.8.8.8`

Reflection attack

Key idea: some people are talkative.

Demo:

- `dig DNSKEY isc.org @8.8.8.8`
- The packet sent is 25 bytes

Reflection attack

Key idea: some people are talkative.

Demo:

- `dig DNSKEY isc.org @8.8.8.8`
- The packet sent is 25 bytes
- The packet received is 461 bytes ($\sim 18\times$)

Reflection attack

Key idea: some people are talkative.

Demo:

- `dig DNSKEY isc.org @8.8.8.8`
- The packet sent is 25 bytes
- The packet received is 461 bytes ($\sim 18\times$)
- Would be a shame if the IP source was not mine...

Reflection attack

Key idea: some people are talkative.

Demo:

- `dig DNSKEY isc.org @8.8.8.8`
- The packet sent is 25 bytes
- The packet received is 461 bytes ($\sim 18\times$)
- Would be a shame if the IP source was not mine...
- Especially since DNS servers are fast and have huge bandwidth!

Reflection attack

Key idea: some people are talkative.

Demo:

- `dig DNSKEY isc.org @8.8.8.8`
- The packet sent is 25 bytes
- The packet received is 461 bytes ($\sim 18\times$)
- Would be a shame if the IP source was not mine...
- Especially since DNS servers are fast and have huge bandwidth!

Same idea applies to *e.g.* NTP

NTP reflection attack

In 2014, `@derptrolling` organised a massive-scale NTP-amplification attack, peaking at

NTP reflection attack

In 2014, `@derptrolling` organised a massive-scale NTP-amplification attack, peaking at 400 Gb/s.

NTP reflection attack

In 2014, `@derptrolling` organised a massive-scale NTP-amplification attack, peaking at 400 Gb/s.

No network infrastructure today can handle 400 Gb/s.

Table of Contents

Wrapping up

"How I hacked my IP camera and found a backdoor"

Wrapping up

"How I hacked my IP camera and found a backdoor"

- Sniffing packets (see Lab) reveals password stored and sent in clear

Wrapping up

"How I hacked my IP camera and found a backdoor"

- Sniffing packets (see Lab) reveals password stored and sent in clear
- Port scanning (see Lab) reveals hidden server on the Telnet port

Wrapping up

"How I hacked my IP camera and found a backdoor"

- Sniffing packets (see Lab) reveals password stored and sent in clear
- Port scanning (see Lab) reveals hidden server on the Telnet port
- $ enables injection (see next lecture), runs commands

Wrapping up

"How I hacked my IP camera and found a backdoor"

- Sniffing packets (see Lab) reveals password stored and sent in clear
- Port scanning (see Lab) reveals hidden server on the Telnet port
- $ enables injection (see next lecture), runs commands
- Leak via DNS using ping

Wrapping up

"How I hacked my IP camera and found a backdoor"

- Sniffing packets (see Lab) reveals password stored and sent in clear
- Port scanning (see Lab) reveals hidden server on the Telnet port
- $ enables injection (see next lecture), runs commands
- Leak via DNS using ping
- Can change root password using injection

Wrapping up

"How I hacked my IP camera and found a backdoor"

- Sniffing packets (see Lab) reveals password stored and sent in clear
- Port scanning (see Lab) reveals hidden server on the Telnet port
- $ enables injection (see next lecture), runs commands
- Leak via DNS using ping
- Can change root password using injection
- Anyway root password was: 123456 (root:LSiuY7pOmZG2s:0:0:Administrator:/:/bin/sh)

Wrapping up

"How I hacked my IP camera and found a backdoor"

- Sniffing packets (see Lab) reveals password stored and sent in clear
- Port scanning (see Lab) reveals hidden server on the Telnet port
- $ enables injection (see next lecture), runs commands
- Leak via DNS using ping
- Can change root password using injection
- Anyway root password was: 123456 (root:LSiuY7pOmZG2s:0:0:Administrator:/:/bin/sh)
- Root password reset on reboot

Wrapping up

"How I hacked my IP camera and found a backdoor"

- Sniffing packets (see Lab) reveals password stored and sent in clear
- Port scanning (see Lab) reveals hidden server on the Telnet port
- $ enables injection (see next lecture), runs commands
- Leak via DNS using ping
- Can change root password using injection
- Anyway root password was: `123456`
  (root:LSiuY7pOmZG2s:0:0:Administrator:/:/bin/sh)
- Root password reset on reboot

`http://jumpespjump.blogspot.fr/2015/09/how-i-hacked-my-ip-camera-and-found.html`

A few more questions

More questions!

A few more questions

More questions!

- What about the security of WiFi networks?

A few more questions

More questions!

- What about the security of WiFi networks?
- How do we trace attack source?

A few more questions

More questions!

- What about the security of WiFi networks?
- How do we trace attack source?
- Security layers (IPsec, DNSSEC, SEND, etc.)?

A few more questions

More questions!

- What about the security of WiFi networks?
- How do we trace attack source?
- Security layers (IPsec, DNSSEC, SEND, etc.)?

If you're interested, research it (or request a lecture)