vulnerabilità delle reti

vulnerabilità apparati

- gli apparati possono essere vulnerabili
  - errata implementazione dei protocolli
    - tutti gli standard lasciano un margine di decisione al progettista
  - vulnerabilità del firmware
    - es. buffer overflow
- ci concentriamo su vulnerabilità delle reti non legate all'implementazione o al software

reti e protocolli vulnerabili

- gran parte delle vulnerabilità specifiche delle reti sono in realtà

  vulnerabilità dei protocolli

- inserire la sicurezza in un protocollo significa costringere tutte le implementazioni a realizzare quelle funzionalità
  - costoso, inefficiente, non sempre necessario, ecc.

protocolli in chiaro e non autenticati

- è facile con uno sniffer ricostruire le sessioni tcp e trovare password
  - es. Wireshark/Ethereal
  - esistono strumenti più sofisticati
    - sudo apt-get install ettercap
    - sudo apt-get install dsniff
- è facile inserire pacchetti sulla rete facendo credere che li ha inviati un’altra macchina

reti locali vs. Internet

- protocolli vulnerabili spesso sono una minaccia solo se l'hacker è "presente" su una lan per cui passa il traffico vulnerabile
- "presente" significa
  - presenza fisica (collegamento ethernet o wifi)
  - controllo remoto di una macchina (win, unix, apparato di rete)
- zone critiche:
  - la lan dell'utente
  - una lan di una server farm (es. web hosting)
  - una lan di un ISP intermedio

fiducia e sniffabilità

- questa vulnerabilità sono minacce solo se…
  - non ci fidiamo dell'ambiente
  - il traffico è "sniffabile"
- sniffare su una lan
  - facile nelle reti vecchie
    - quelle inerentemente brodcast: 10base2, 10baseT con hubs
  - leggermente più complesso per reti switched

reti switched

- le tecnologie per reti locali nascono come inerentemente broadcast
  - ma ora le reti sono tutte switched
- si dice che in una rete switched non si possa sniffare quasi nulla
- purtroppo le reti switched sono molto vulnerabili

mac flood

- quando uno switch satura la sua source address table si comporta come un hub
  - a questo punto lo sniffer vede tutto
- molto invasivo
  - alcuni switch vanno in crash
  - le spie dello switch segnalano traffico molto intenso

terminologia

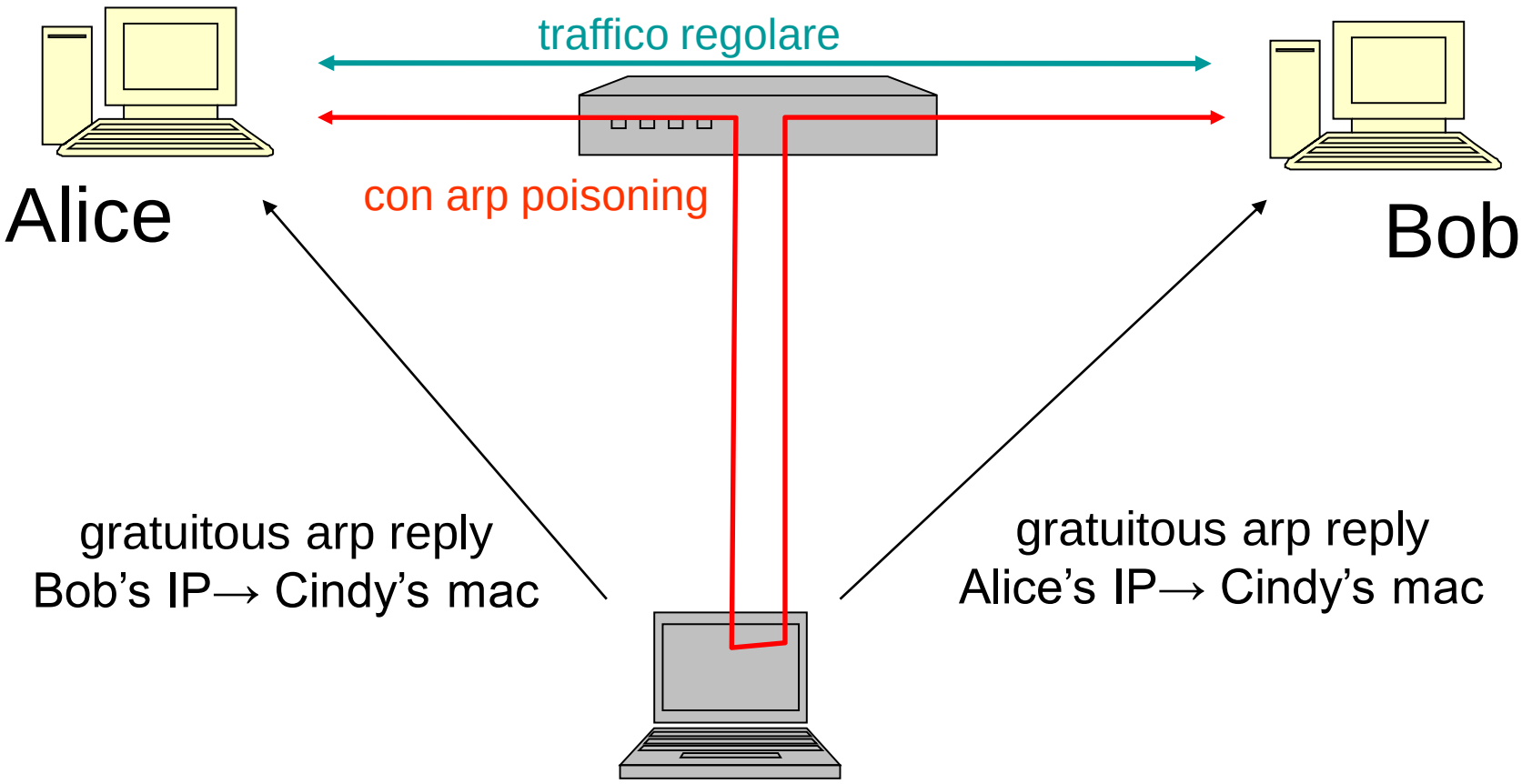
- *spoofing o forgery*: i termini sono usati quasi indifferentemente per denotare cambiamenti illeciti di dati
  - tipicamente in rete
  - header/campi di pacchetti o di messaggi
    - es. l'indirizzo sorgente
- *poisoning*: alterazione di una cache o altro stato

arp poisoning (o spoofing)

- le implementazioni di arp sono stateless
  - da standard, praticamente tutte
  - aggiornano la arp cache ogni volta che ricevono un'arp reply… anche se non hanno inviato alcuna arp request!

- si può "avvelenare" la arp cache inviando delle arp reply "gratuite"
  - è visibile dalla macchina avvelenata (arp -a)
- le entry statiche risolvono i problema
  - ma rendono la vita impossibile!

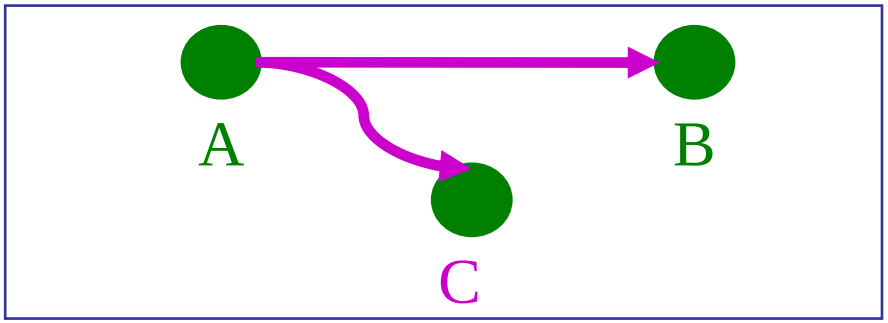
arp poisoning



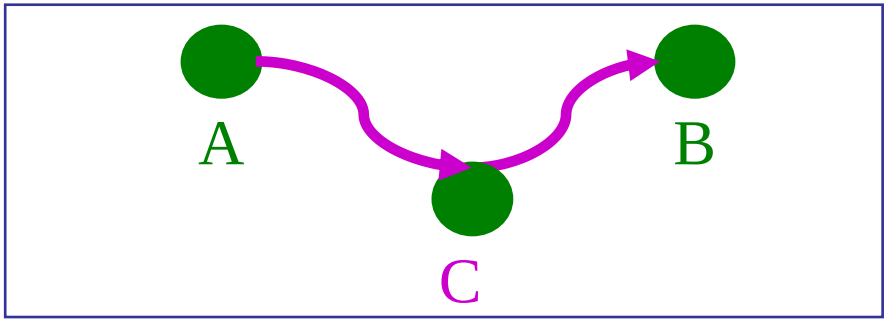
moving to ipv6

- well... sooner or later ARP will be a legacy protocol
- ipv6 use Neighbor Discovery (ND)
  - rfc 4861
- same behavior
  - Neighbor solicitation ≡ ARP request
  - Neighbor advertisement ≡ ARP reply
- cached, stateless request
- same security problems of ARP
- SeND – **secure** neighbor discovery

attacchi Man in the Middle (MitM)



MitM passivo
solo sniffing



MitM attivo
anche modifica dei dati

MitM attivo è semplice per protocolli udp based
richiede un lavoro complesso su protocolli tcp based (gestione dei numeri di sequenza)

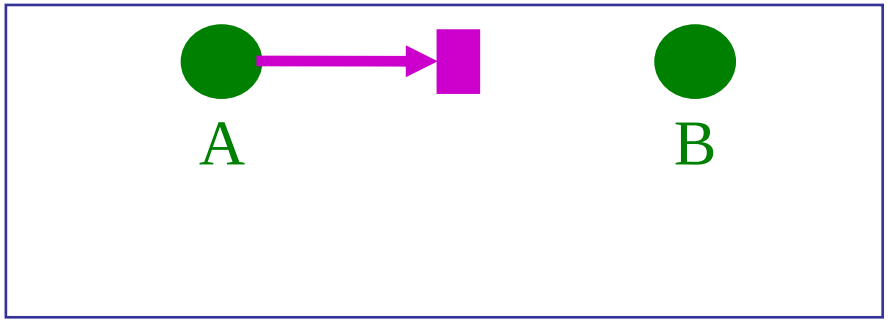
MitM terminology: active vs. passive

- consider the network stack
- a MitM can be active below a certain level and passive above

example

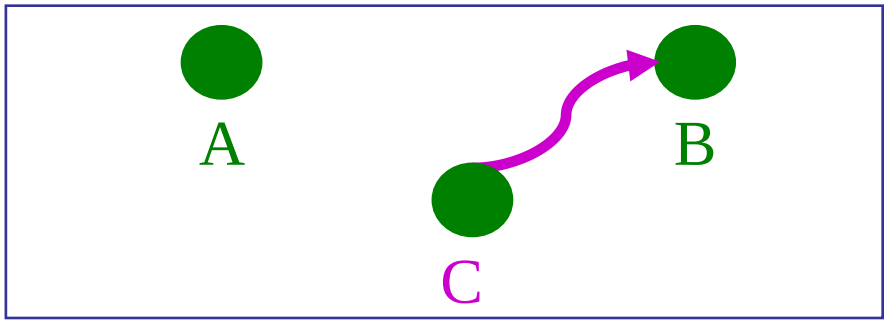
- ARP poisoning is **active** at levels 1 and 2
  - since it affects MAC addresses
- it does not affect level 3 addresses
  - it may or may not affect IP TTL and checksum
- it can be used to realize **passive** sniffing of UDP data (level 4)

denial of service (DoS)



- DoS su tutta la LAN
  - raramente è fatto saturando la rete
  - più facile saturare risorse di calcolatori
  - broadcast storm
    - per le macchine ogni broadcast ricevuto è un interrupt
- si può inibire una singola macchina con arp poisoning
  - es. dirottare tutto il traffico senza instradarlo a destinazione
  - o instradandolo selettivamente

ip address spoofing

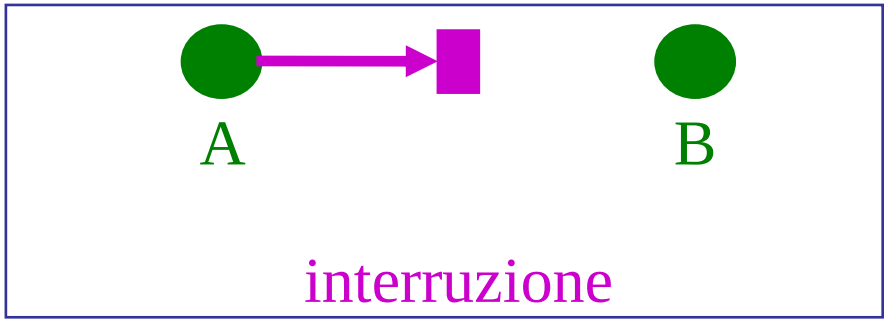


- l'indirizzo ip è facile da modificare in modo da impersonare una altra macchina
  - per mezzo di arp poisoning la macchina proprietaria dell'ip può essere neutralizzata (vedi DoS di una singola macchina)
- non riceveremo mai la risposta
  - a meno di arp poisoning di C su B
- anche su Internet
  - ma per ricevere la risposta bisogna attaccare il routing

DoS con ping smurf

- interesse storico
- esempio di sfruttamento di IP spoofing per DoS
- C vuol fare DoS su A
- C invia un echo_request con sorgente A (spoofed) e destinazione bcast (cioè “a tutte le macchine della sottorete”)
- tutte le macchine della sottorete risponderanno ad A con echo_reply
  - non tutte le macchine rispondono ai ping bcast
  - la frequenza di echo reply ricevuti da A è (echo request inviati da C) * (numero pc in subnet che rispondono al bcast)

tcp DoS (tcp reset)



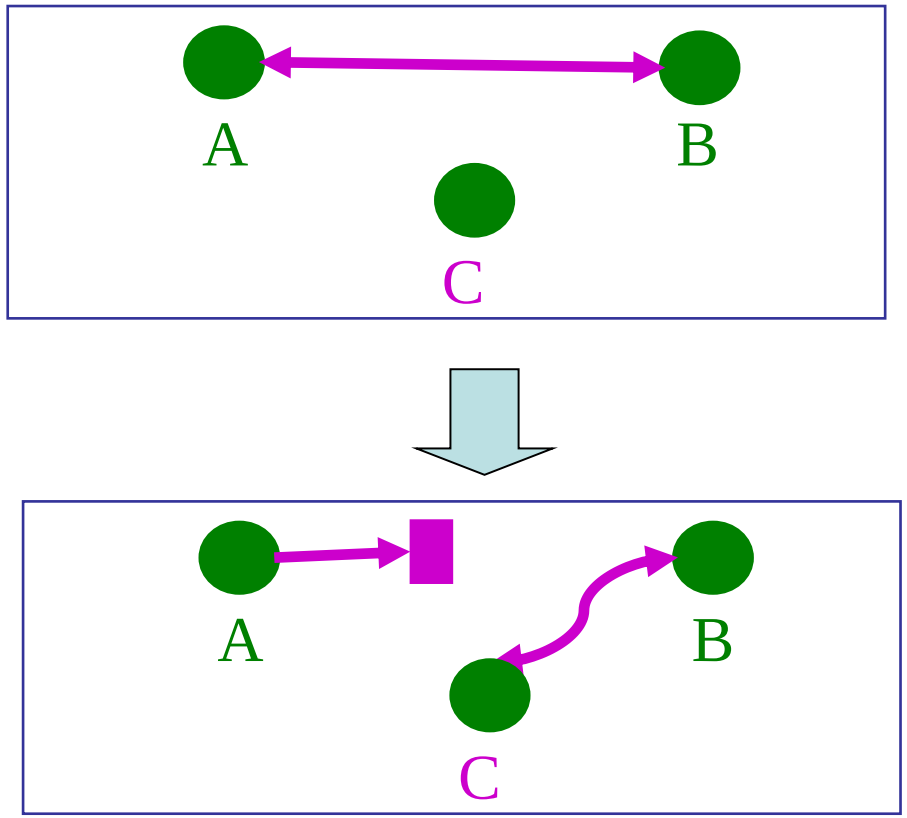
- una sessione tcp attiva può essere “buttata giù”:
  - una delle due parti può essere “resettata”
- ecco come…
  - paccetto tcp “forgiato” con la corretta quadrupla <ip, porta, ip, porta>
  - flag RST attivo
  - numero di sequenza scelto opportunamente
  - **funziona anche su Internet** poiché non ha bisogno di risposta

TCP reset e inter-domain routing

- se le sessioni tcp sono critiche può essere molto pericoloso
- le sessioni tcp più critiche sono quelle su cui si basa BGP
  - il loro reset rende intere porzioni di internet irraggiungibili
  - estensione MD5 Cisco
    - sostanzialmente una estensione di tcp per autenticare l'header
    - metodo generale ma usata solo per BGP

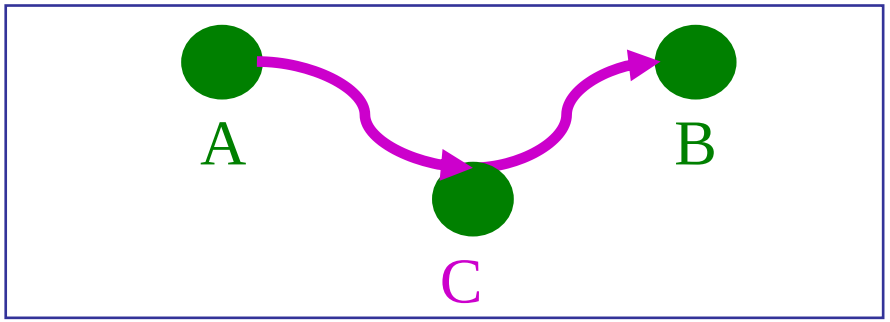
tcp session hijacking

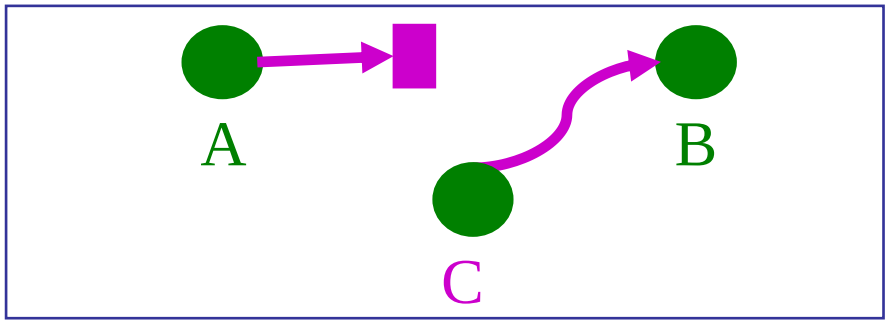
- l'obiettivo è di trasformare una sessione tcp tra A e B in una sessione tcp tra C e B senza che B se ne accorga



tcp session hijacking

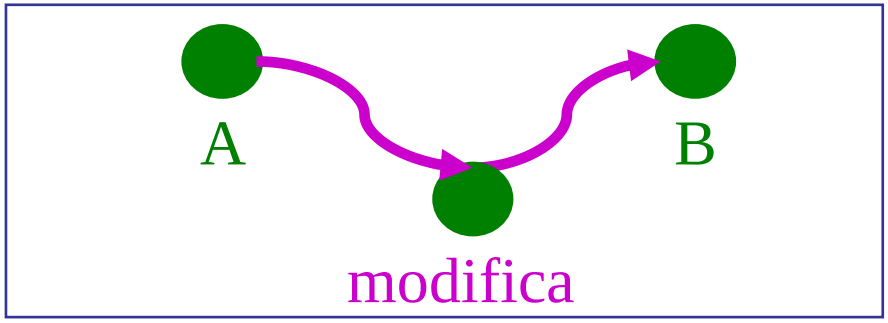
- fase1
  - MitM passivo
  - arp poisoning





- fase 2 (rubare la sessione)
  - A riceve un reset (tcp DoS su A)
  - C continua a fare arp poisoning su B
  - C continua la sessione tcp al posto di A continuando con i numeri di sequenza di A
  - B non si accorge del cambiamento di soggetto

tcp MitM attivo



- il MitM può modificare singoli bytes dei pacchetti tcp (no inserimenti e cancellazioni)
  - nessun problema con i numeri di sequenza
- inserire nuovi bytes nel flusso tcp
  - B: manda ack per numeri di sequenza che la sorgente non ha inviato
  - gli ack possono essere droppati dal MitM
  - i numeri di sequenza nei pacchetti successivi possono essere slittati dal MitM (tecnica già usata da NAT per FTP)
- risincronizzazione
  - obiettivo: terminare l'arp poisoning ma mantenere la sessione A↔B attiva
  - modo1: perdendo alcuni caratteri della sorgente
    - dipende dal protocollo (ad esempio è possibile per telnet: l'utente non vede l'eco di alcuni tasti digitati, ma il problema è temporaneo)
  - modo2: inviando un TCP SYN
    - dipende dall'implementazione, oramai considerata una vulnerabilità

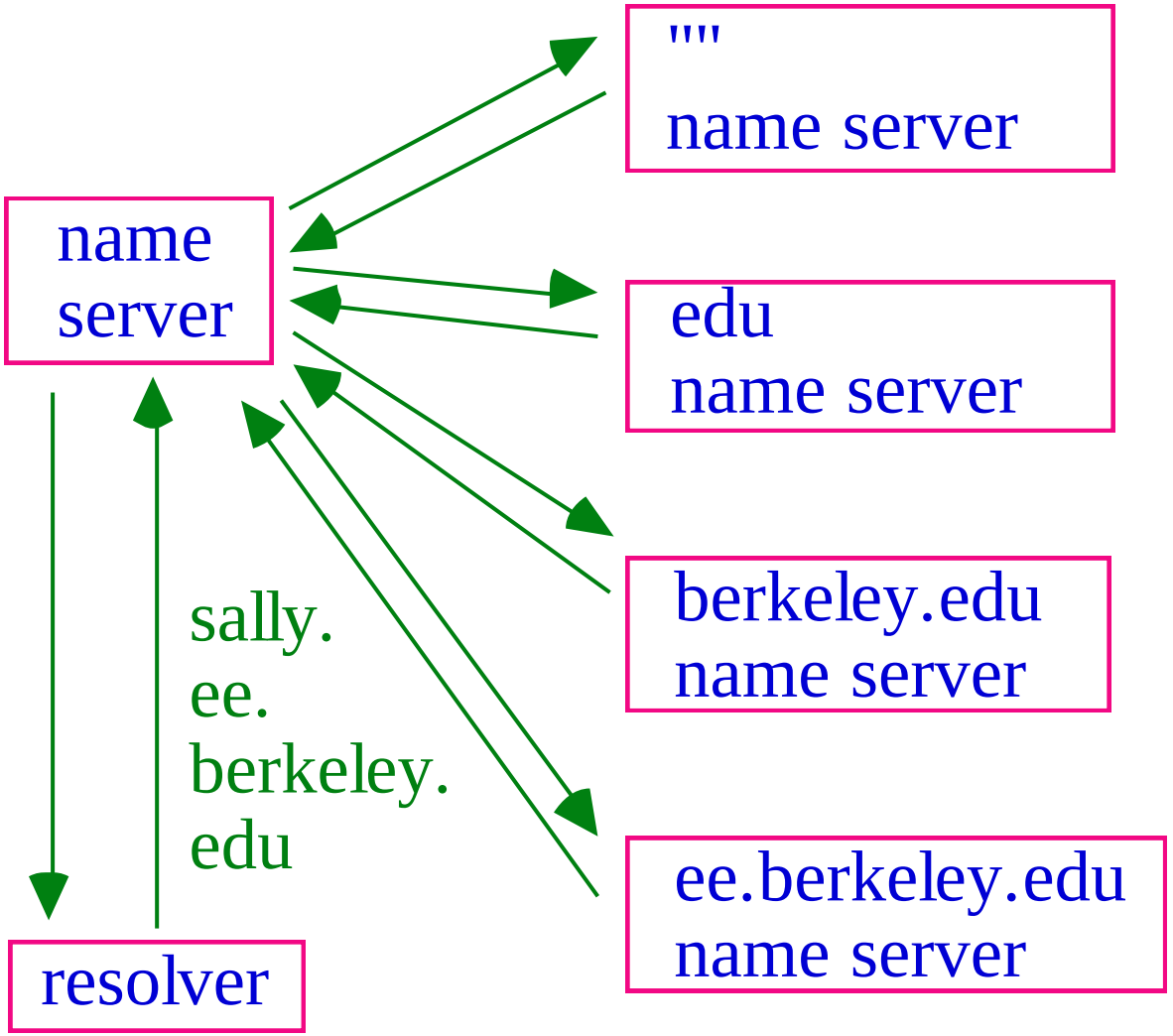
vulnerabilità del DNS

- il protocollo DNS non è autenticato
- spoofing: rispondere prima del server
  - tipicamente il protocollo è molto lento
  - facile intercettare le richieste (ad es. con arp poisoning)
  - solo in rete locale
- è una delle minacce più pericolose per il web

se non ci possiamo fidare del DNS...

- …è molto importante autenticare il (web) server
  - vedi **ssl/tls e https** nel seguito
  - oppure autenticare il DNS
    - DNSSEC: ancora poco diffuso

DNS



DNS cache poisoning

- DNS cache poisoning
  obiettivo: modificare la cache di un DNS server, da un punto qualsiasi di Internet
  - alle successive richieste il name server risponde con l'indirizzo ip deciso dall'attaccante

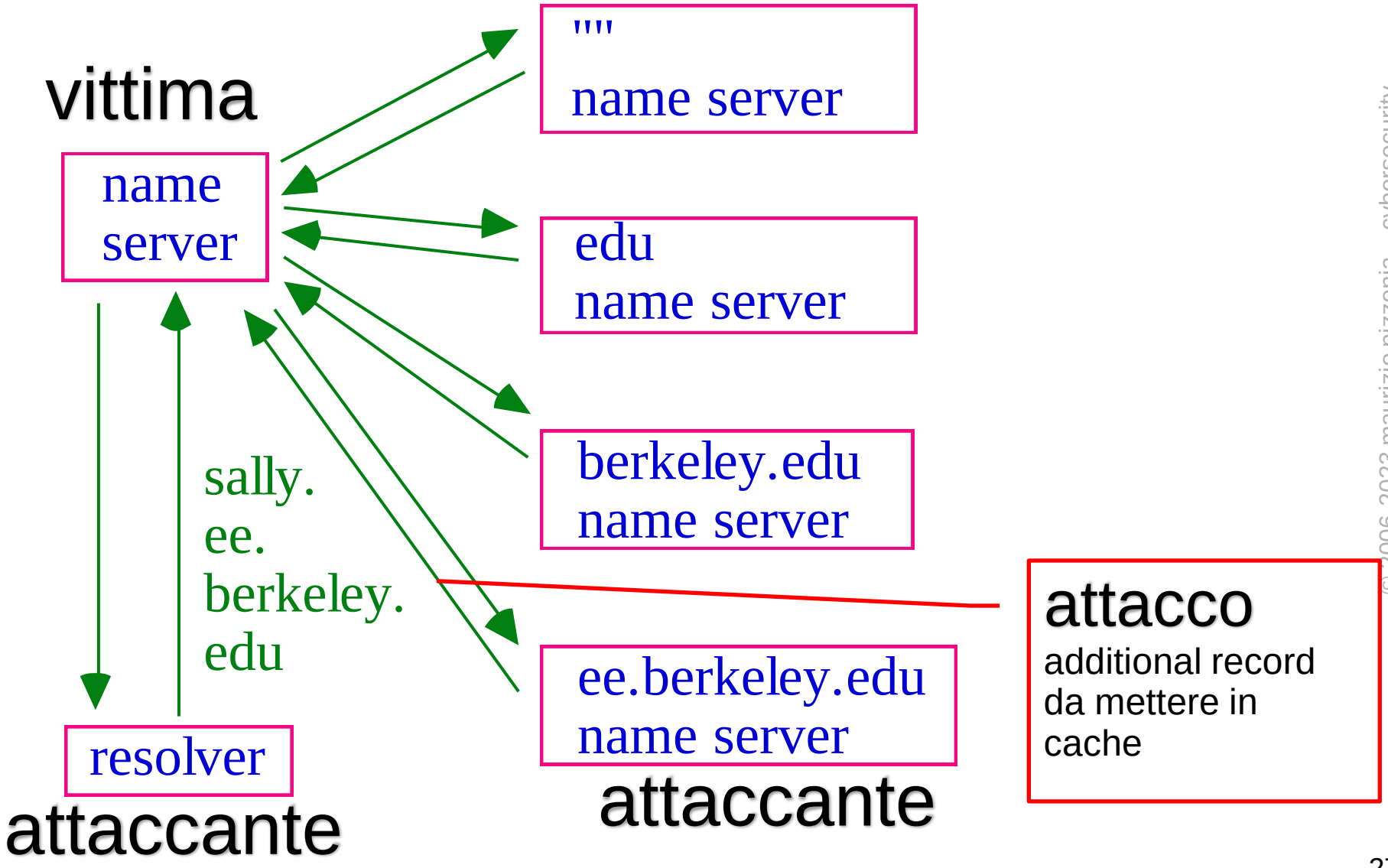
- normalmente richiede che il name server (NS) serva richieste ricorsive

DNS pharming

- alcuni NS “creduloni” mettevano in cache i record della sezione “additional records” qualunque fosse il loro contenuto
  - anche se non relative alle richieste
  - nota che gli additional records dovrebbero contenere risoluzioni per nomi citati nelle altre sezioni (es. sezione “answers”)

vedi figura a slide successiva →

DNS pharming

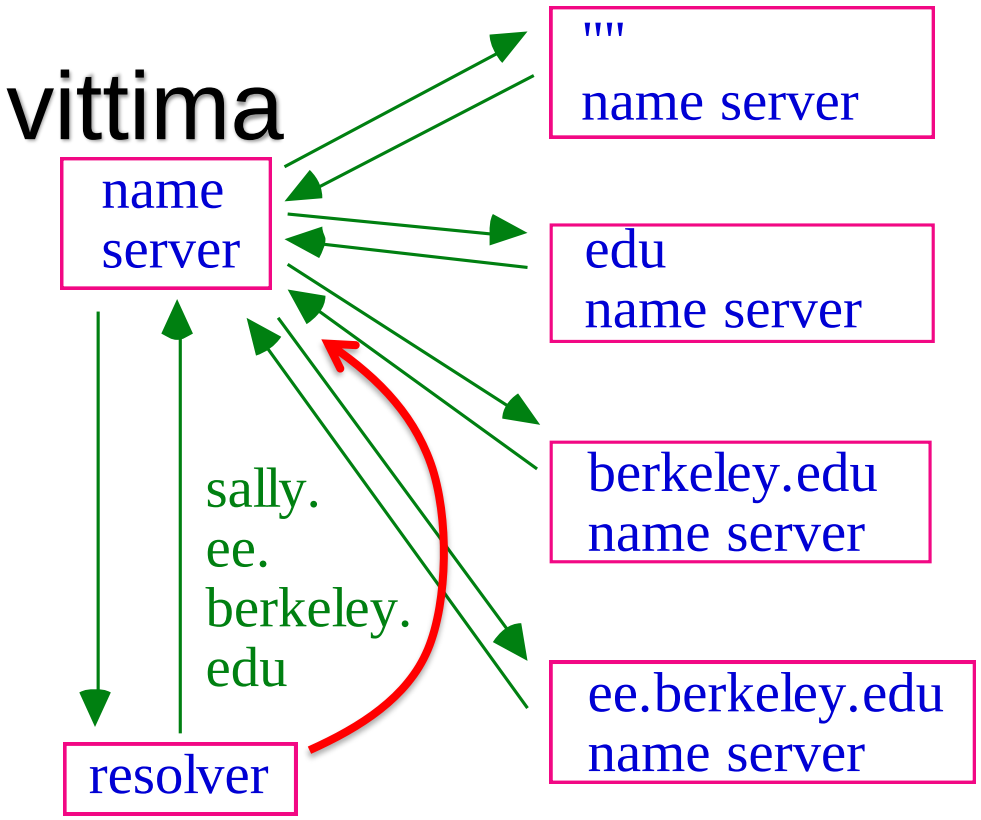


DNS pharming

- oramai i NS eseguono tutti le verifiche di consistenza della sezione additional records

vulnerabilità di Kaminsky

- potenzialmente un attaccante potrebbe impersonare il name server



l'attaccante impersona ee.berkeley.edu inserendo un indirizzo malevolo nella risposta che name server registra in cache

vulnerabilità di Kaminsky

- l'attaccante deve rispondere prima del server originale
- il name server vittima accetta la risposta quando....
  - IP sorgente corretto
  - ID della richiesta corretto
  - resource record corretto
  - porta destinazione e sorgente corrette

- l'attaccante deve indovinare tutto
- l'attaccante può inviare molte risposte, la probabilità di successo si può calcolare
- l'attacco funziona se la probabilità di indovinare tutto è alta
- ci aspettiamo che tale probabilità sia bassa, e invece…

vulnerabilità di Kaminsky

- certe osservazioni e tecniche rendono l'attacco a un NS ricorsivo molto facile se non si prendono adeguate contromisure
  - Dan Kaminsky Febbraio-Luglio 2008
  - annuncio: http://www.kb.cert.org/vuls/id/800113
  - l'analisi probabilistica è una variante del birthday attack (vedi parte di crittografia)

vulnerabilità di Kaminsky

condizioni per il successo dell'attacco

- IP sorgente corretto
  - spoofing
  - facile
- resource record corretto
  - richieste sollecitate dall'attaccante
  - facile

vulnerabilità di Kaminsky

condizioni per il successo dell'attacco

- ID della richiesta predicibile
  - vedi generazione di numeri casuali nella parte di crittografia
  - **spazio degli ID piccolo, solo 16 bit, rende possibile un attacco brute force**
- contromisura standard: usare la porta sorgente come una estensione dell'ID

- porta sorgente della richiesta
  - nelle implementazioni vulnerabili è fissa
  - deve essere scelta random come l'ID per rendere l'attacco circa 65000 volte più difficile

Internet: vulnerabilità del DNS

- documentazione dell'impatto ad ottobre 2008
  - prodotti noti come vulnerabili 39%
  - prodotti noti come non vulnerabili 11%
  - prodotti per cui non è noto lo stato 50%

- ad oggi la maggior parte delle installazione dei NS sono sicure
- ma non sempre è possibile usare porte random
    - efficienza NS, limiti del sistema operativo

Distributed DoS: SYN flood

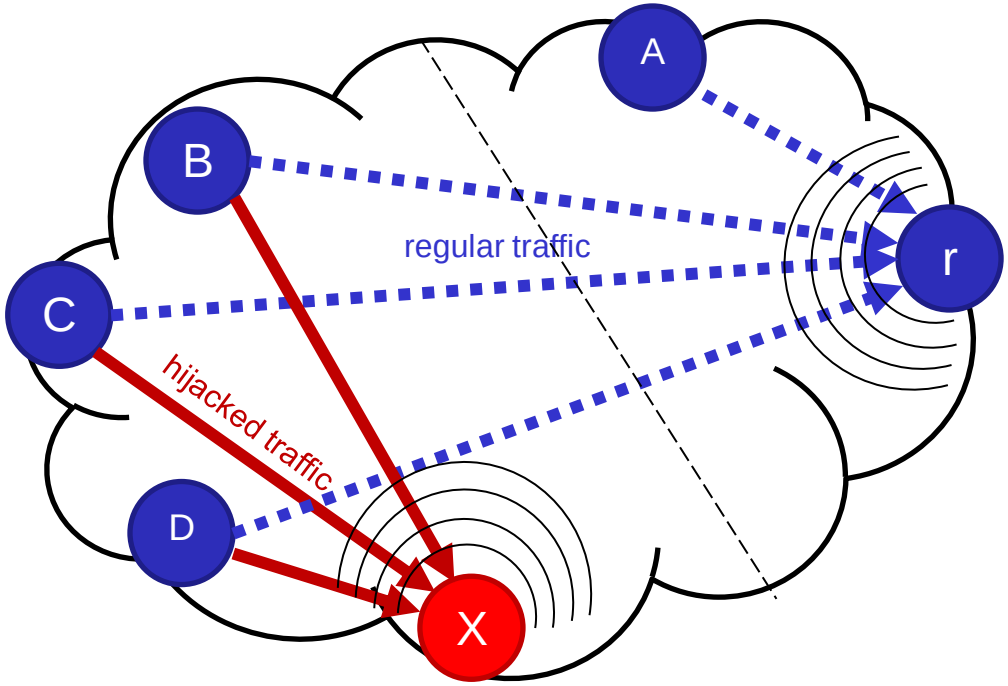
- obiettivo**: saturare le risorse del server** costringendolo ad allocare un gran numero di connessioni tcp
- tipicamente tramite syn flood
- ip sorgente spoofed (casuali)
  - la sorgente non ha bisogno di ricevere il syn+ack
- alla ricezione del syn il server…
  - alloca i buffer
  - risponde SIN+ACK
  - attende un ACK
  - mantiene allocato lo spazio fino al timeout
- difficile da evitare
  - esistono delle tecniche interessanti per contenere il problema: fanno uso accorto delle risorse quando arriva un SYN

Distributed DoS: DNS amplification

- obiettivo: **saturare la banda del server**
- attaccante: query a DNS con IP spoofed
  - l'IP della richiesta è quello della vittima
- DNS: risponde alla vittima
- **amplification**: la query è creata in modo da avere una risposta "grande"
  - DNS max 512 bytes per risposta
  - Extension mechanism for DNS, RFC 6891
    - max 4096 bytes per risposta
    - es. usato per DNSSEC
  - amplificazione x66 (60B→4KB)

route hijacking

- consiste nell'annunciare, nel protocollo di routing, da una certo router X (sotto il controllo dell'attaccante) una rotta r che non dovrebbe essere annunciata
- effetto:
  "succhiare" tutti i pacchetti destinati a *r* verso *X*,
  a meno che non ci siano annunci di *r* più vicini (ad es. il legittimo proprietario)
  - cioè la rete si partiziona in **bacini di attrazione**
  - legittimo per configurazioni anycast



route hijacking

- intra-dominio: OSPF, RIP, ecc.
  - è un problema di sicurezza dell'ISP
  - risolvibile con una azione del SOC/CSIRT dell'ISP
  - strumento usato dagli ISP per contrastare DDoS (configurazione detta "blackholing")
- inter-dominio: BGP
  - nessuna vera autorità centrale che possa agire
  - spesso sono errori di configurazione
  - usato legittimamente per supportare anycast
    - es. root name server
  - osservato spesso con indirizzi non assegnati o rotte molto generiche, usate per inviate spam e poi ritirate
  - e per il *Biggest Security Hole…*

BGP route hijacking: tactic contrast

- a route hijacking attack can be temporarily contrasted announcing more specific routes covering the attacked route

example

- Attack: prefix P=193.200.100.0/**23**, belonging to ISP X, is currently announced by an attacker A
  - customers of X in P receive traffic only if packet source is closer to X than to A
- Contrast: X can announce two *more specific* routes 193.200.10**0**.0/**24** 193.200.10**1**.0/**24** covering P
  - these routes propagate independently from P
  - by best match rule, routers chose the more specific routes for routing traffic towards customers of X

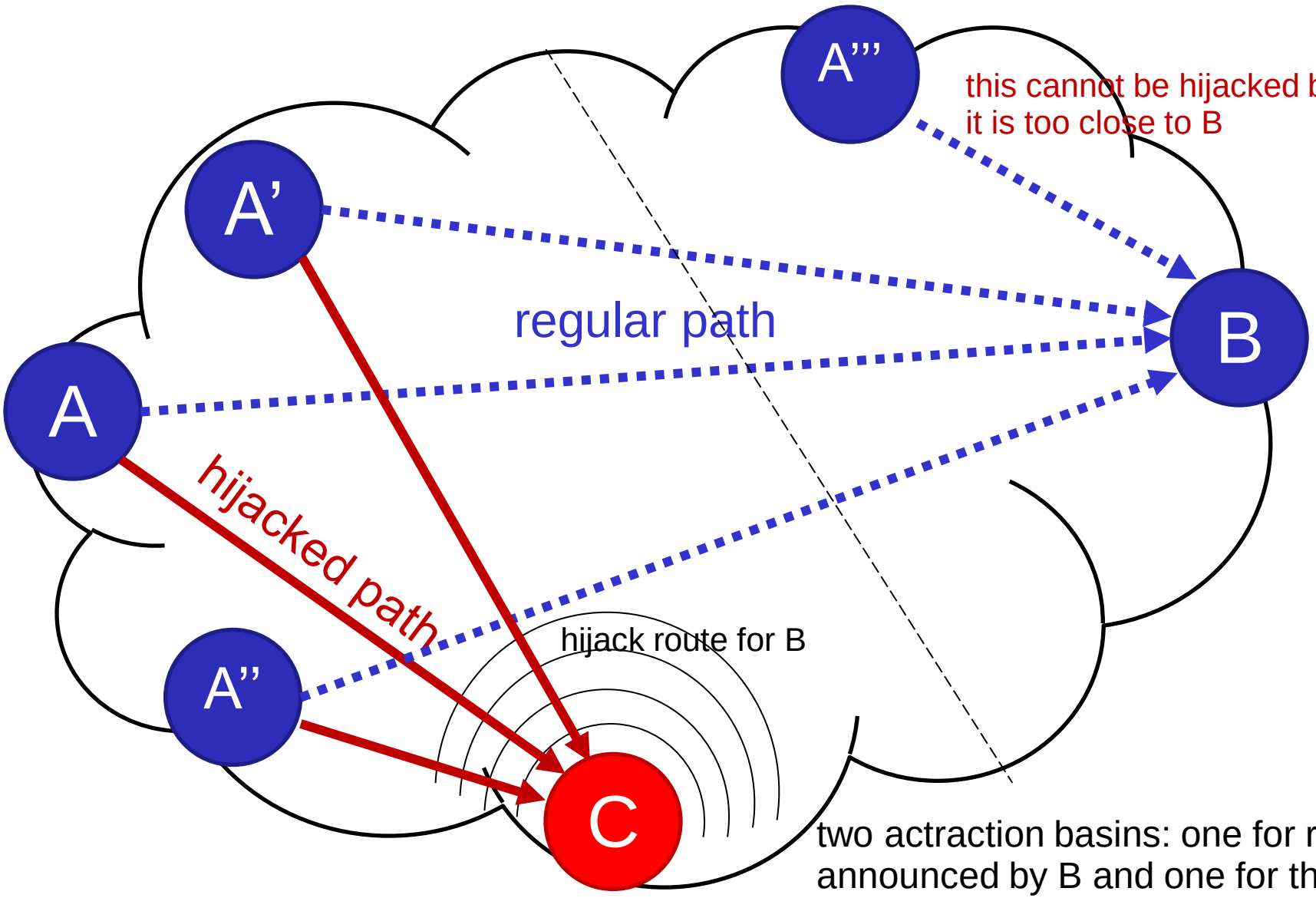
The Internet “biggest security hole”

- obiettivo: MitM interdominio
- dimostrato “in vitro” da Tony Kapela e Alex Pilosov (agosto 2008)
- osservato in “in the wild”
  - https://www.wired.com/2013/12/bgp-hijacking-belarus-iceland/
    https://web.archive.org/web/20140330093454/http://www.renesys.com/2013/11/mitm-internet-hijacking/

standard hijacking in BGP

- l'attaccante C controlla un router BGP collegato ad Internet
- nel routing hijacking tradizionale C non si può fare MitM tra A e B perché non si possono inviare pacchetti alla vittima,
  - infatti questi tornerebbero all'attaccante

standard hijacking in BGP

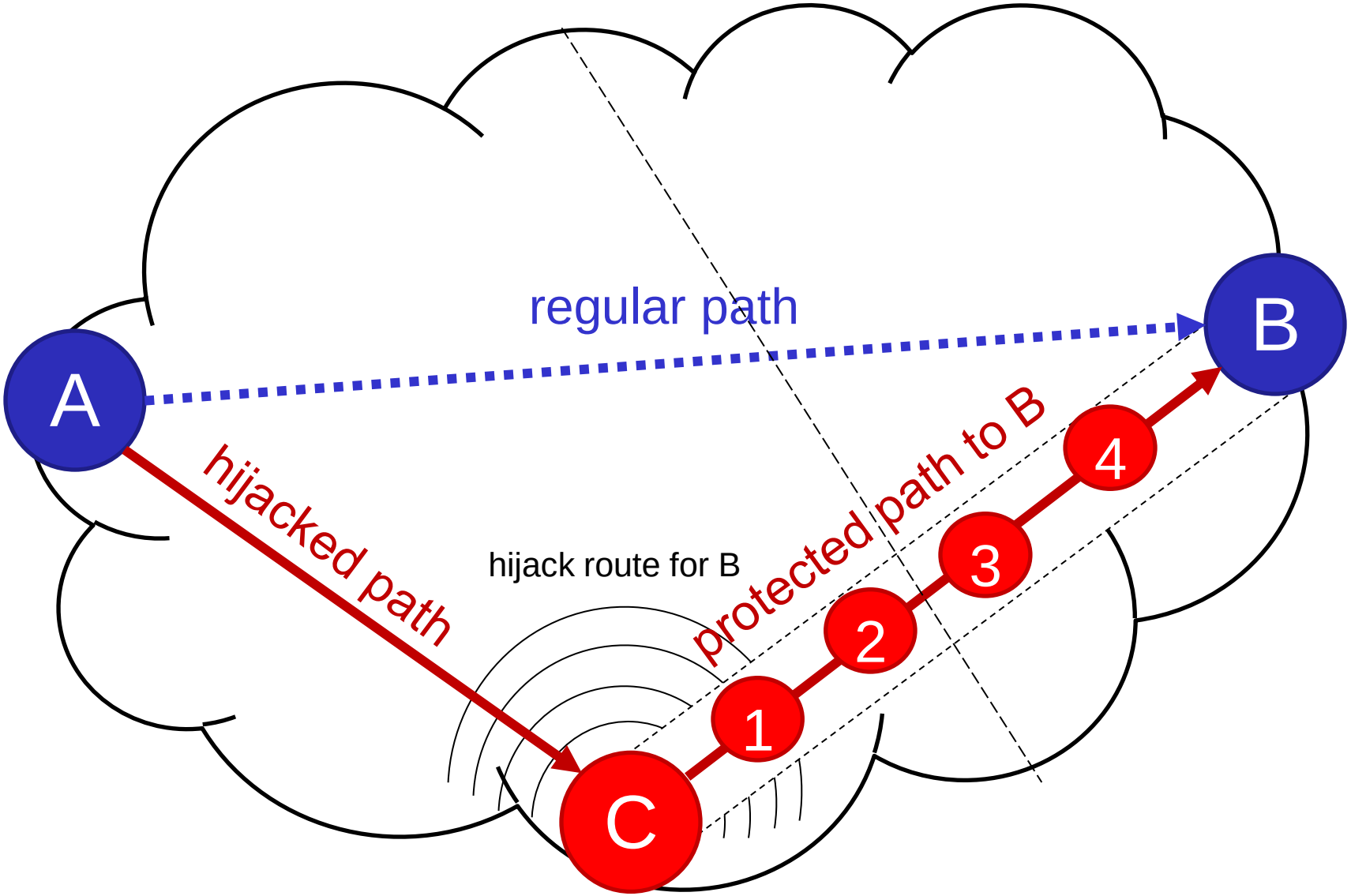


two actraction basins: one for route announced by B and one for the one announced by C

The Internet “biggest security hole”

- l’idea:
  per inviare i pacchetti alla vittima **proteggiamo** i router nel percorso dall’attaccante alla vittima
  - in modo che possano seguire il routing senza hijacking
  - la protezione usa la caratteristica di BGP detta *loop avoidance*
    - gli AS da proteggere sono inseriti nell’AS-path

The Internet “biggest security hole”



The Internet “biggest security hole”

- procedura per MitM del traffico destinato alla vittima
  1. traceroute verso la vittima
  2. trovare tutti gli AS dagli indirizzi ip del traceroute (facile: mtr -z <ipdest>)
  3. fare l’hijacking inserendo nell’AS path tutti gli AS calcolati al punto 2
     - BGP loop avoidance fa si che tali AS non ricevano (o ignorino l’annuncio)
  4. Inserire una rotta statica verso il primo hop nella direzione della vittima

- tutto il bacino di attrazione manda il traffico all’attaccante che poi può inoltrarlo alla vittima