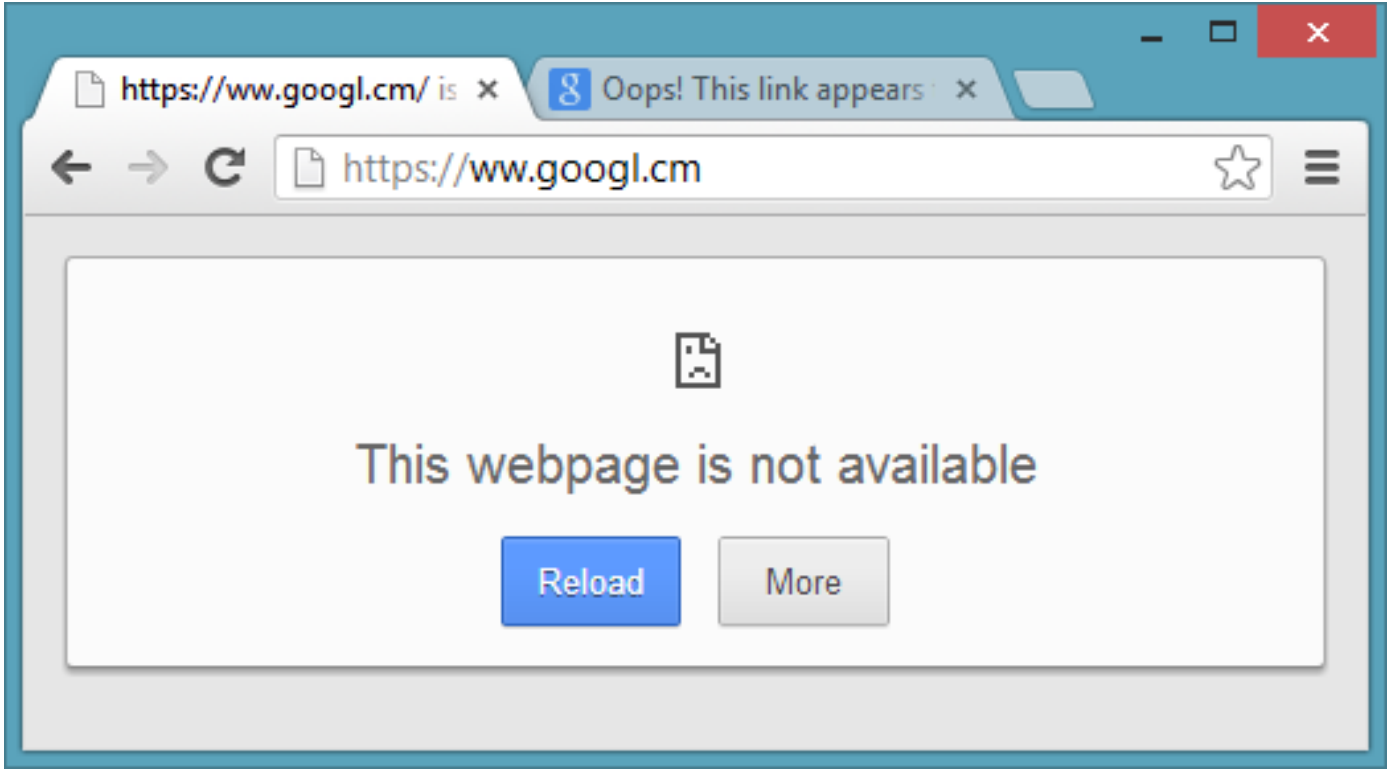
News

- Office meeting moved to Thursday 13.15 for this week
- No questions on World Cup qualification

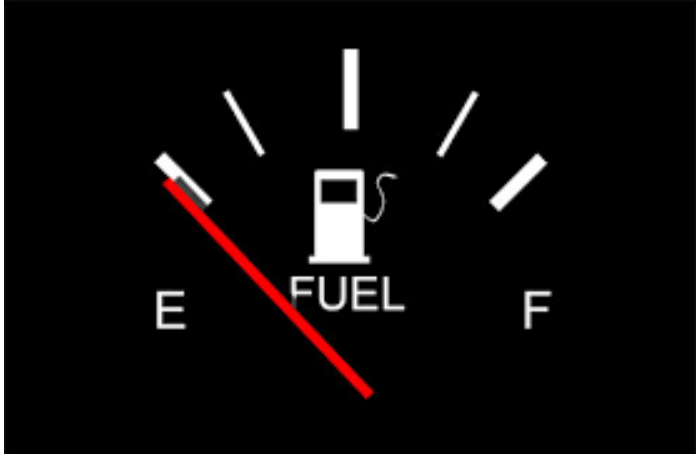
Roberto Guanciale

Computer Security DD2395

Denial of Service


https://ww.googl.cm/ is
Oops! This link appears
https://ww.googl.cm
This webpage is not available
Reload
More

Denial **O**f **S**ervice



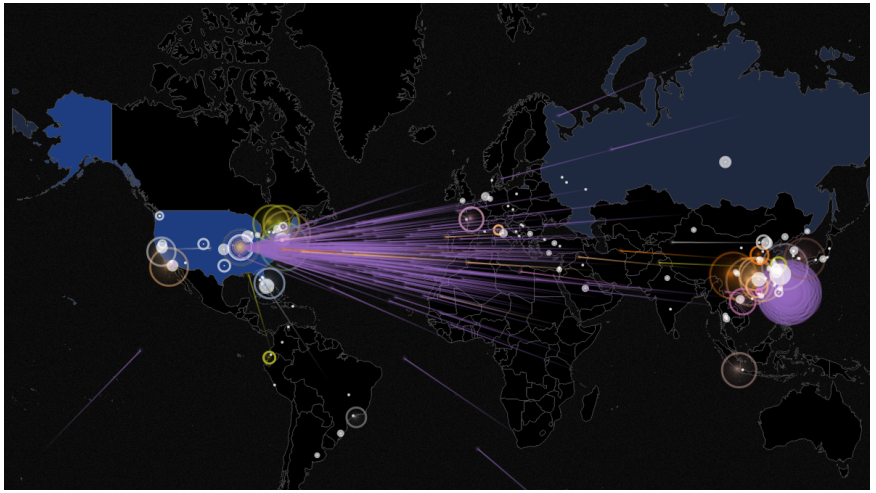
prevents or impairs the authorized use of

- networks
- systems
- applications

By **exhausting** or **damaging** resources

Denial Of Service

- Networked
- Based on SW vulnerabilities
- Physical



Denial Of Service

- Networked
- Based on SW vulnerabilities
- Physical

Denial Of Service

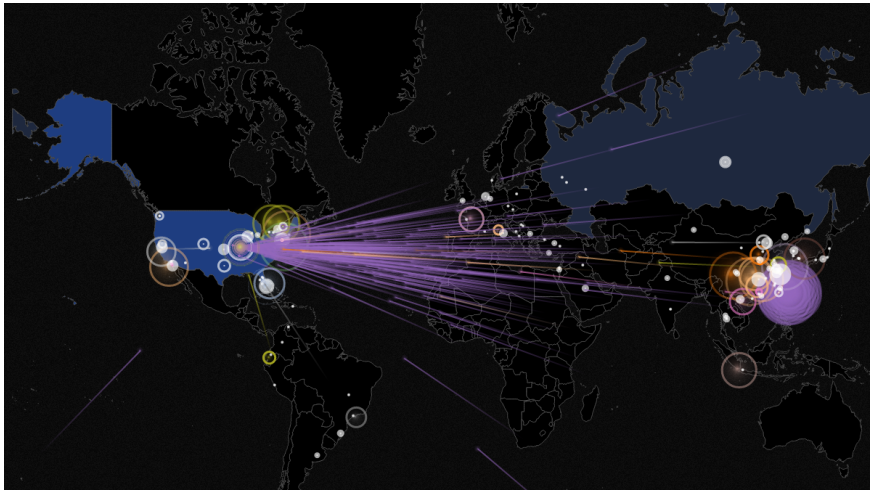
- Networked
- Based on SW vulnerabilities
- Physical

Denial Of Service

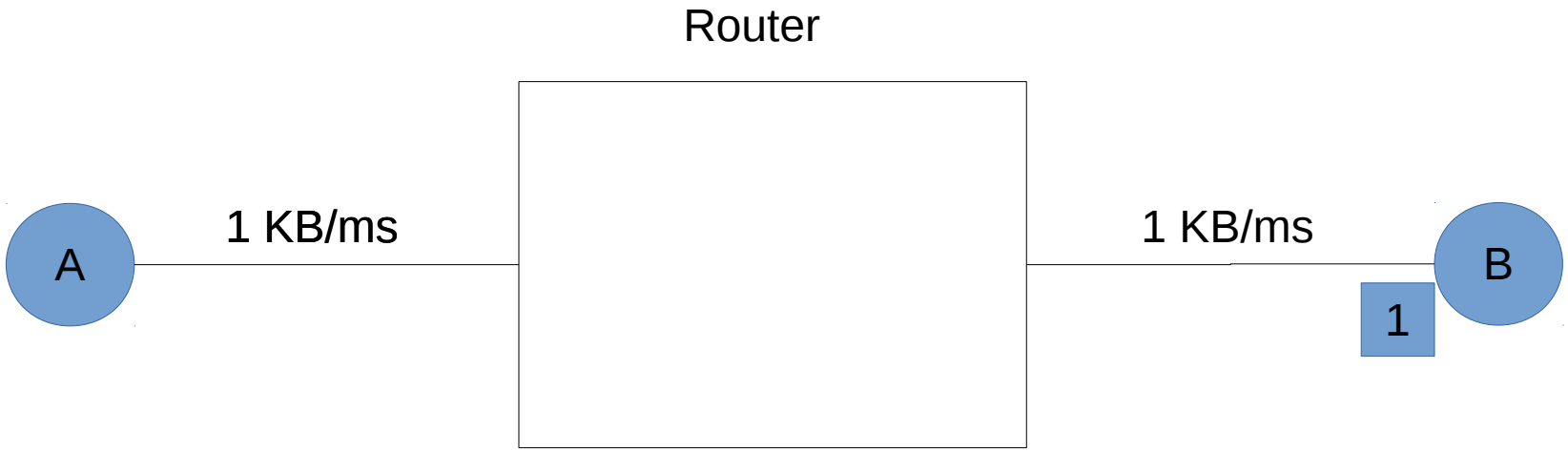
- Networked
- Based on SW vulnerabilities
- Physical

Denial Of Service

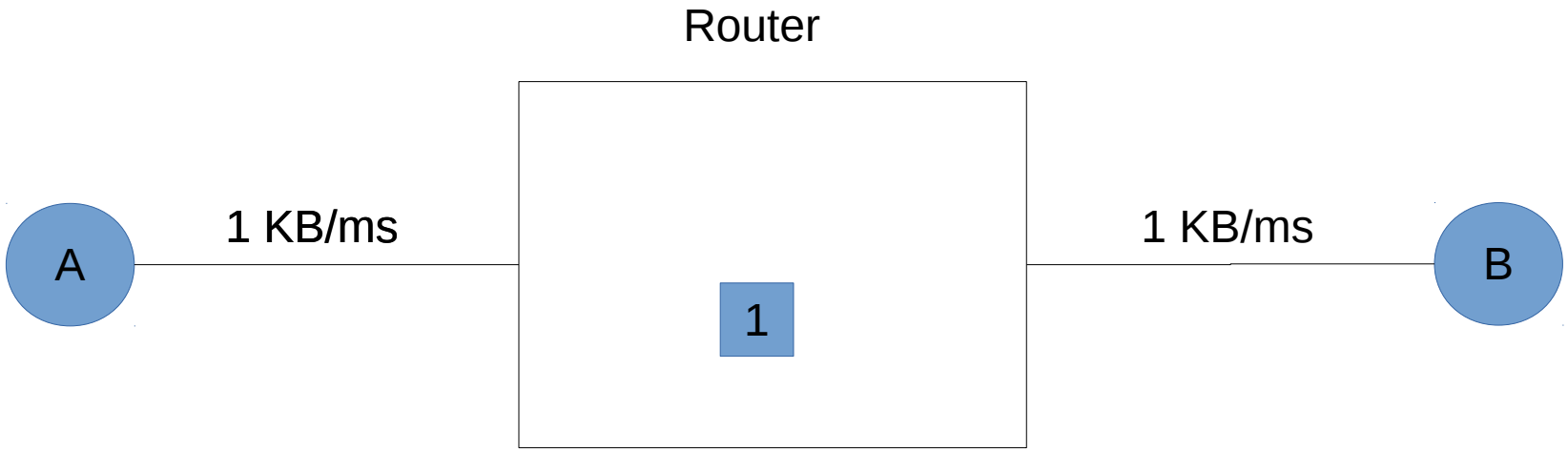
- **Networked**
- Based on SW vulnerabilities
- Physical



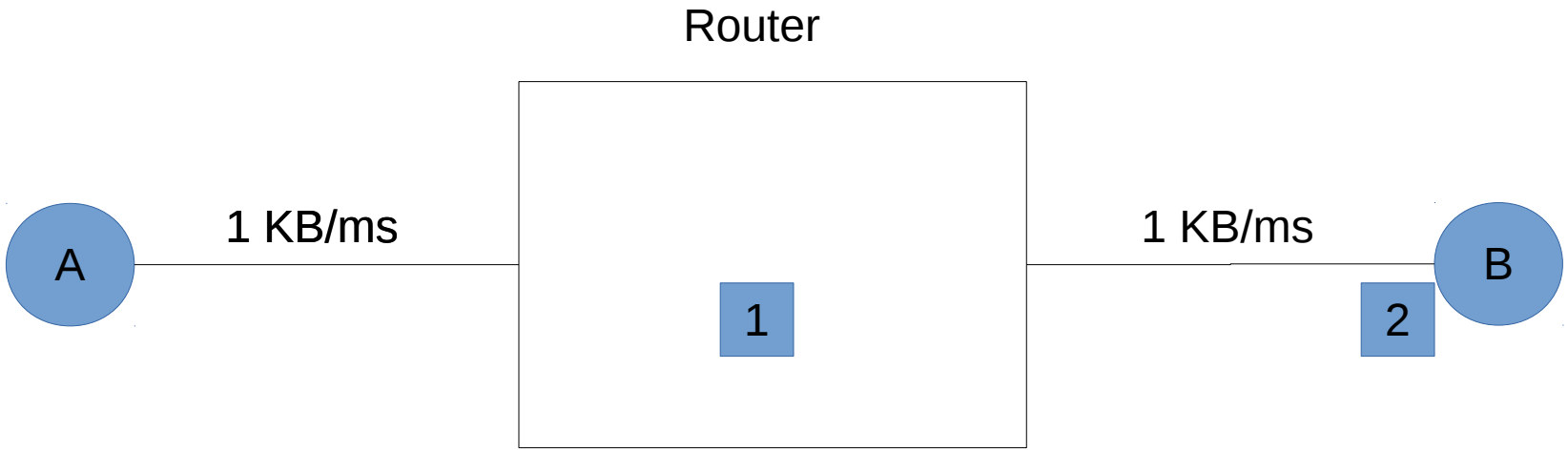
ICMP (ping) flood



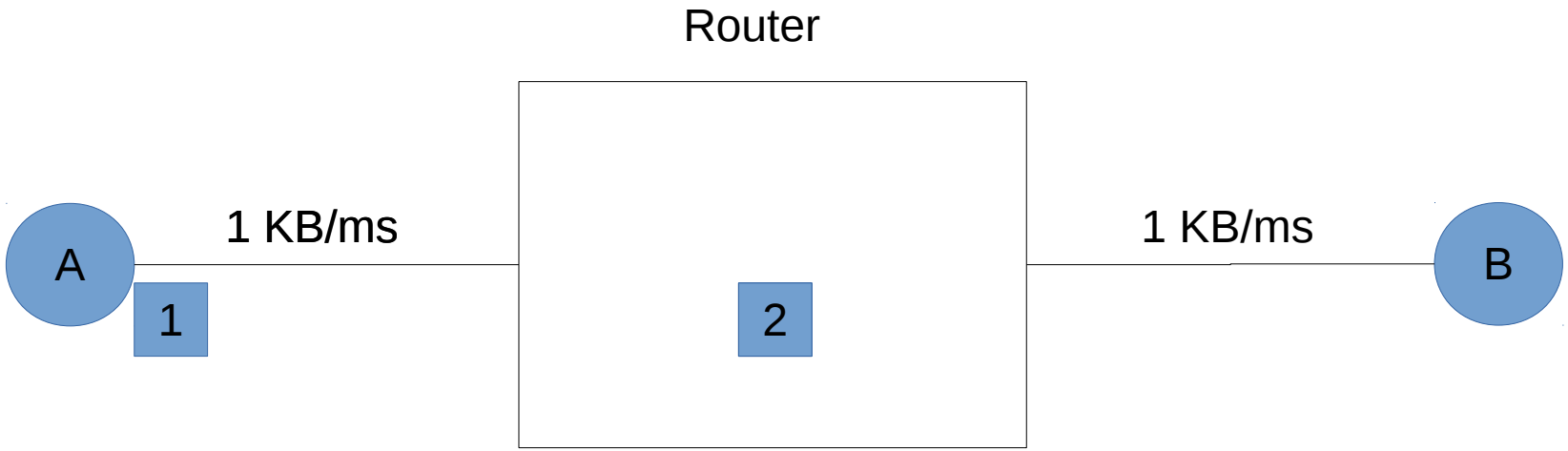
ICMP (ping) flood



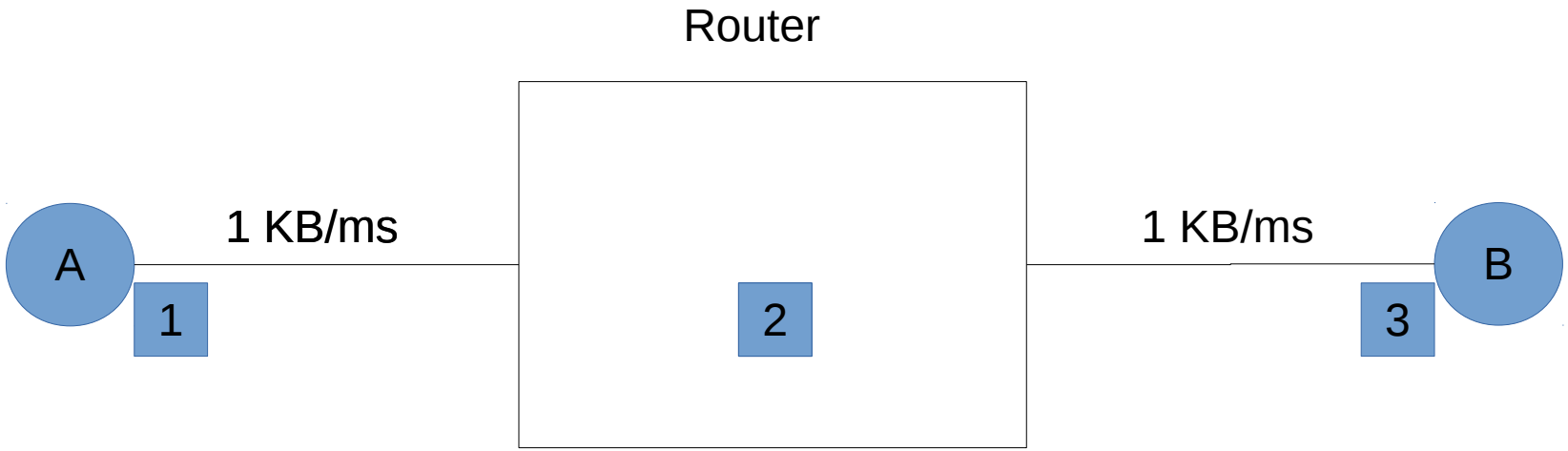
ICMP (ping) flood



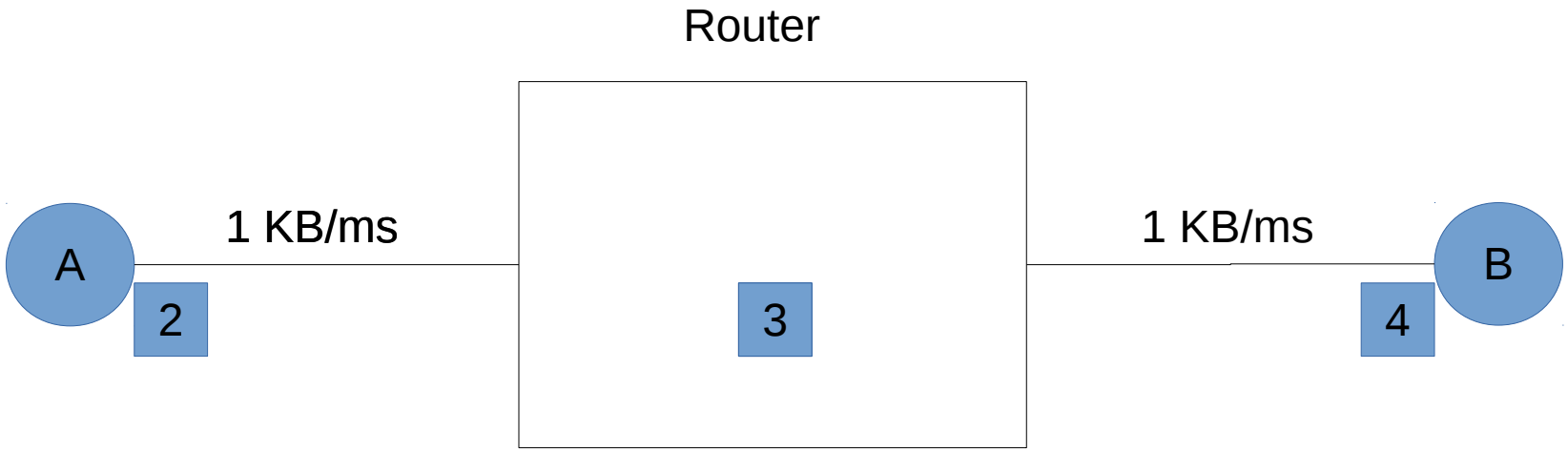
ICMP (ping) flood



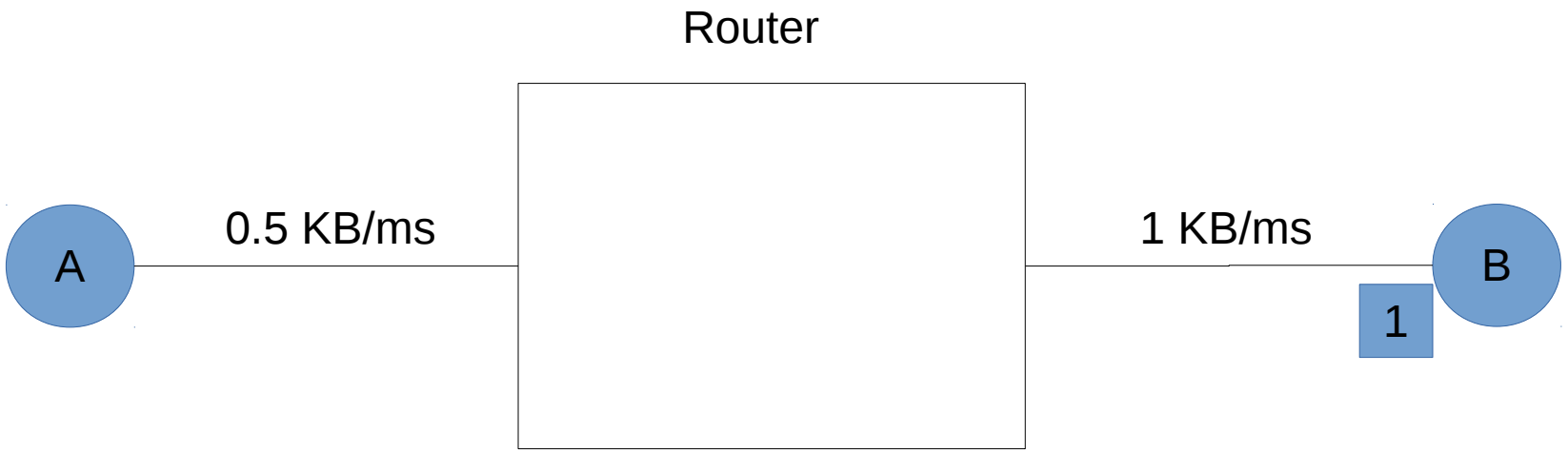
ICMP (ping) flood



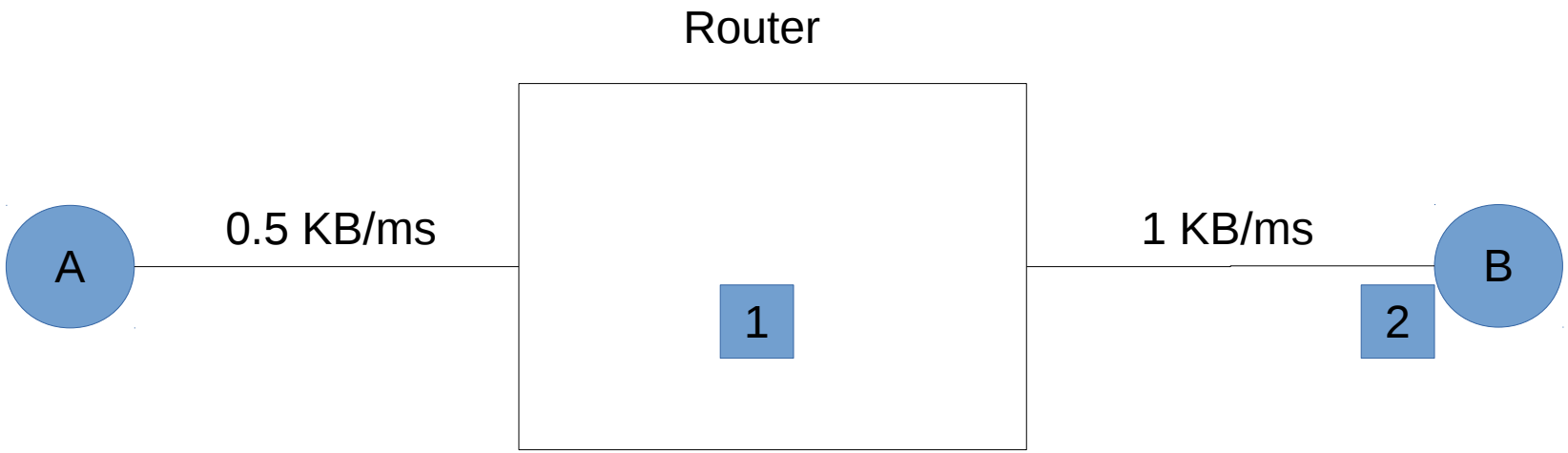
ICMP (ping) flood



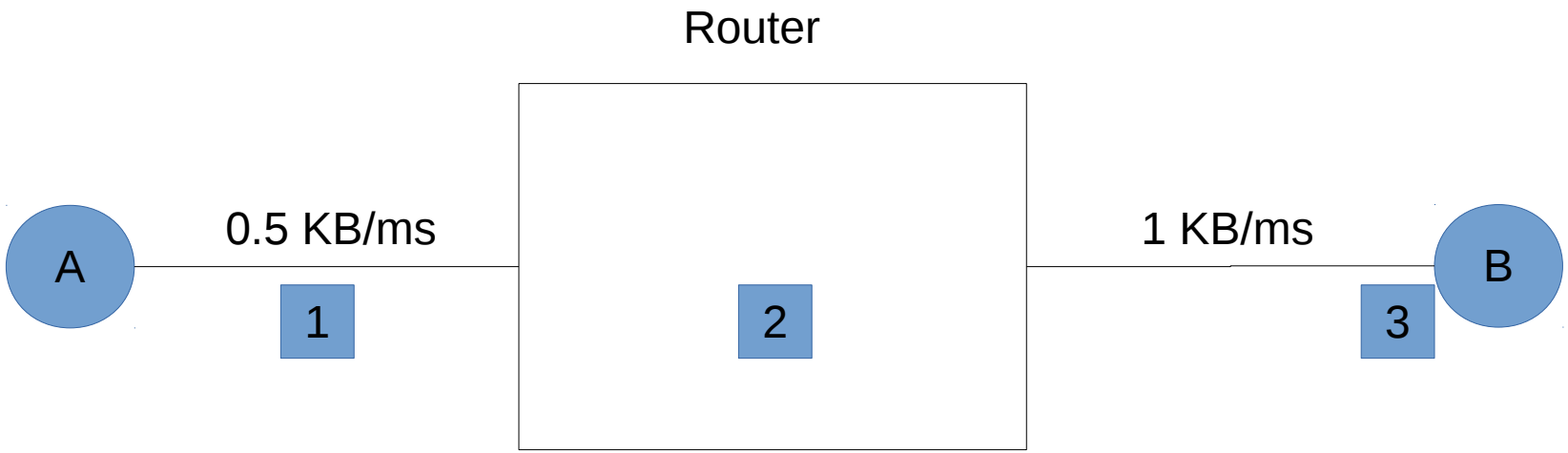
ICMP (ping) flood



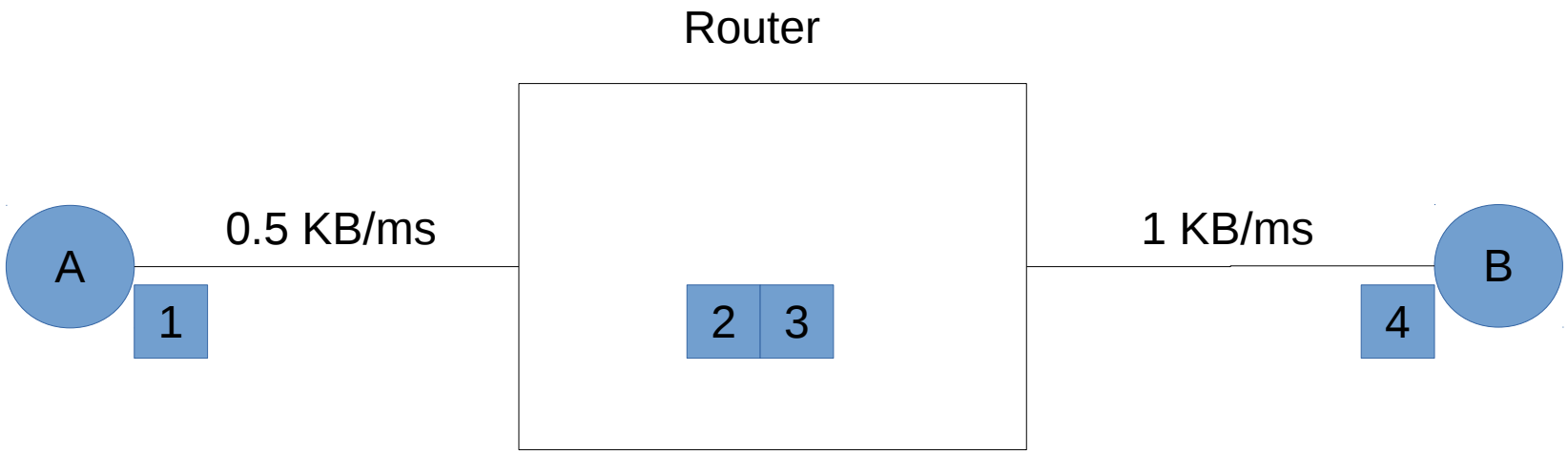
ICMP (ping) flood



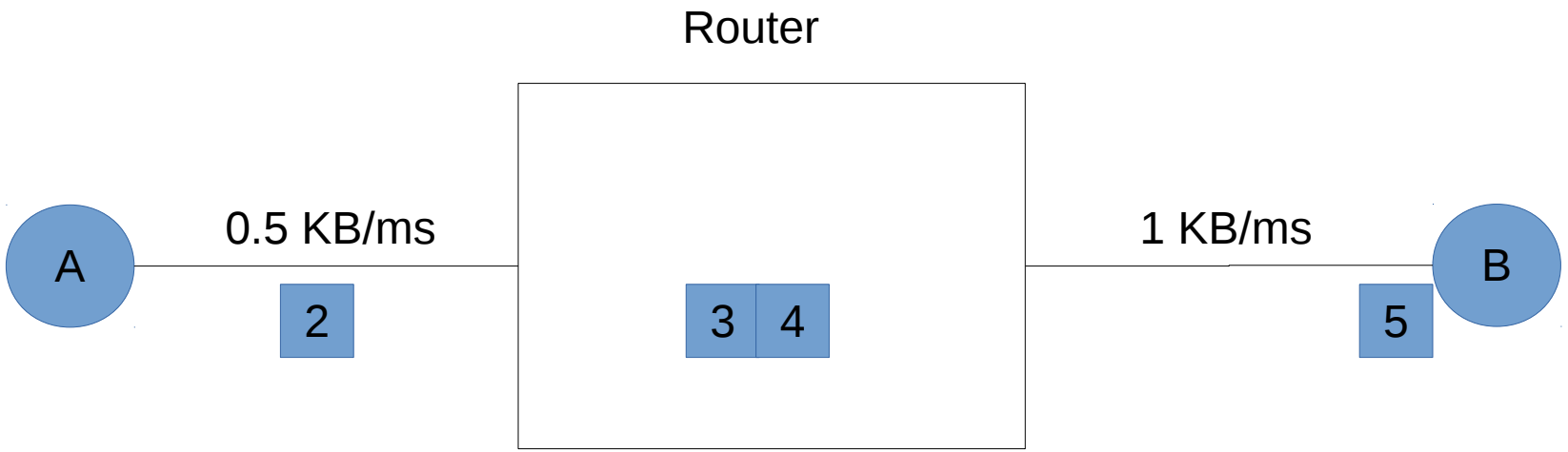
ICMP (ping) flood



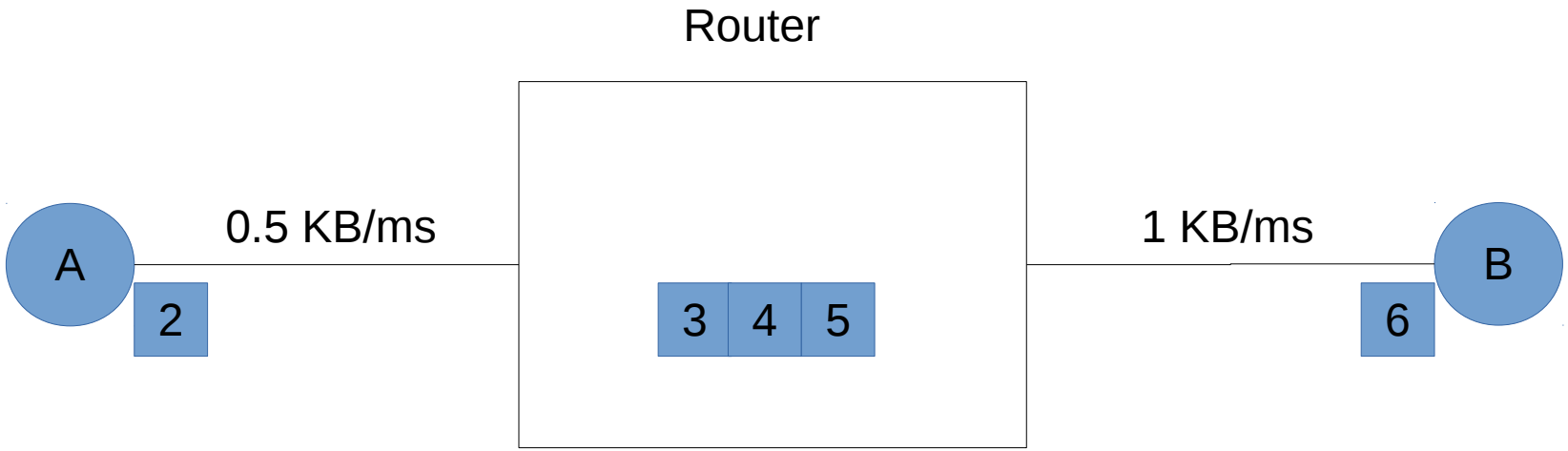
ICMP (ping) flood



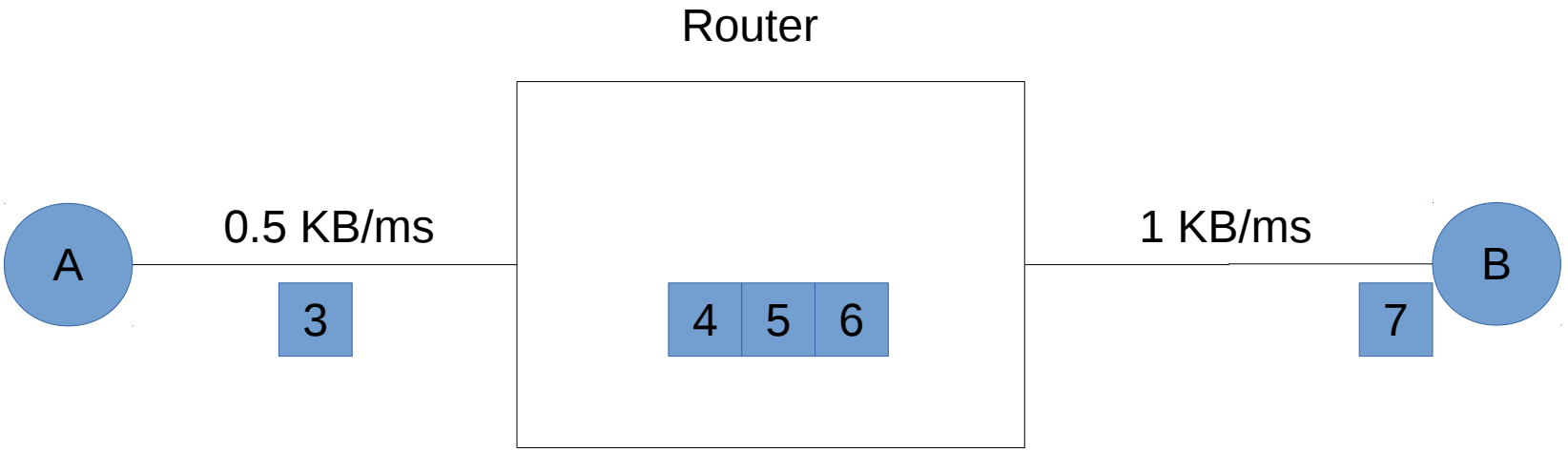
ICMP (ping) flood



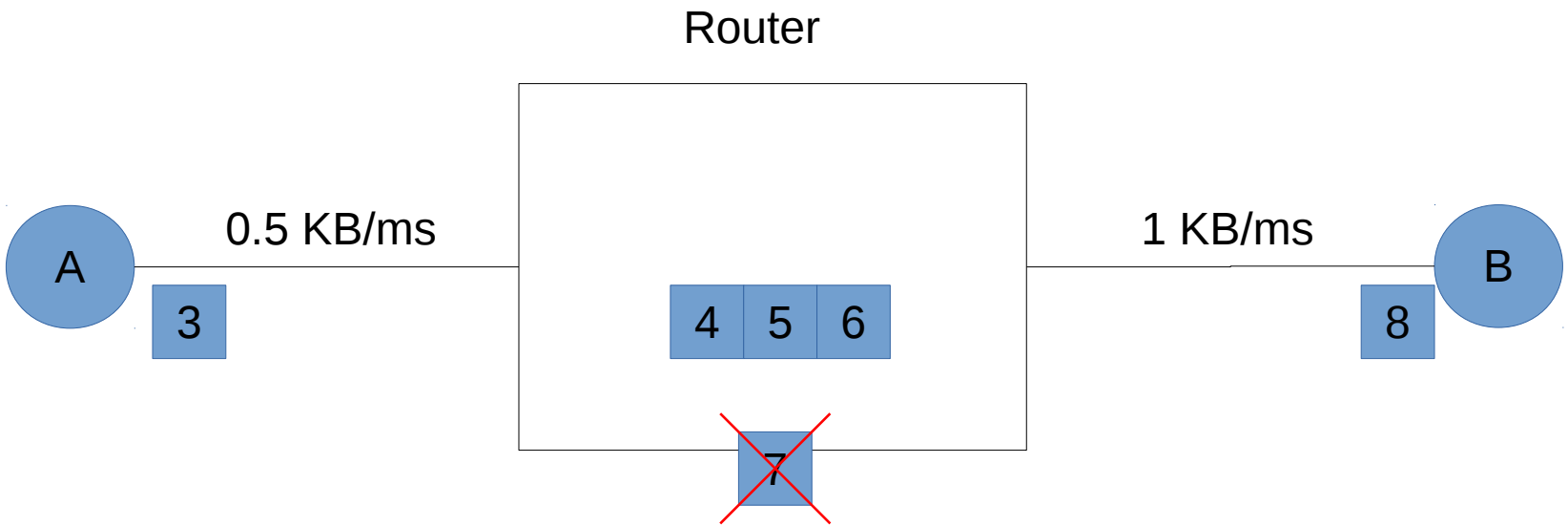
ICMP (ping) flood



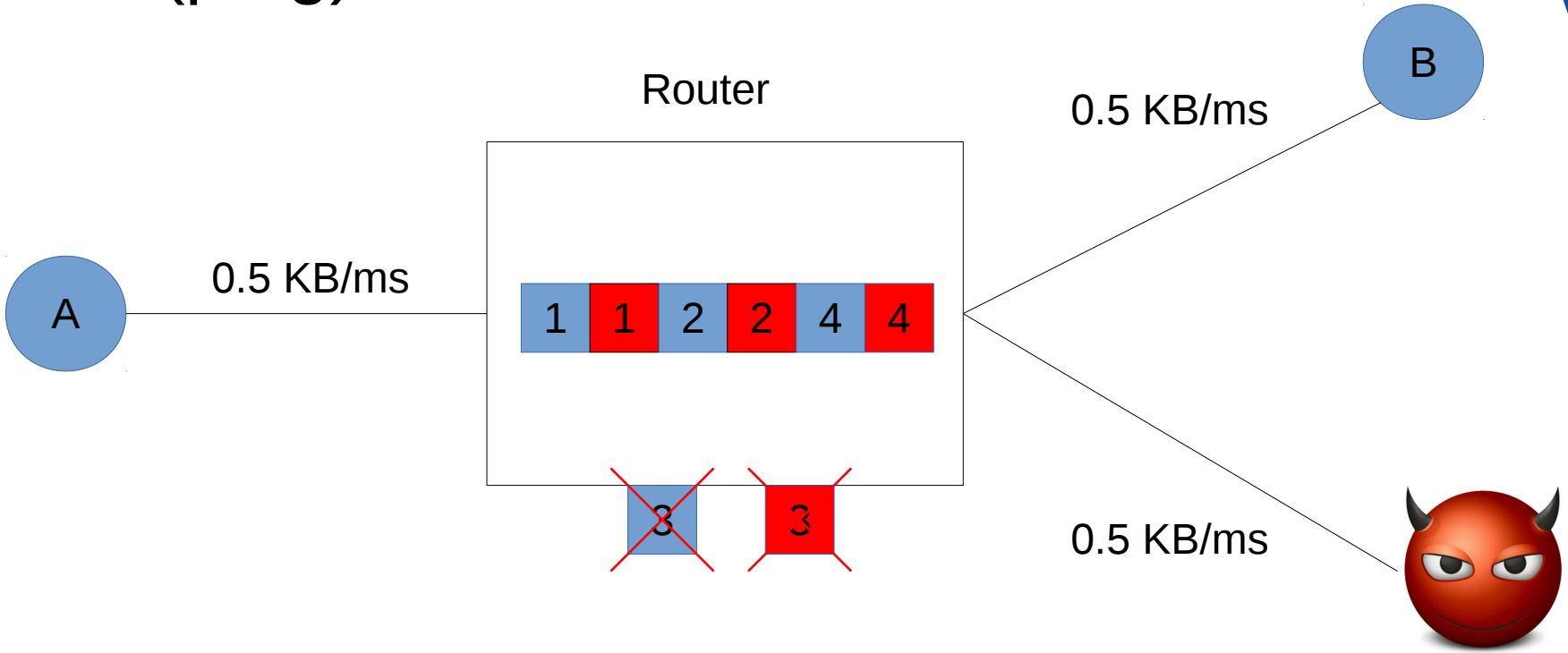
ICMP (ping) flood



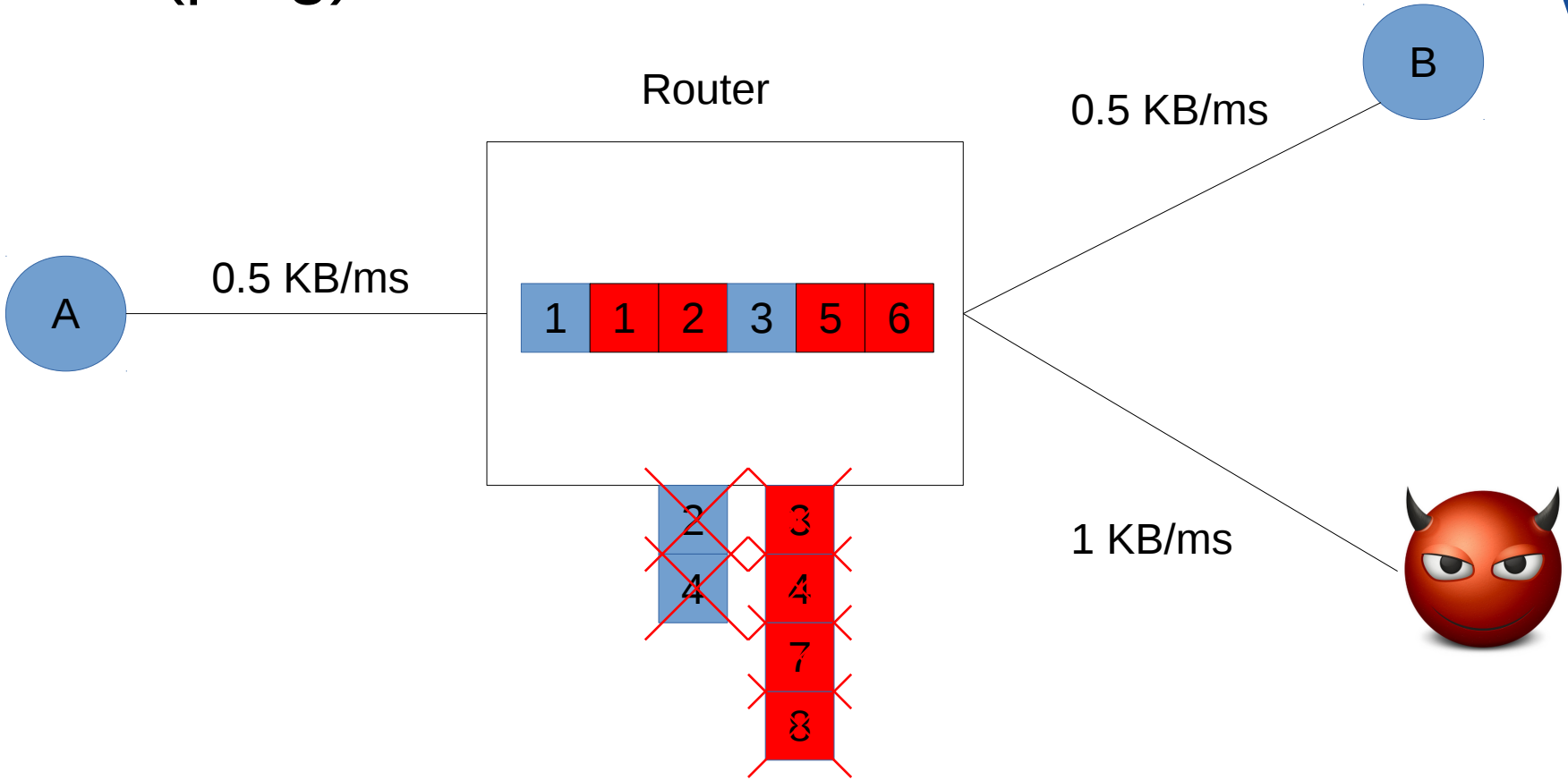
ICMP (ping) flood



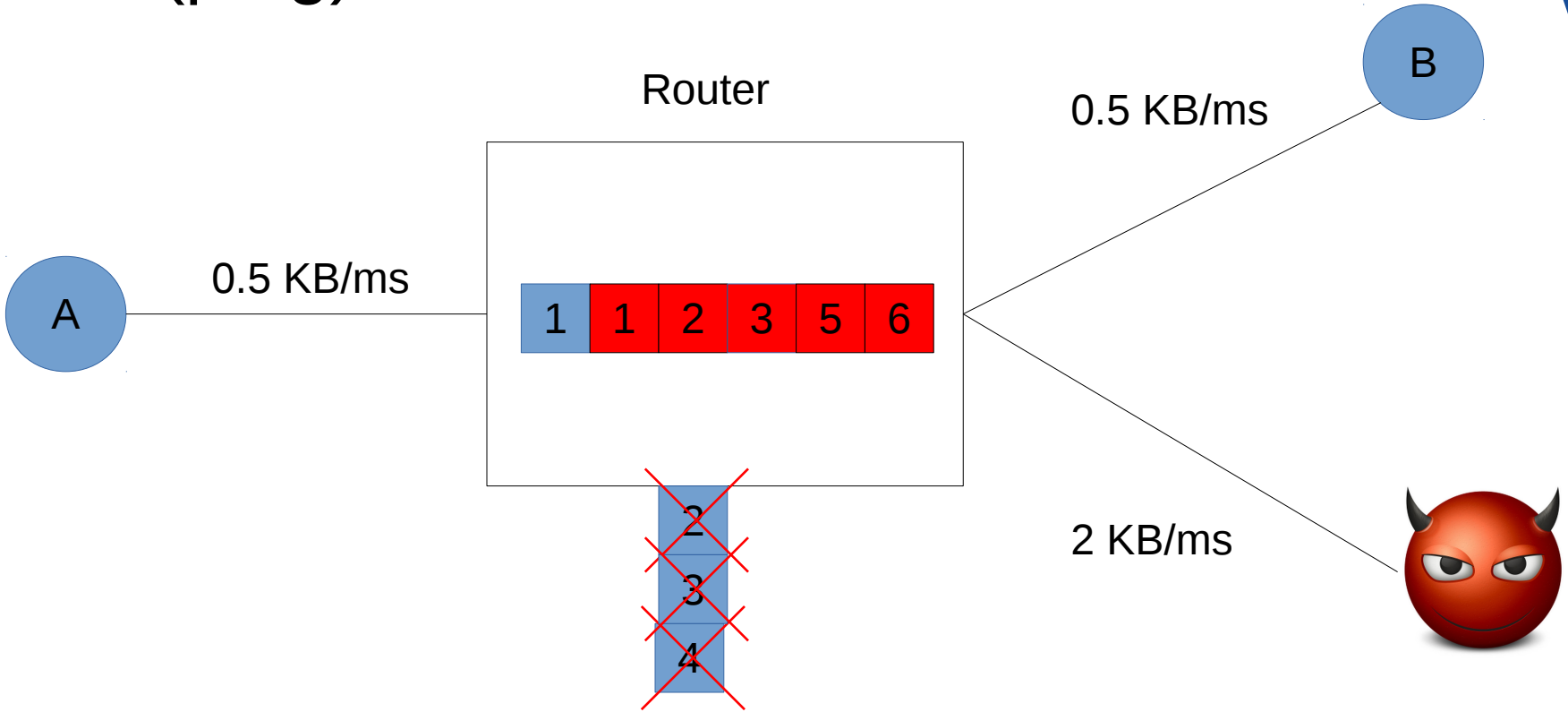
ICMP (ping) flood



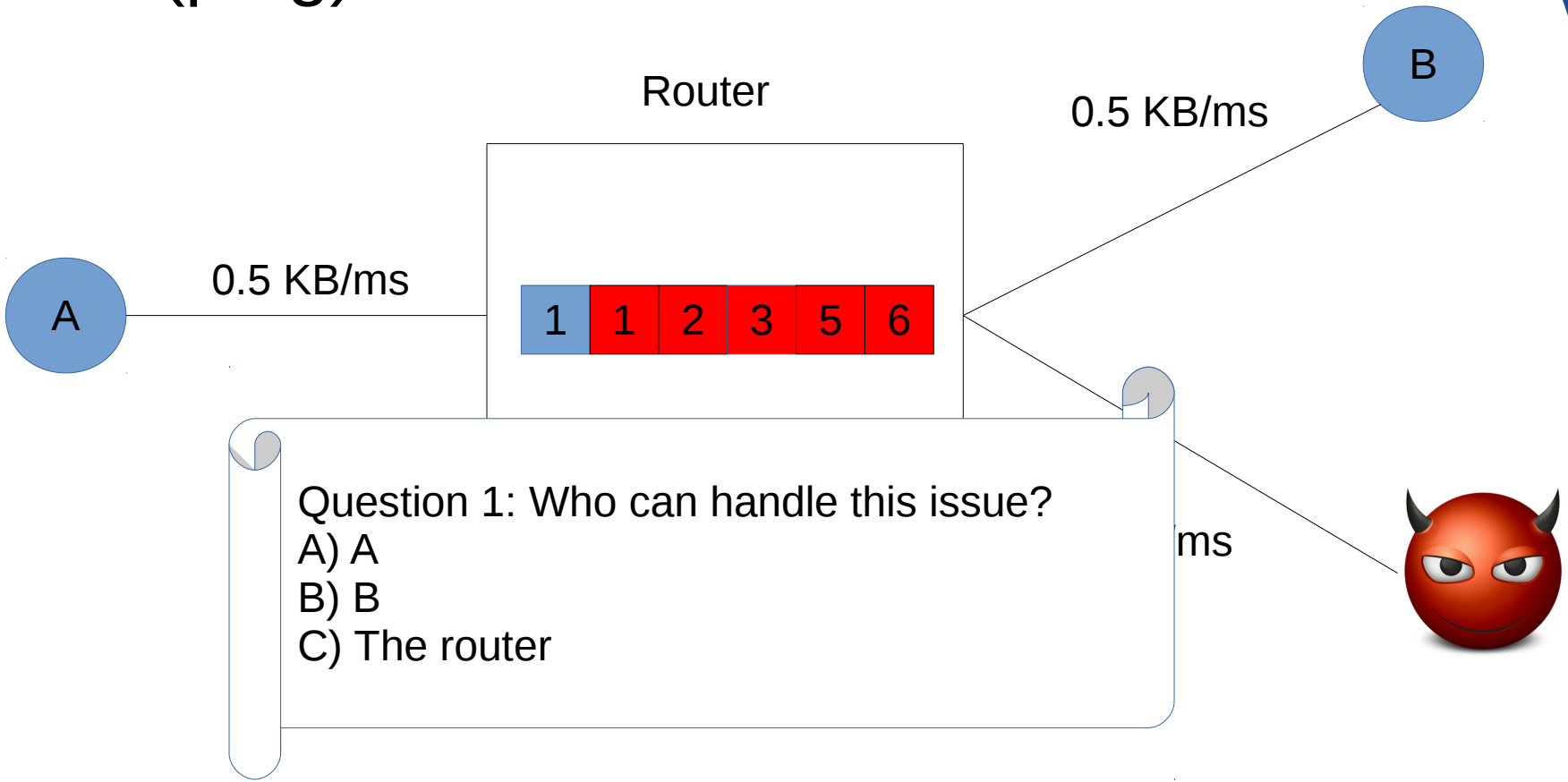
ICMP (ping) flood



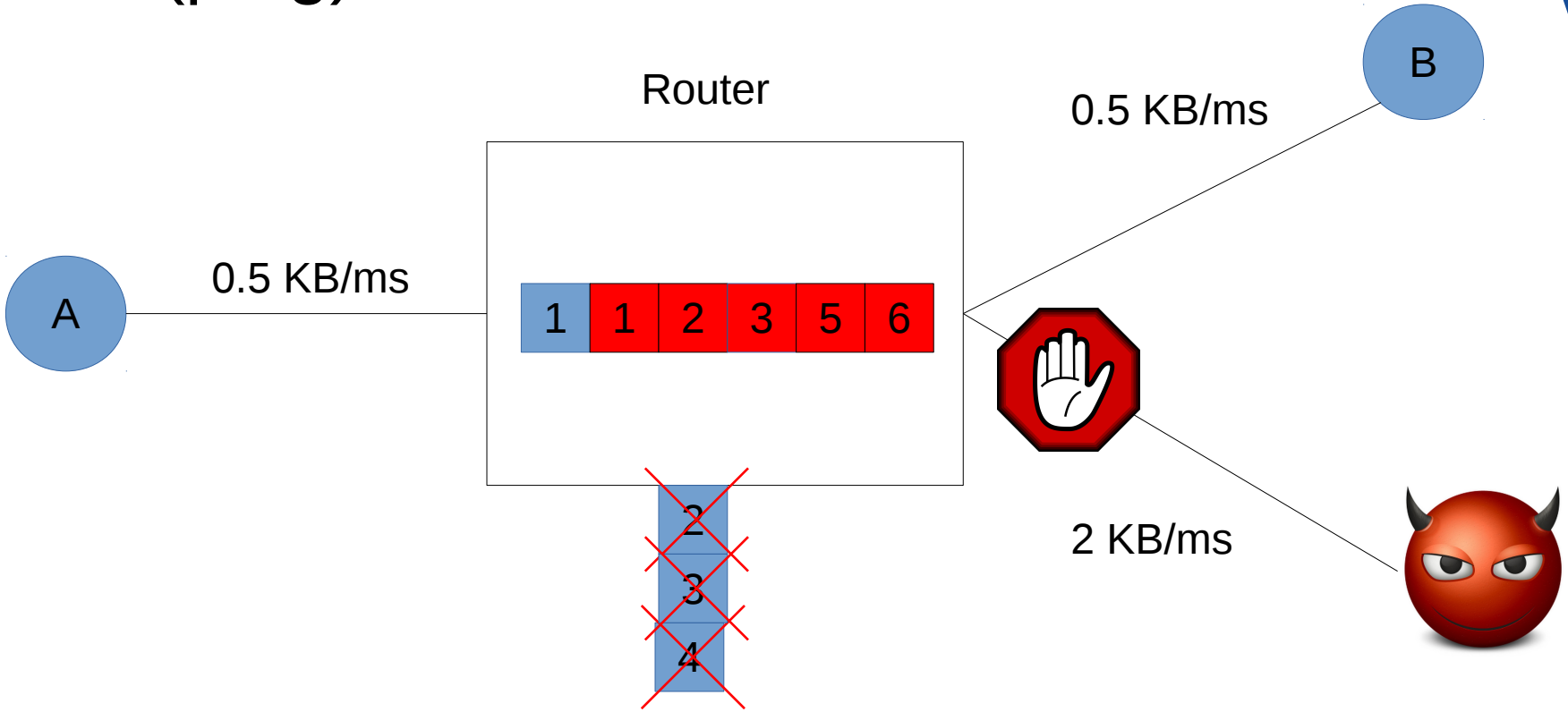
ICMP (ping) flood



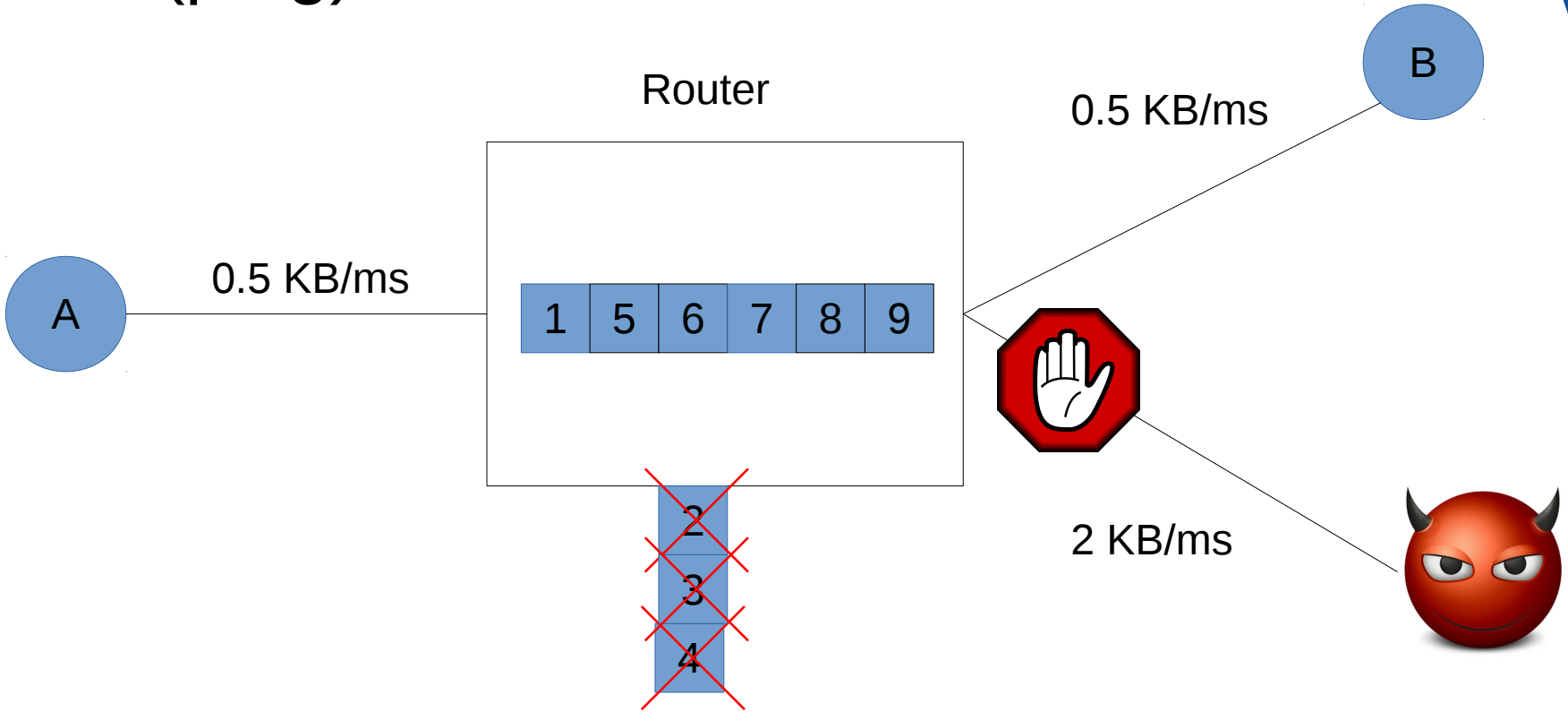
ICMP (ping) flood



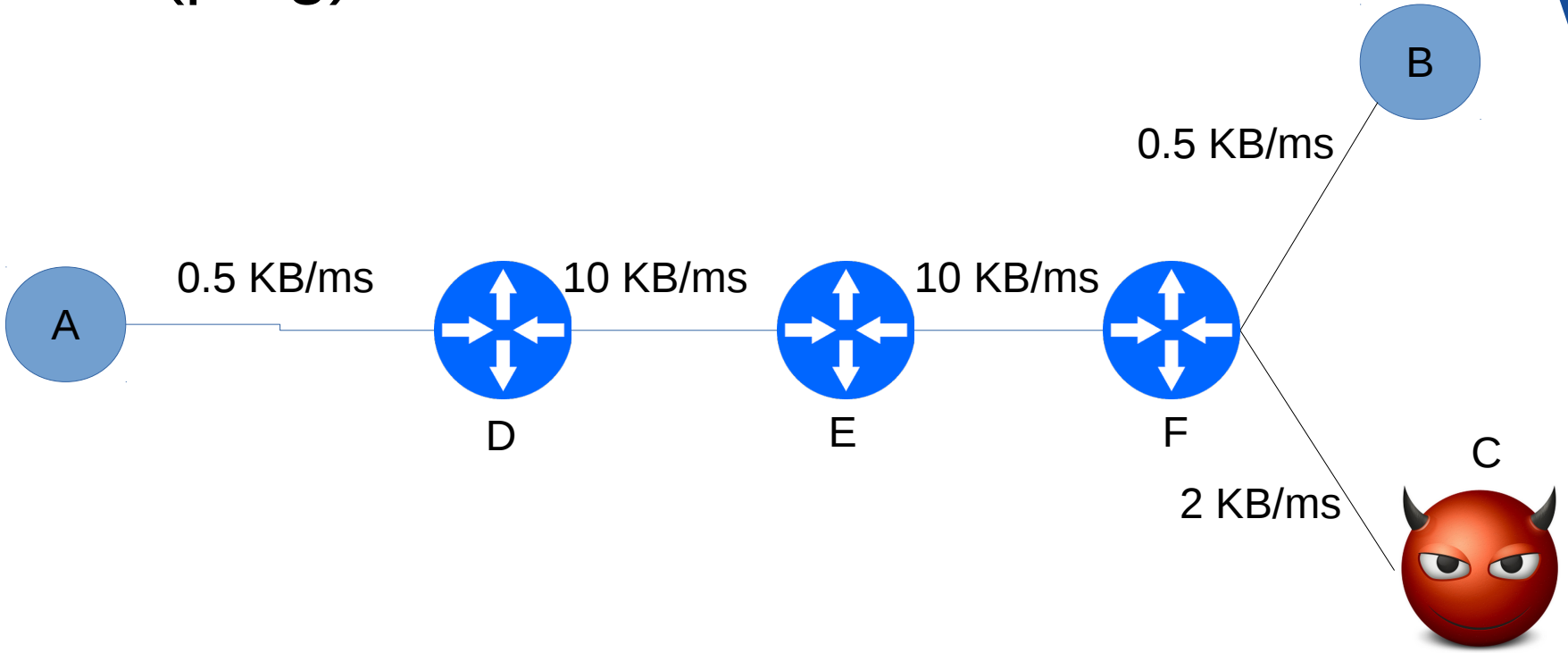
ICMP (ping) flood



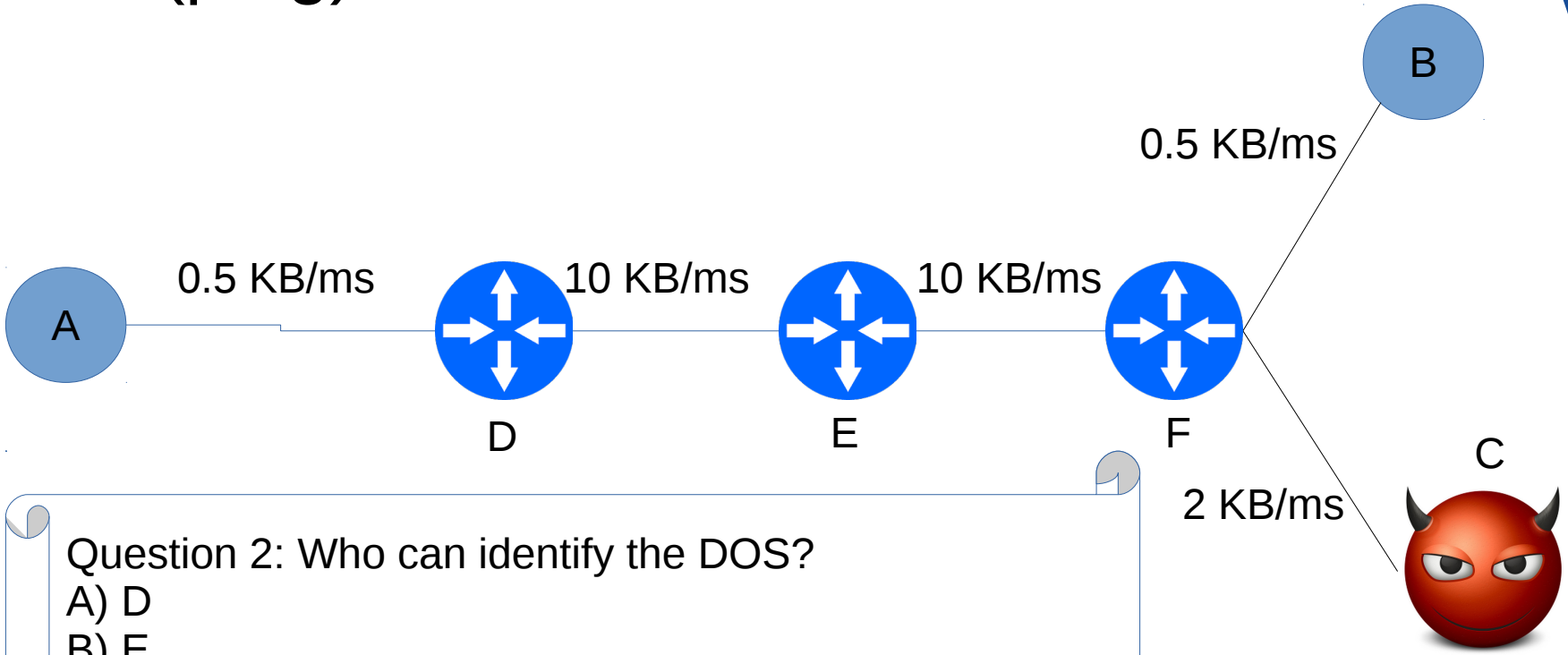
ICMP (ping) flood



ICMP (ping) flood



ICMP (ping) flood



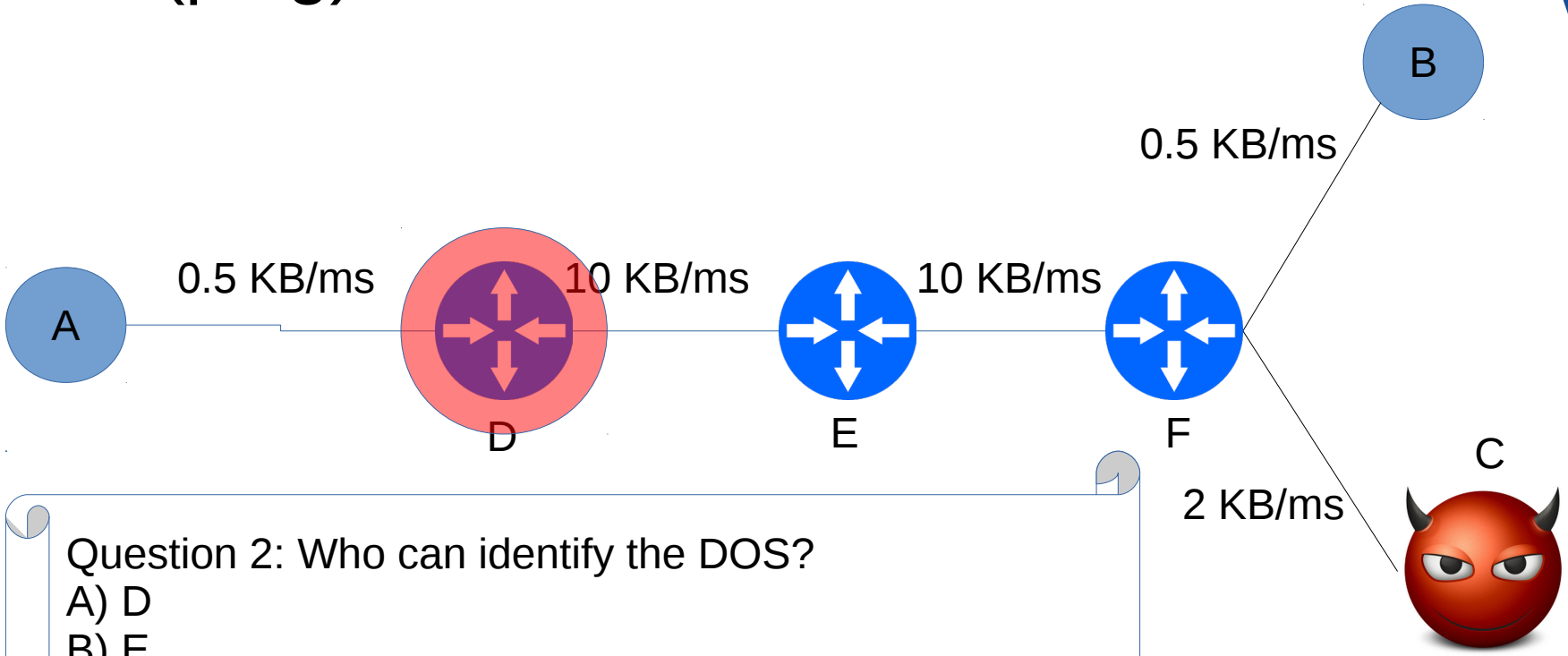
Question 2: Who can identify the DOS?

A) D

B) E

C) F

ICMP (ping) flood



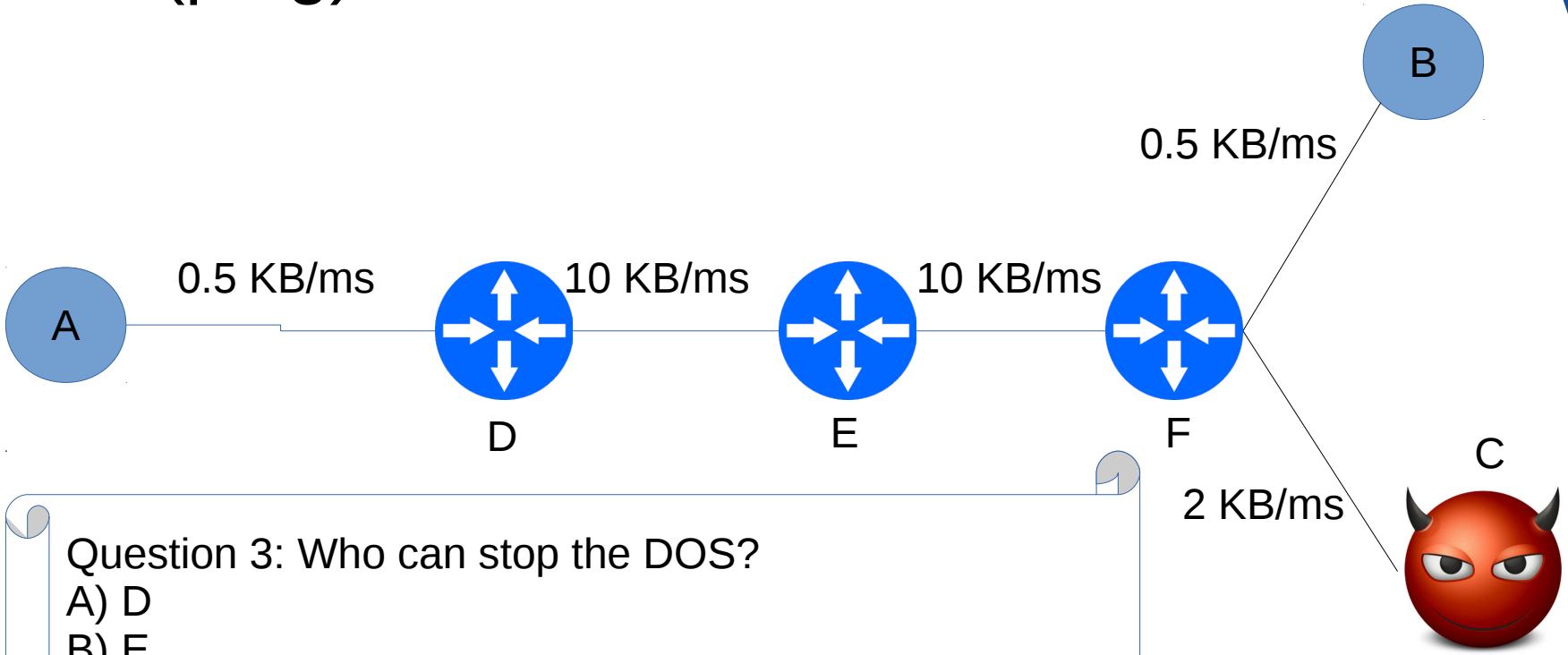
Question 2: Who can identify the DOS?

A) D

B) E

C) F

ICMP (ping) flood



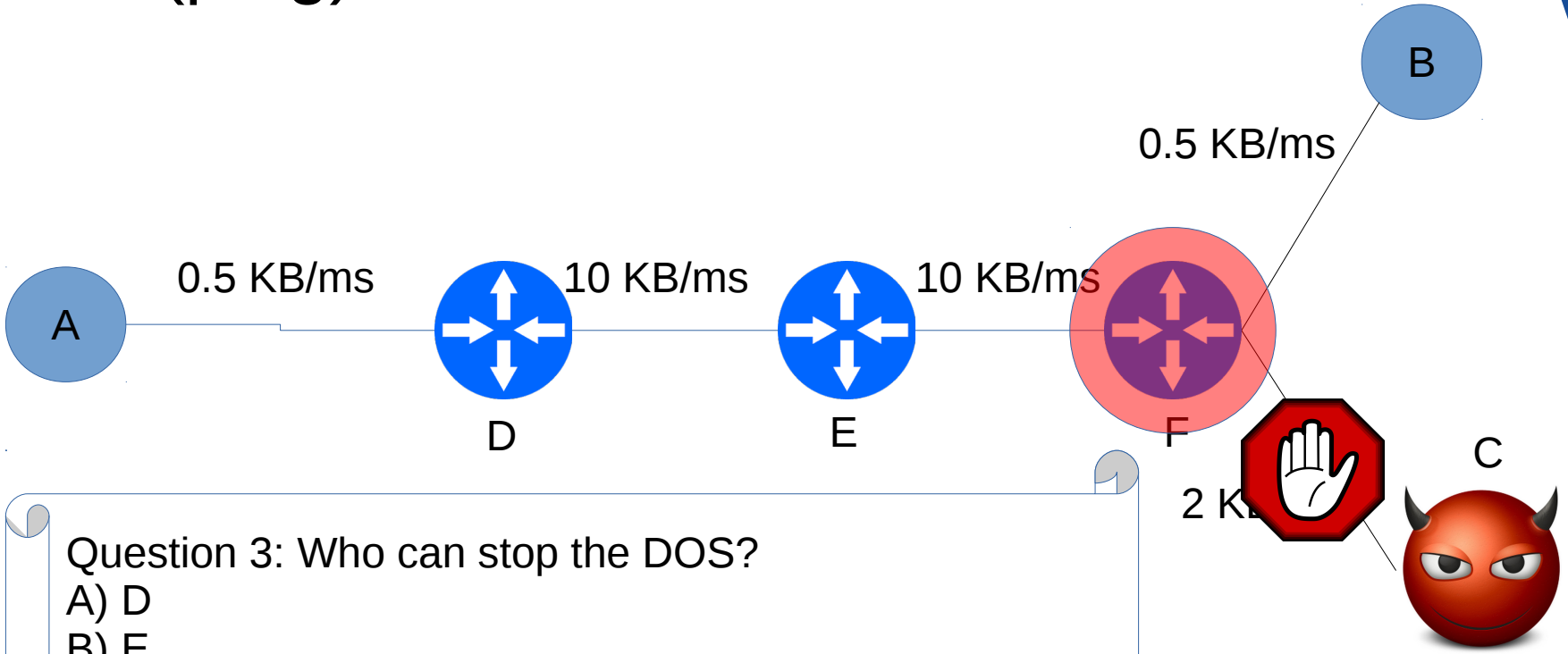
Question 3: Who can stop the DOS?

A) D

B) E

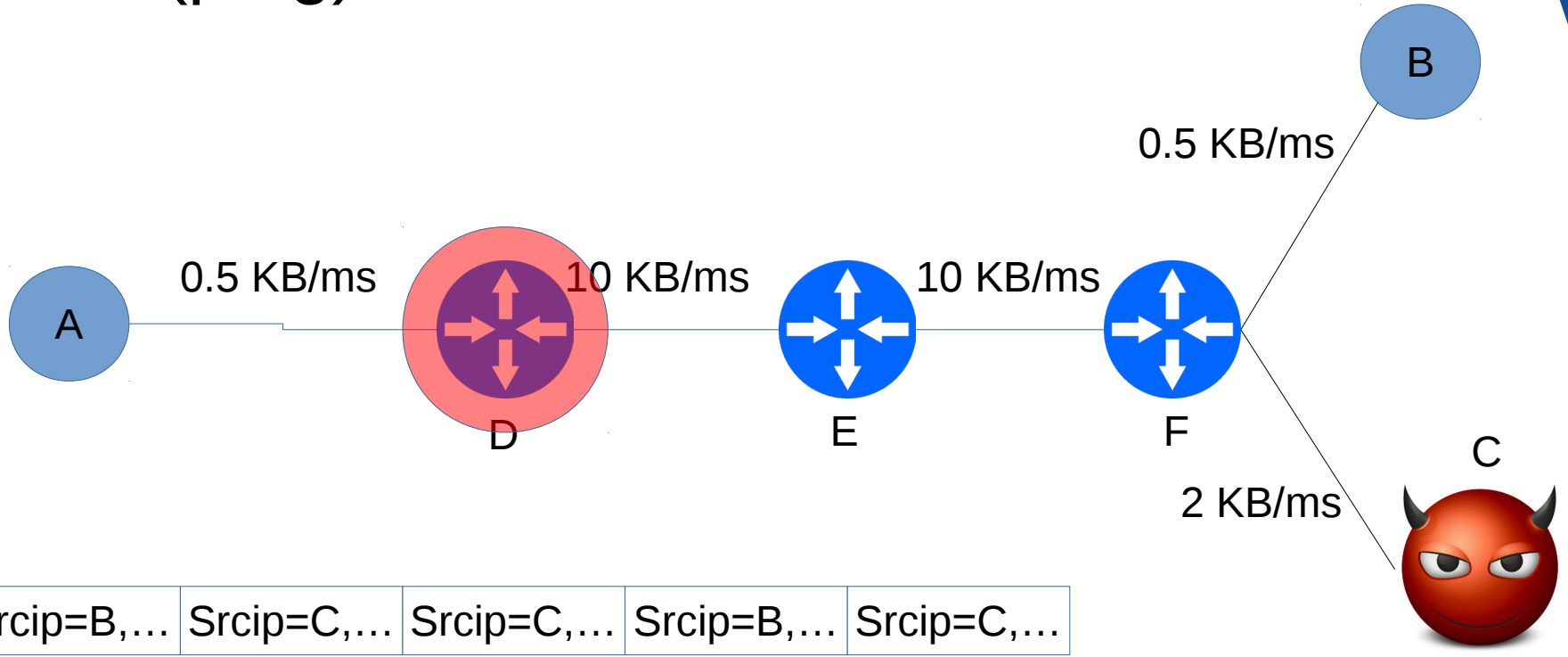
C) F

ICMP (ping) flood



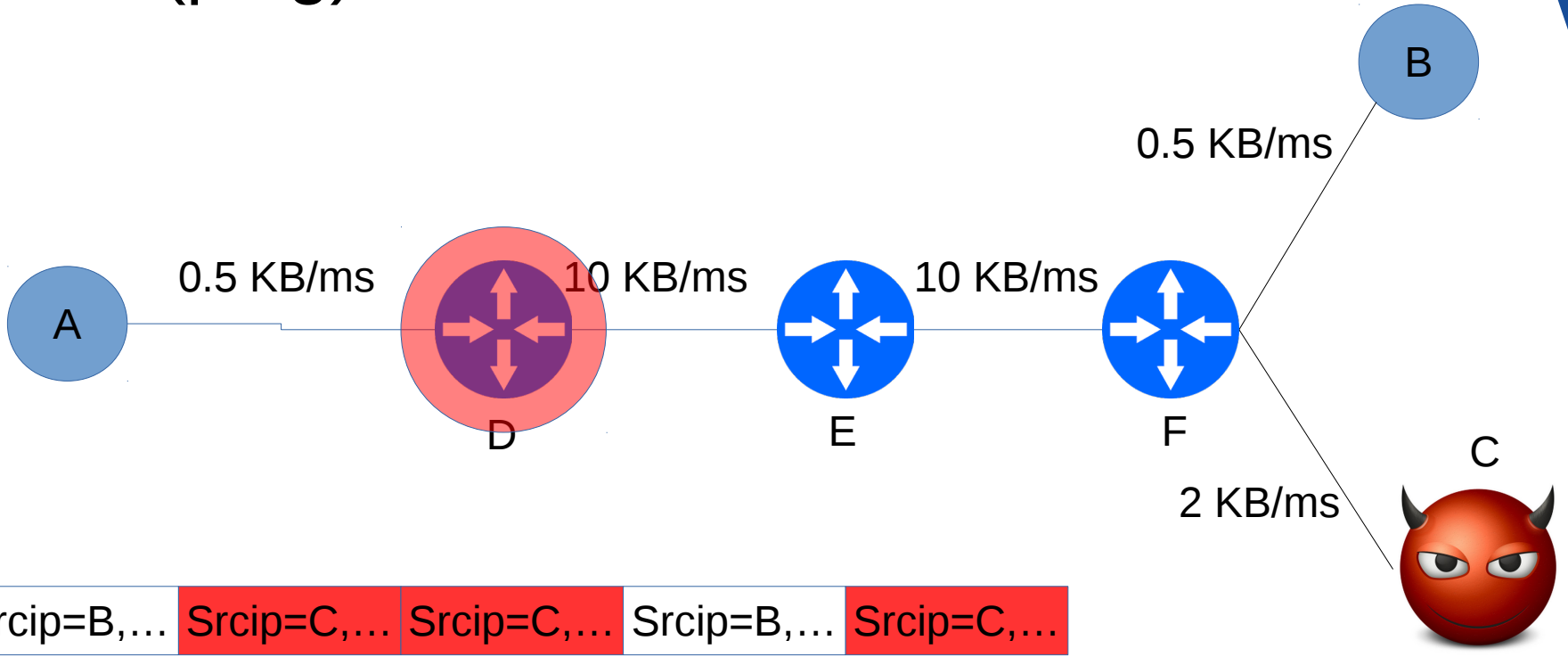
Question 3: Who can stop the DOS?
A) D
B) E
C) F

ICMP (ping) flood

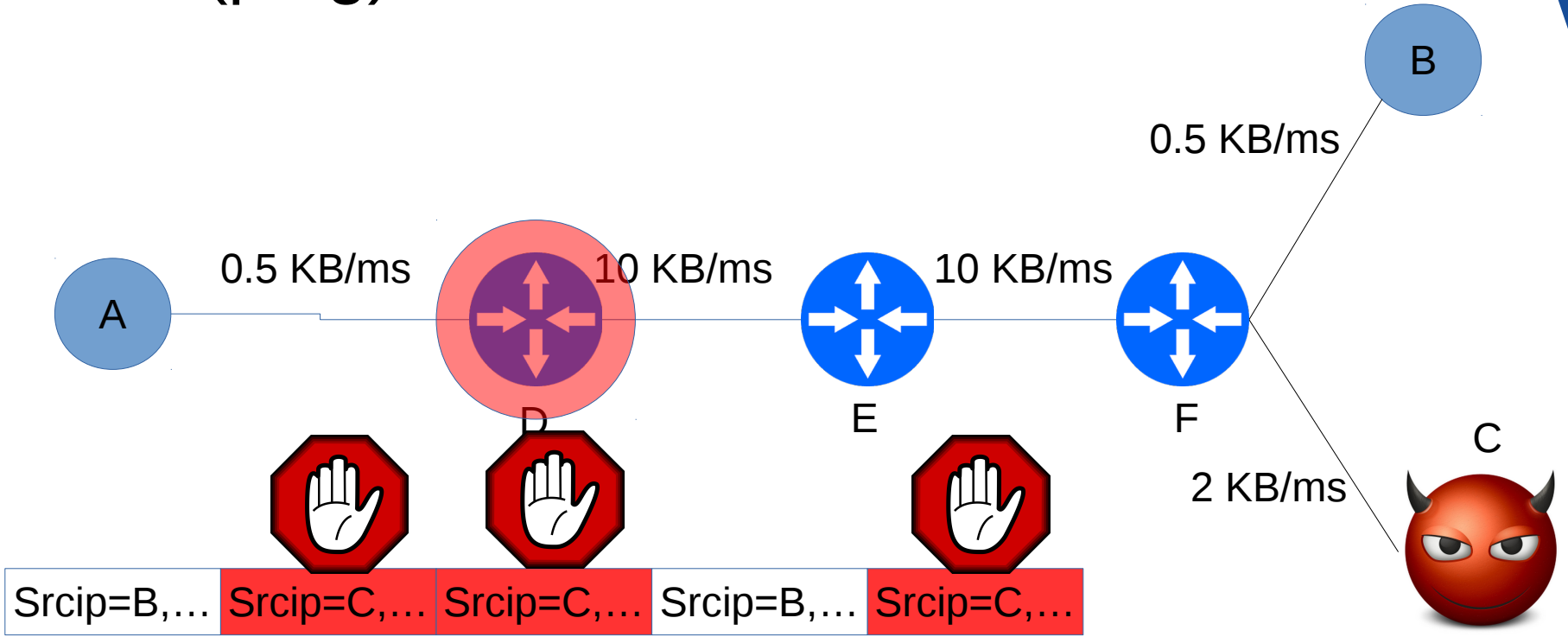


Srcip=B,…	Srcip=C,…	Srcip=C,…	Srcip=B,…	Srcip=C,…

ICMP (ping) flood

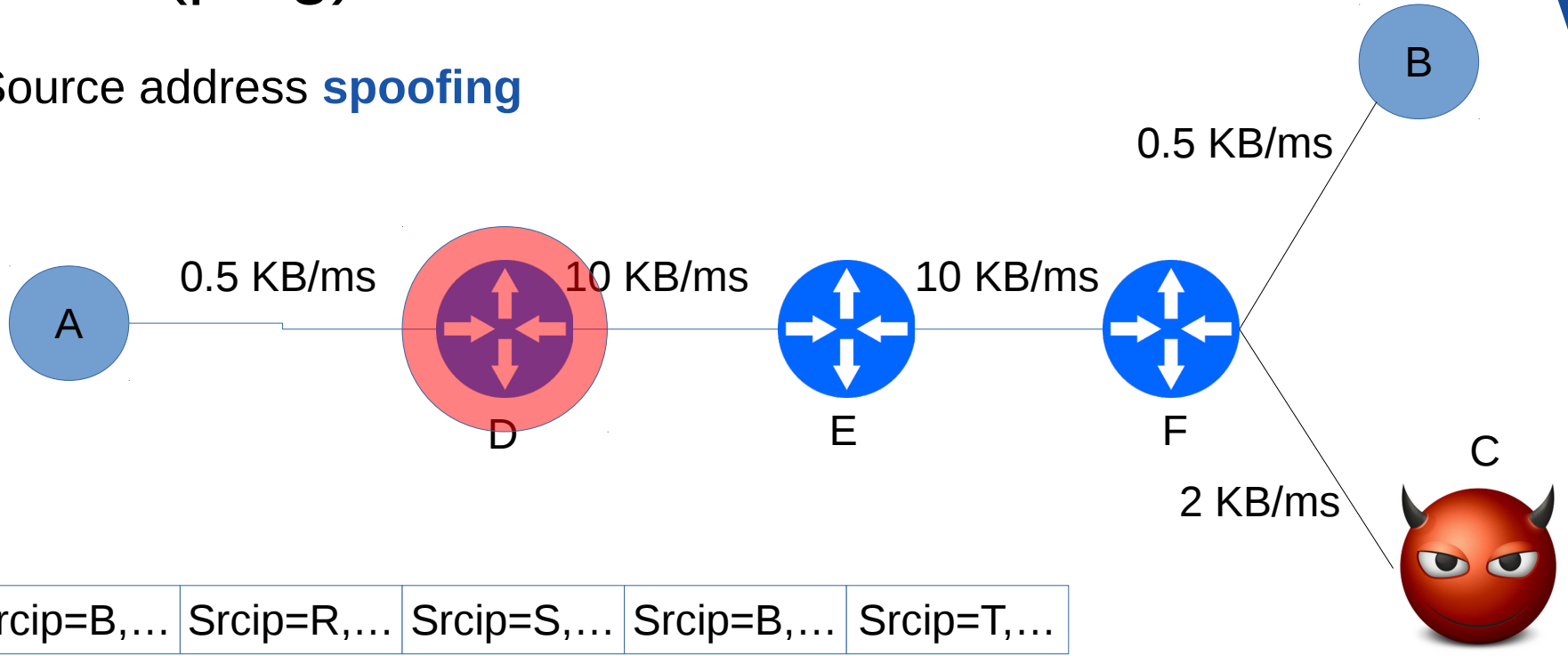


ICMP (ping) flood



ICMP (ping) flood

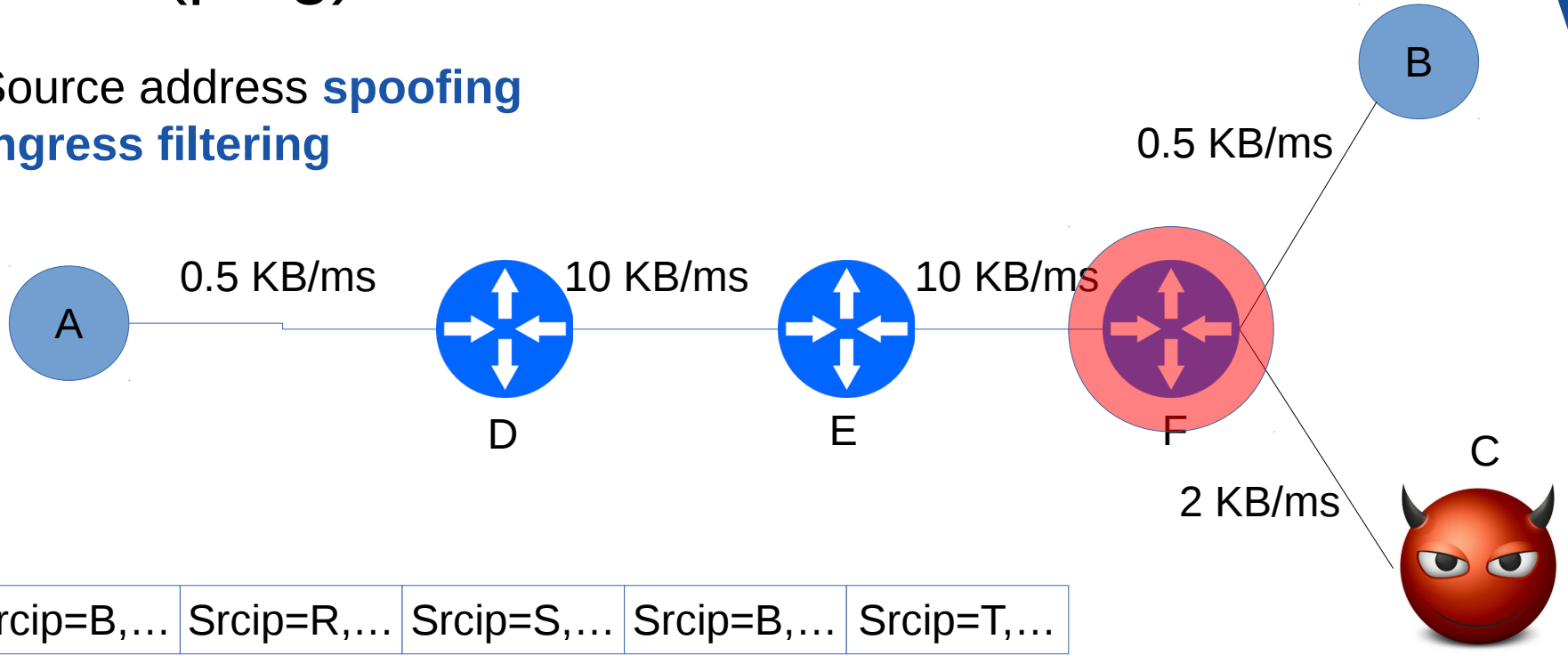
Source address **spoofing**



Srcip=B,…	Srcip=R,…	Srcip=S,…	Srcip=B,…	Srcip=T,…

ICMP (ping) flood

Source address **spoofing**
Ingress filtering



Srcip=B,…	Srcip=R,…	Srcip=S,…	Srcip=B,…	Srcip=T,…

ICMP (ping) flood

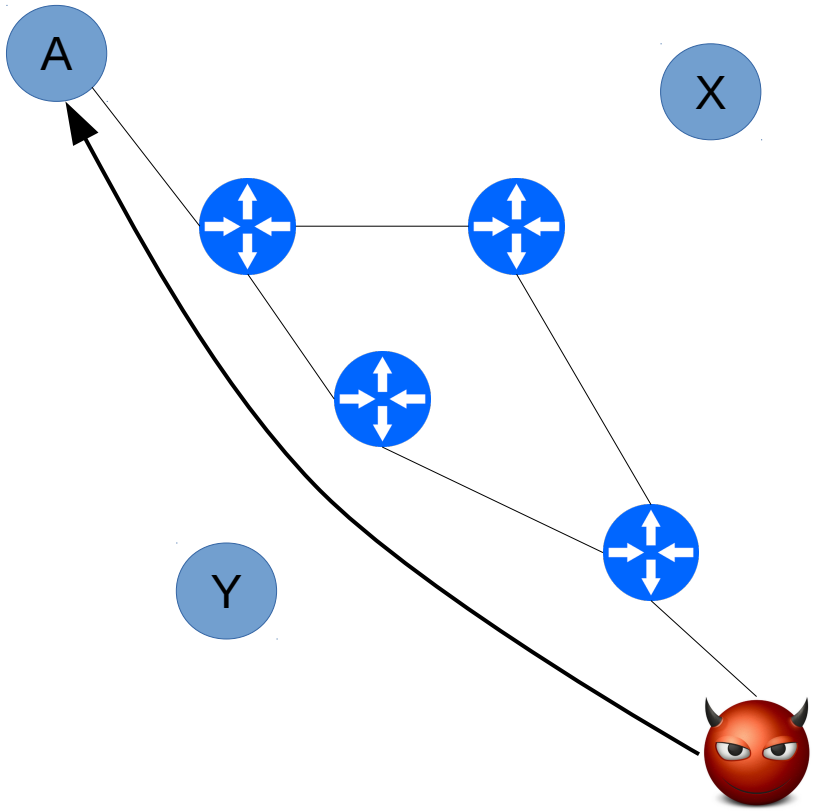
Source address **spoofing**

- Hide attacker identity

ICMP (ping) flood

Source address **spoofing**

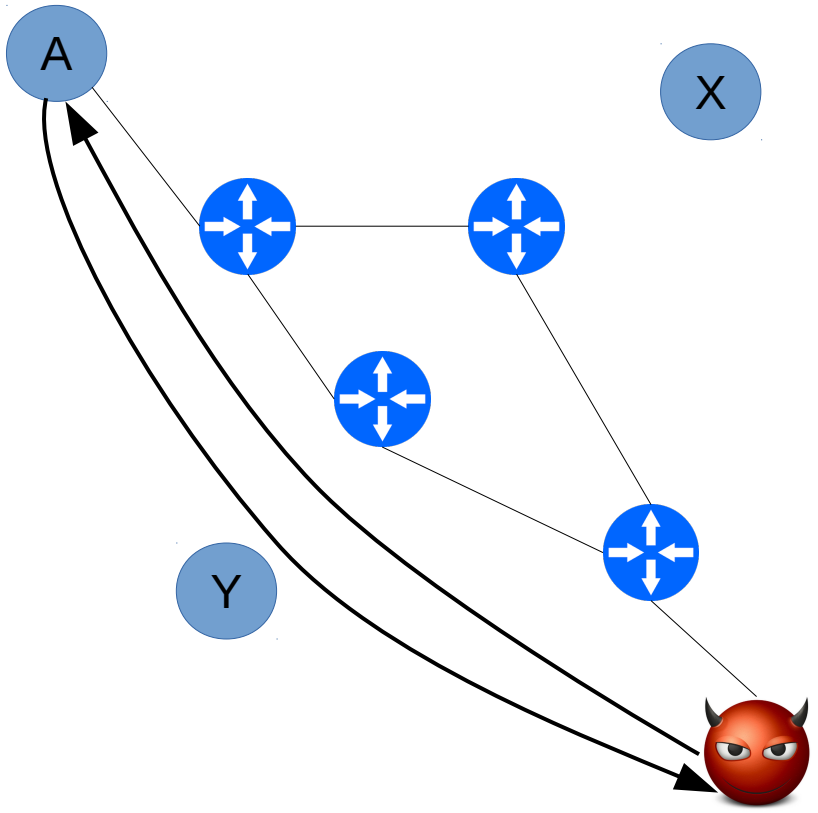
- Hide attacker identity
- Avoid back traffic



ICMP (ping) flood

Source address **spoofing**

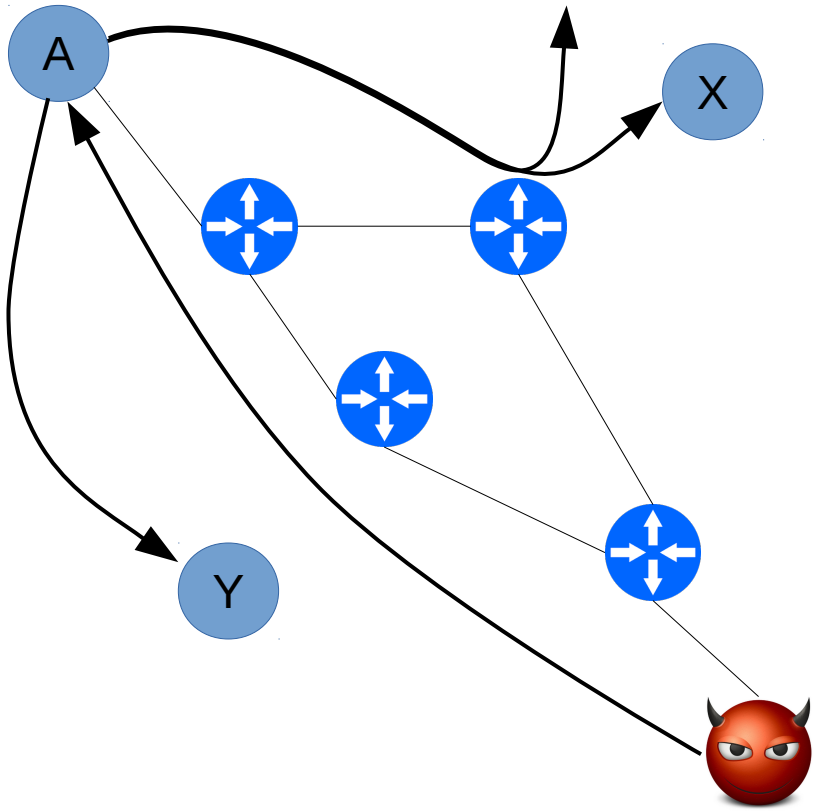
- Hide attacker identity
- Avoid back traffic



ICMP (ping) flood

Source address **spoofing**

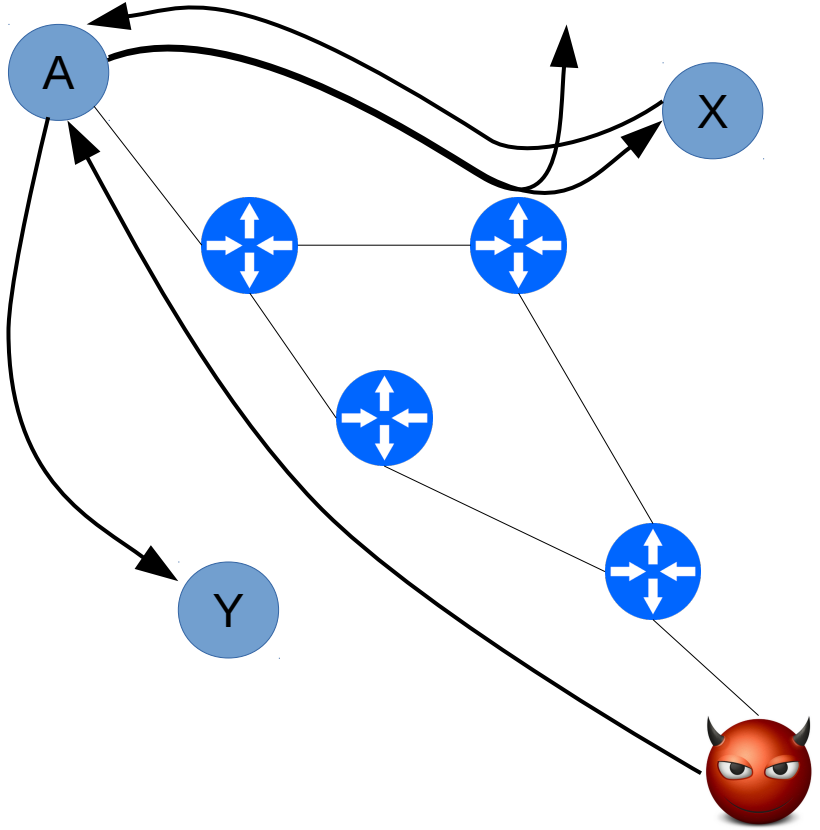
- Hide attacker identity
- Avoid back traffic



ICMP (ping) flood

Source address **spoofing**

- Hide attacker identity
- Avoid back traffic
- IMCP Answers increase the victim’s congestion

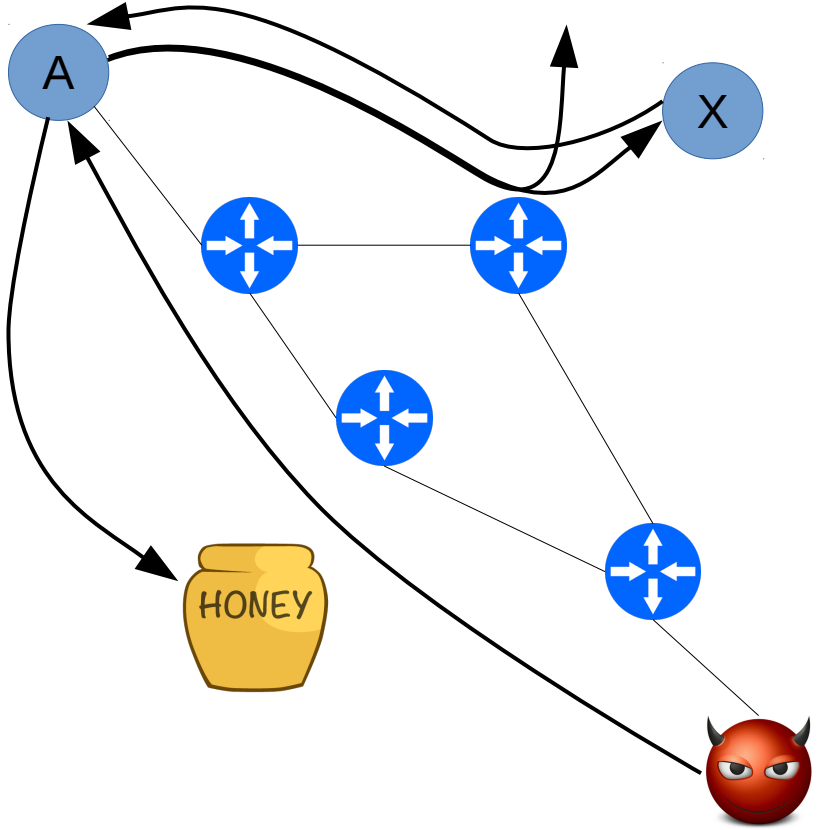


Network congestion

ICMP (ping) flood

Source address **spoofing**

- Hide attacker identity
- Avoid back traffic
- IMCP Answers increase the victim’s congestion
- IMCP Answers can be used to identify attacks



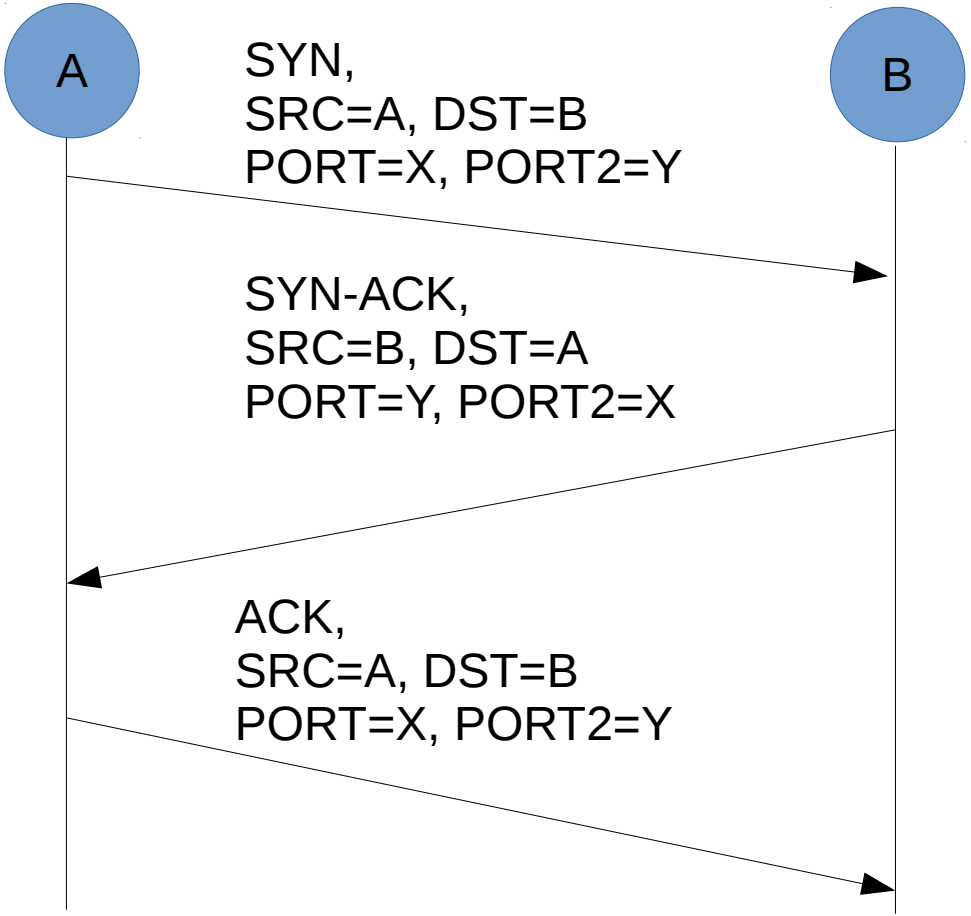
Flooding attacks

- **Require more bandwidth than the victim**
- **ICMP** flood
  - Use ICMP packets: e.g. echo requests
  - typically allowed through, some required
- **UPD** flood
  - DNS, PTP, VoIP
- **TCP SYN** flood
  - use TCP SYN (connection request) packets

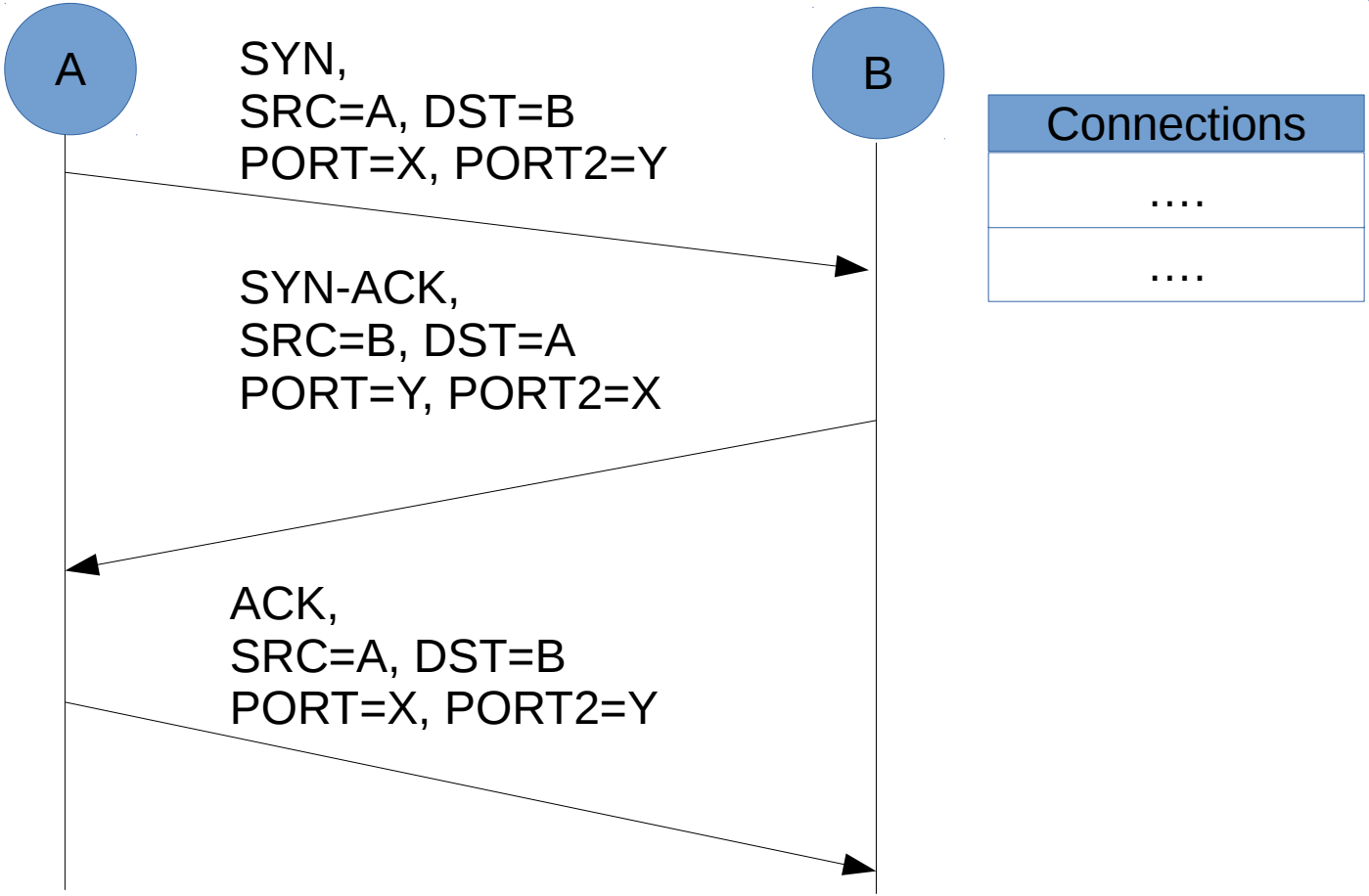
TCP SYN Spoofing

- attacks ability of a server to respond to future connection requests
- overflowing tables used to manage them
- possible even if the attacker has small bandwidth

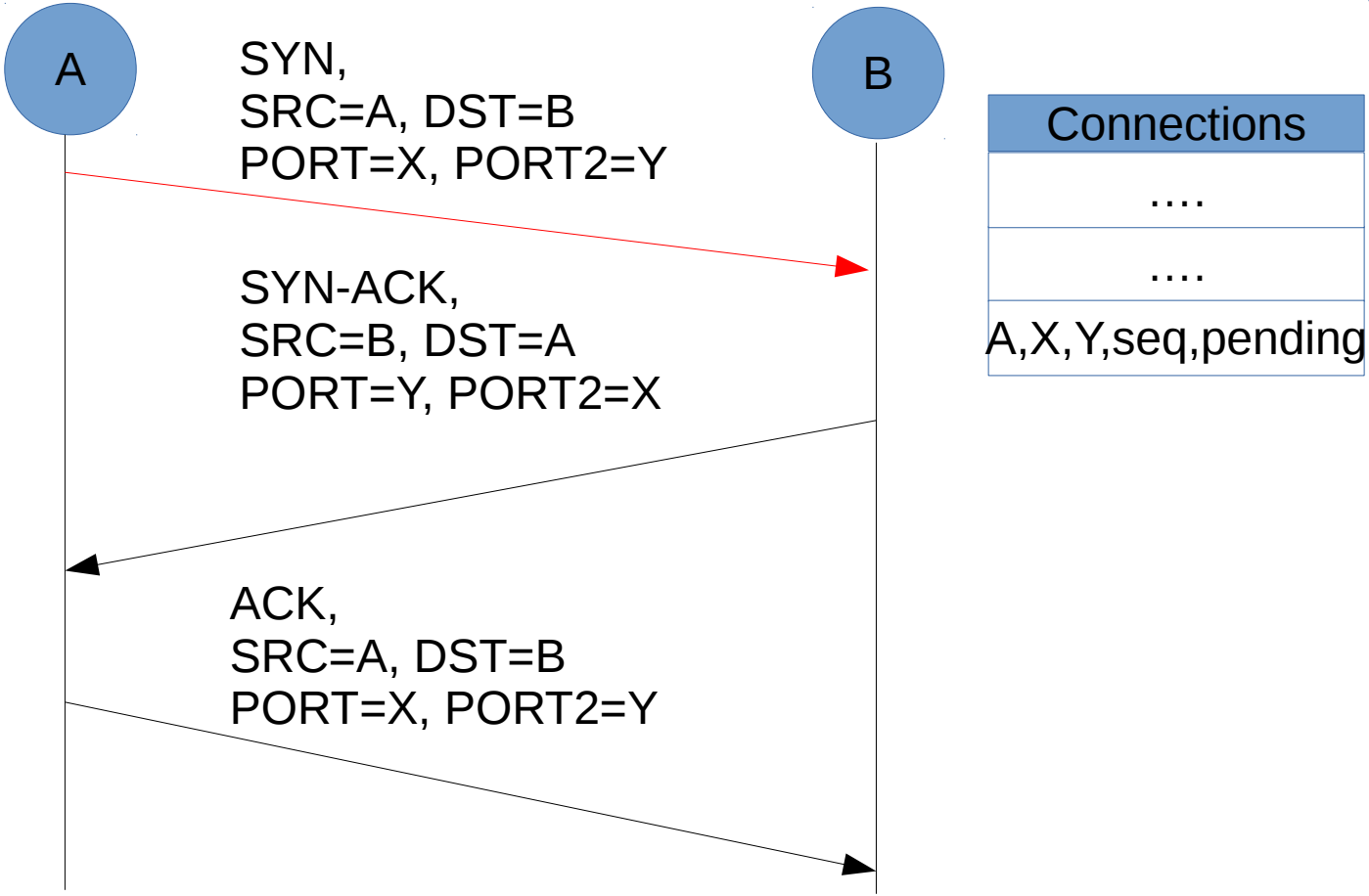
TCP SYN Spoofing



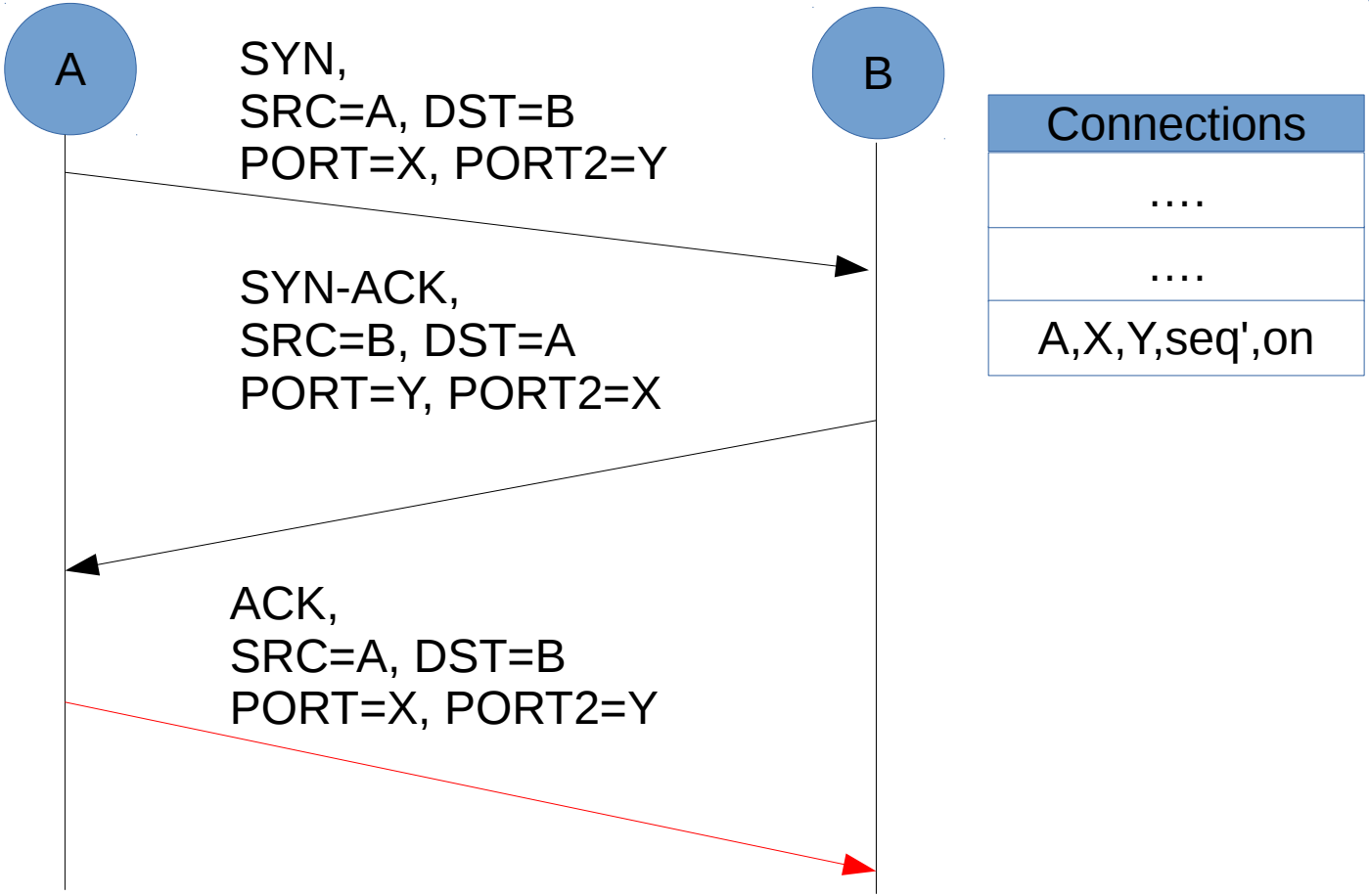
TCP SYN Spoofing



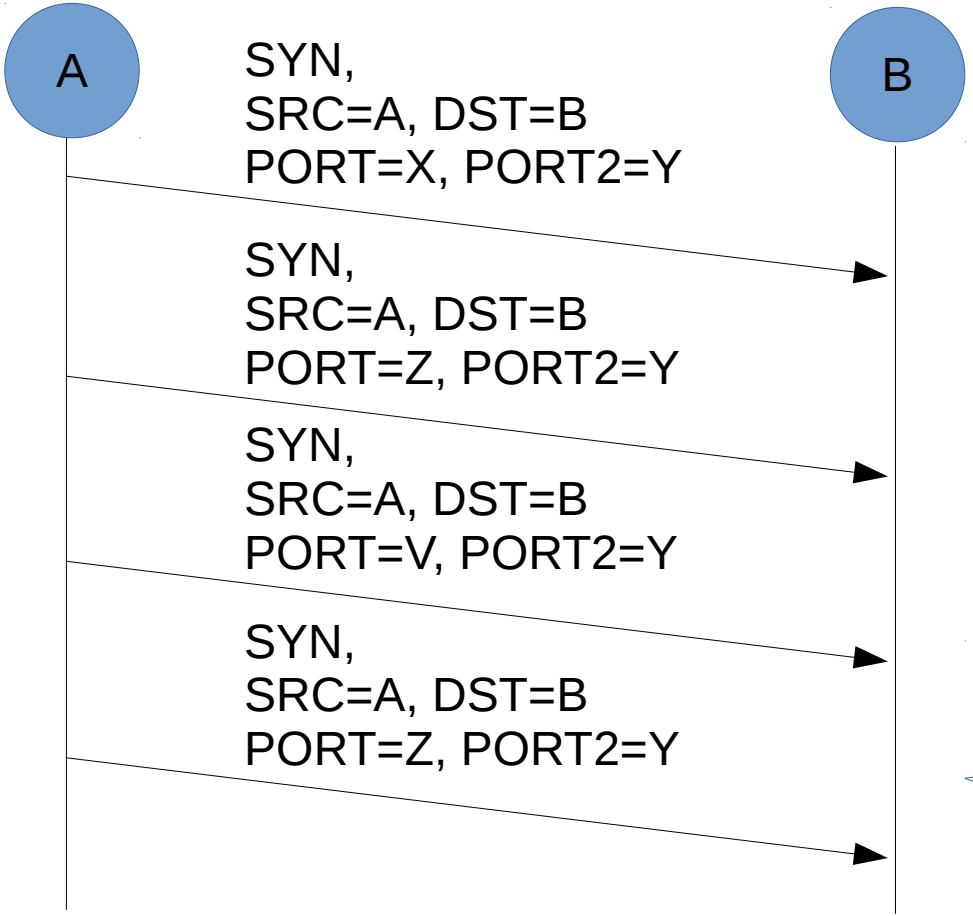
TCP SYN Spoofing



TCP SYN Spoofing



TCP SYN Spoofing



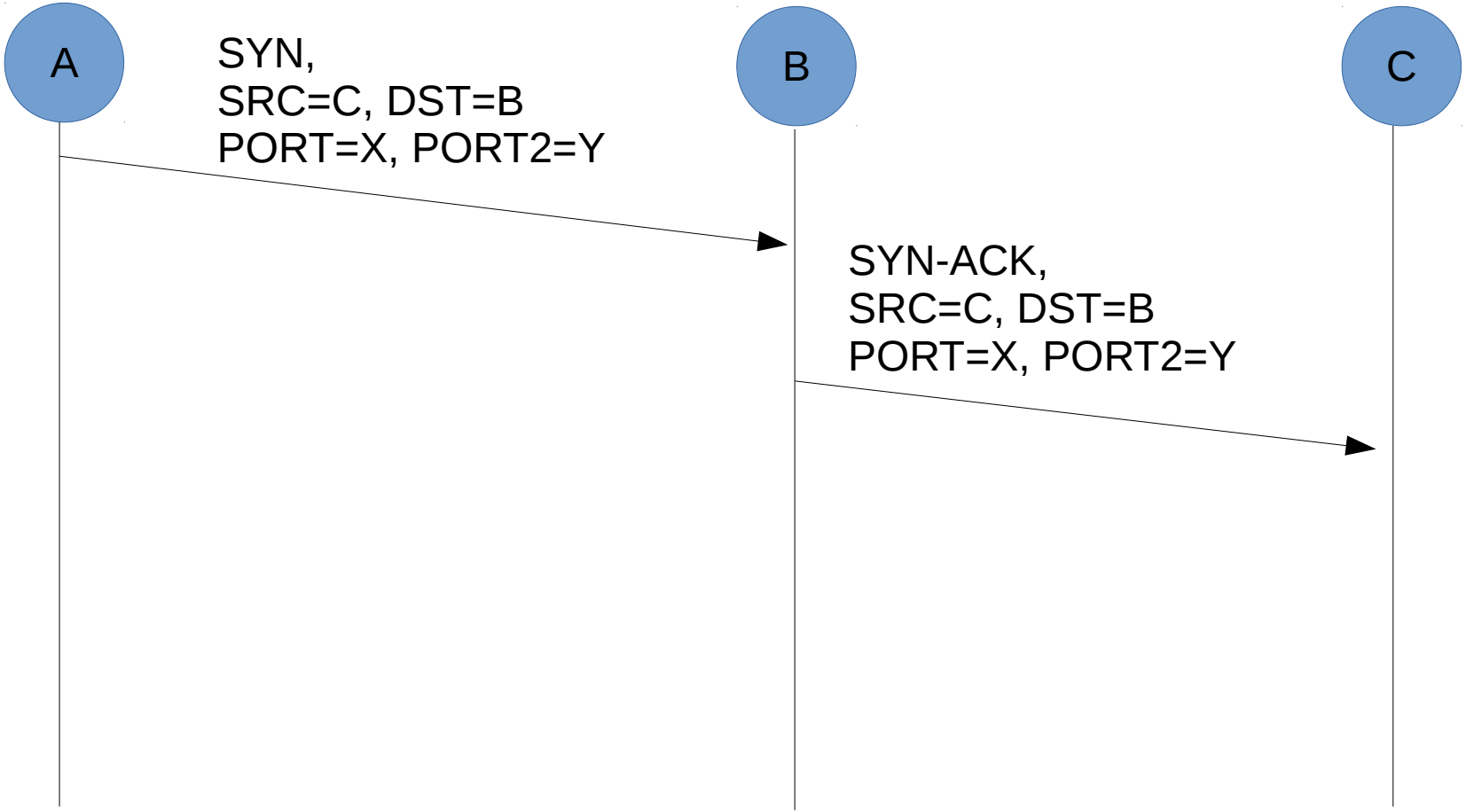
Connections
….
….
A,Y,X,…,pending
A,Y,Z,…,pending
A,Y,V,…,pending
A,Y,W,…,pending

Memory Full

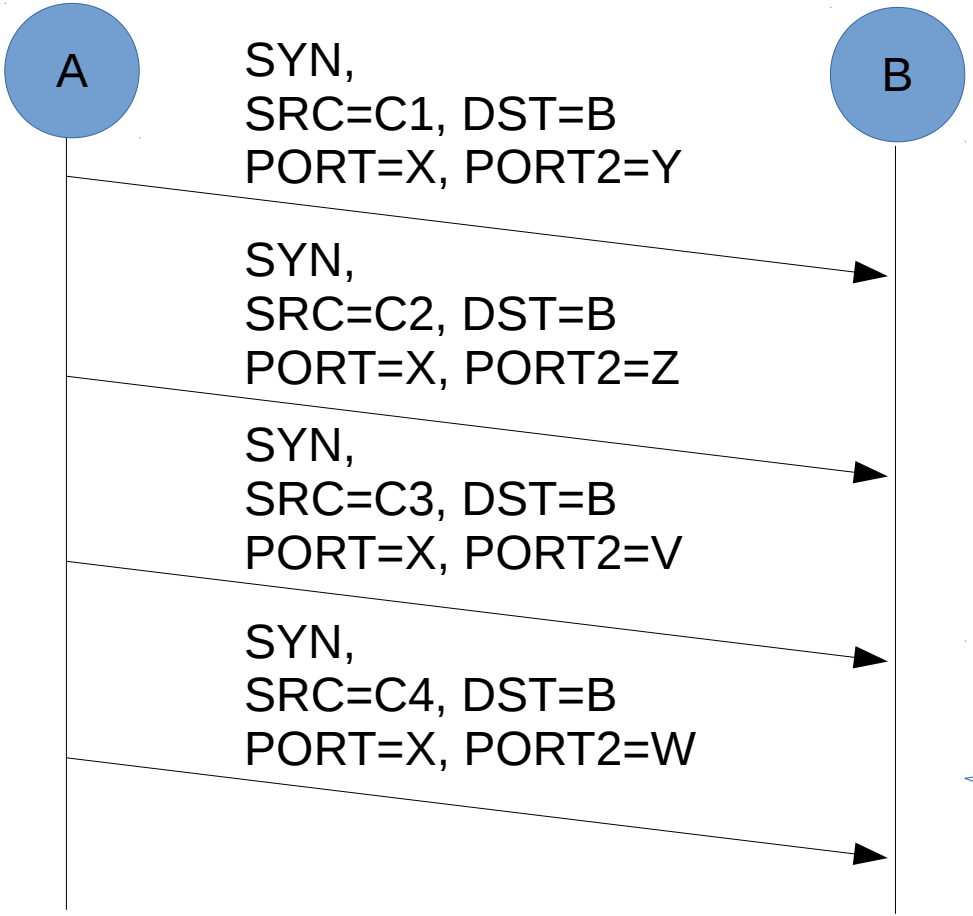
TCP SYN Spoofing

- Time-out connections
- Limit number of established and pending connections per host

TCP SYN Spoofing



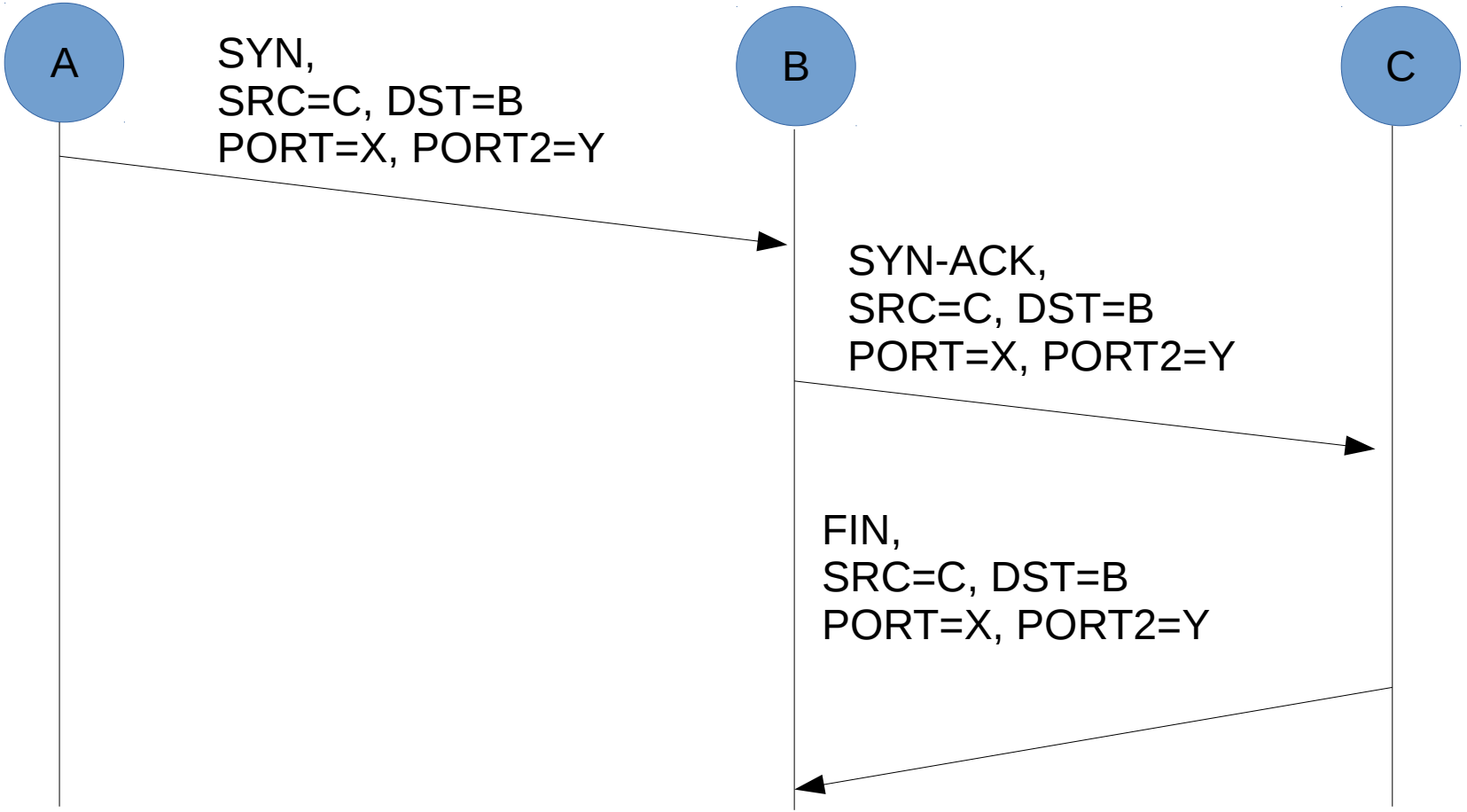
TCP SYN Spoofing



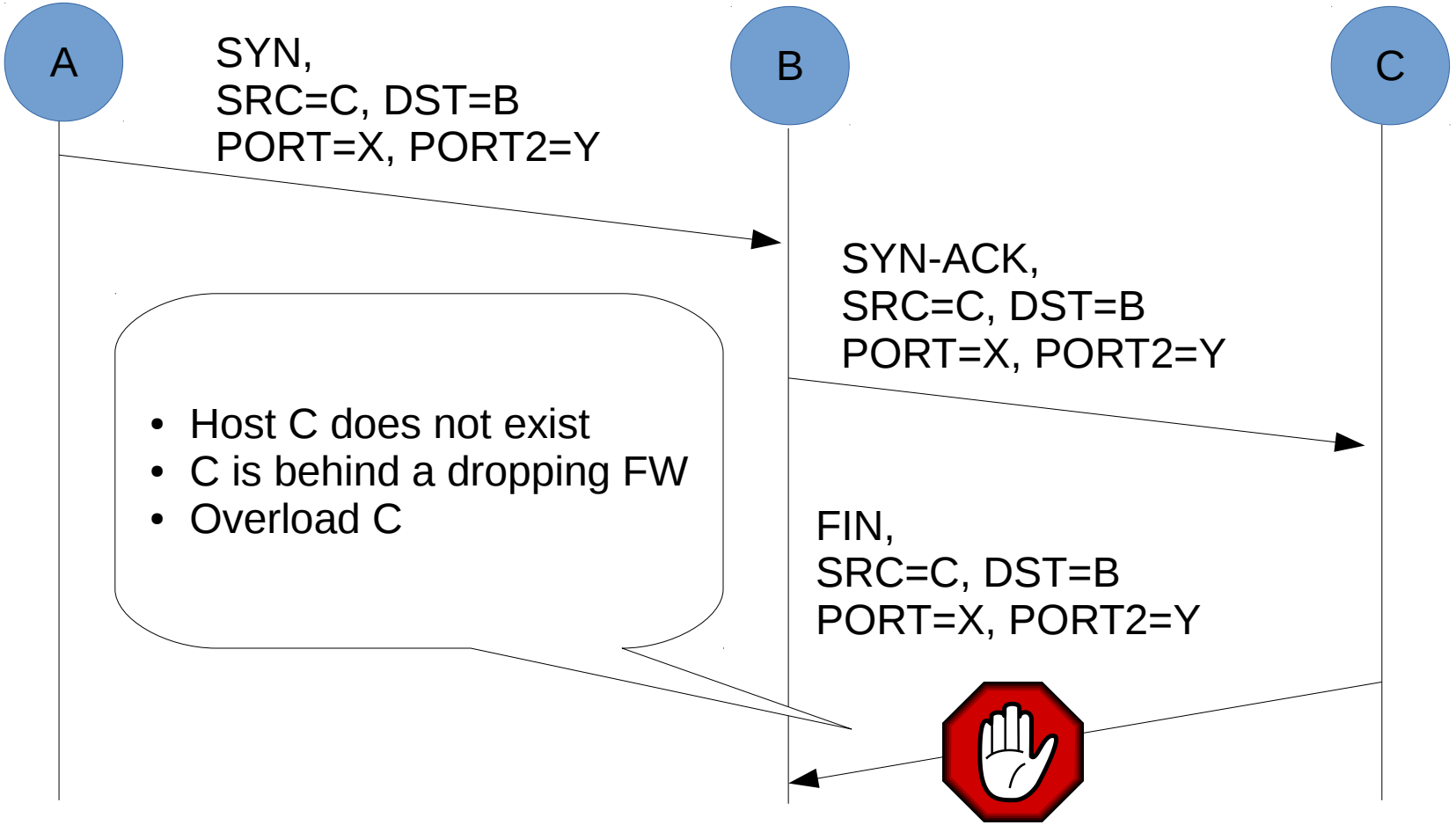
Connections
….
….
C1,X,Y,…,pending
C2,X,Z,…,pending
C3,X,V,…,pending
C4,X,W,…,pending

Memory Full

TCP SYN Spoofing



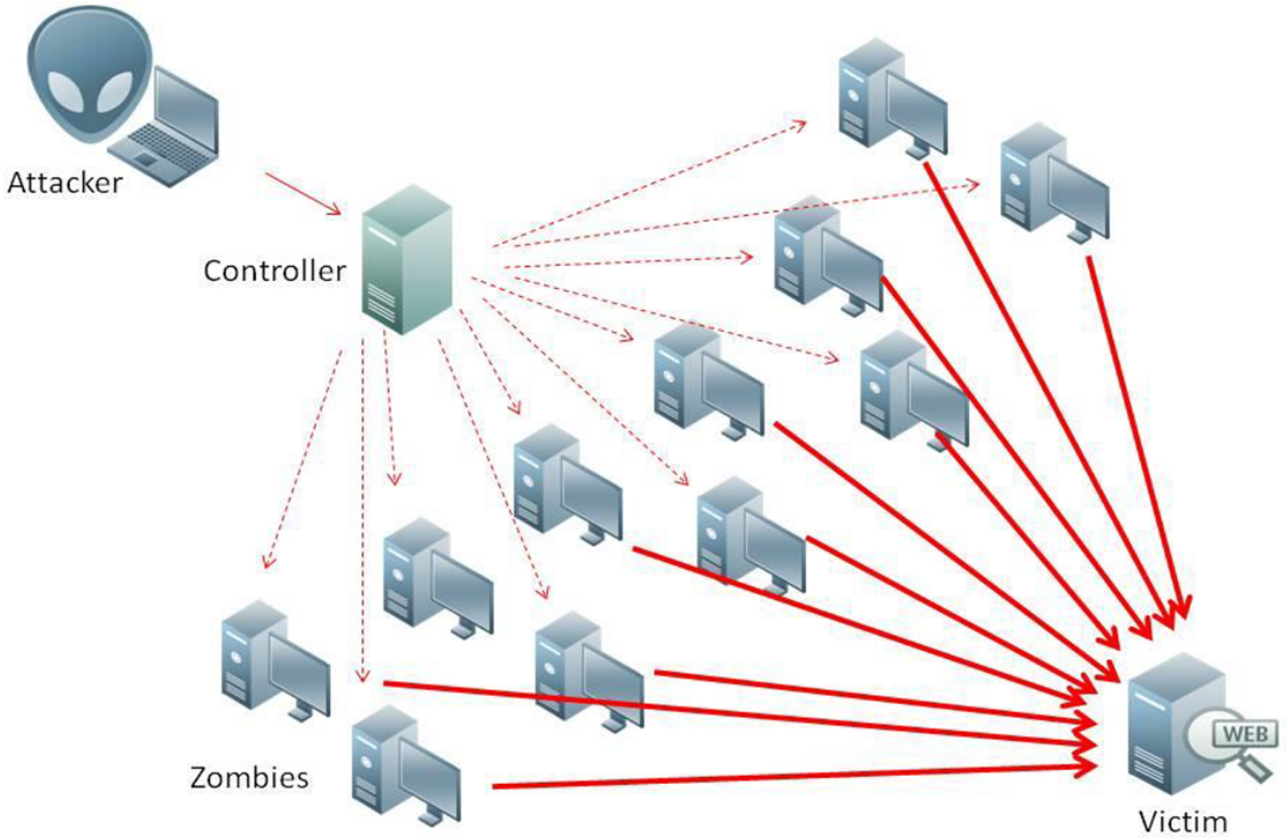
TCP SYN Spoofing



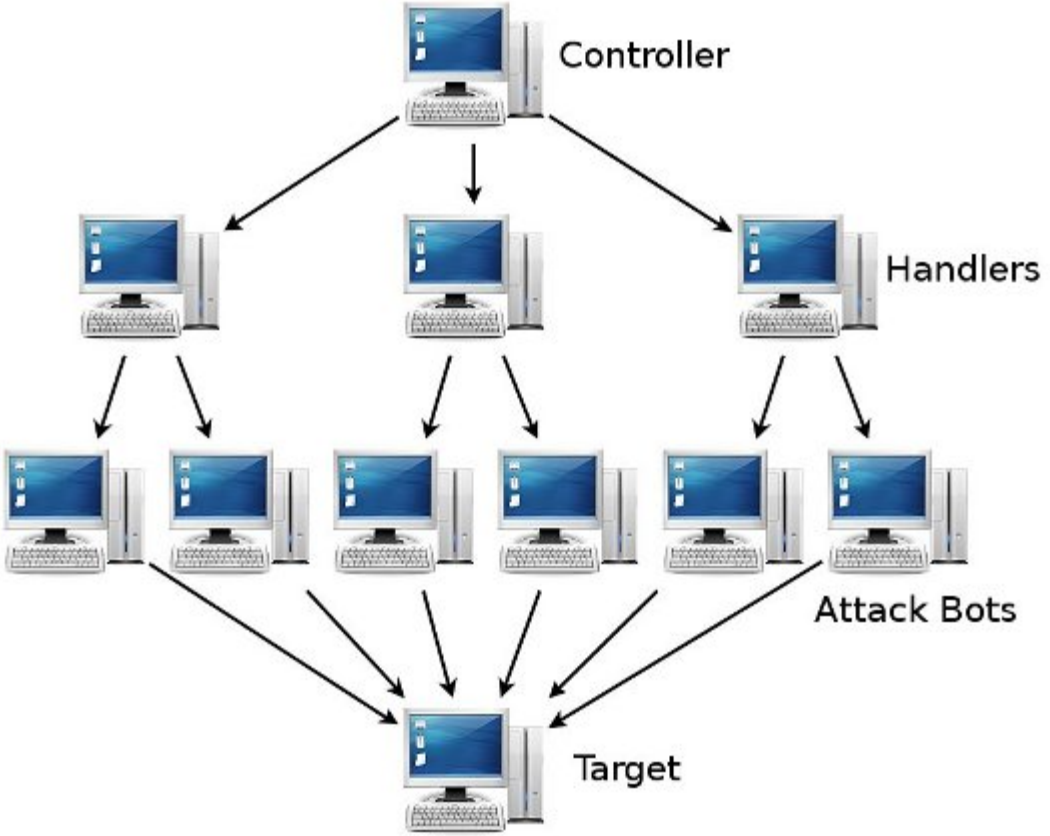
Distributed DOS

- have limited volume if single source used
- multiple systems allow much higher traffic volumes
  - Distributed Denial of Service (DDoS) Attack
- often compromised PCs / workstations
  - **Zombies/bots** with backdoor programs installed
  - forming a **botnet**

Distributed DOS



Distributed DOS

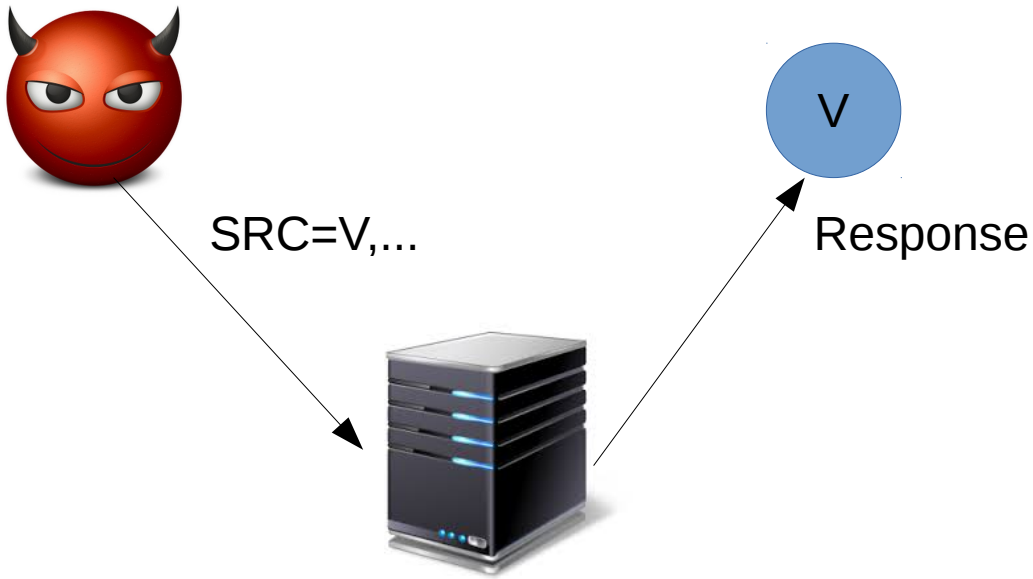


Distributed DOS

- MIRAI (21 October 2016)
- Internet-of-Things' (IoT) botnet
  - Vulnerable DVRs, surveillance cameras, …
  - 500,000 compromised IoT devices, 665 Gbps
- Multiple DOS attacks
  - SYN-flooding, UDP flooding, …
- Target Internet services firm Dyn (DNS servers)
  - Affected Twitter, Reddit, Spotify, SoundCloud, WhatsApp

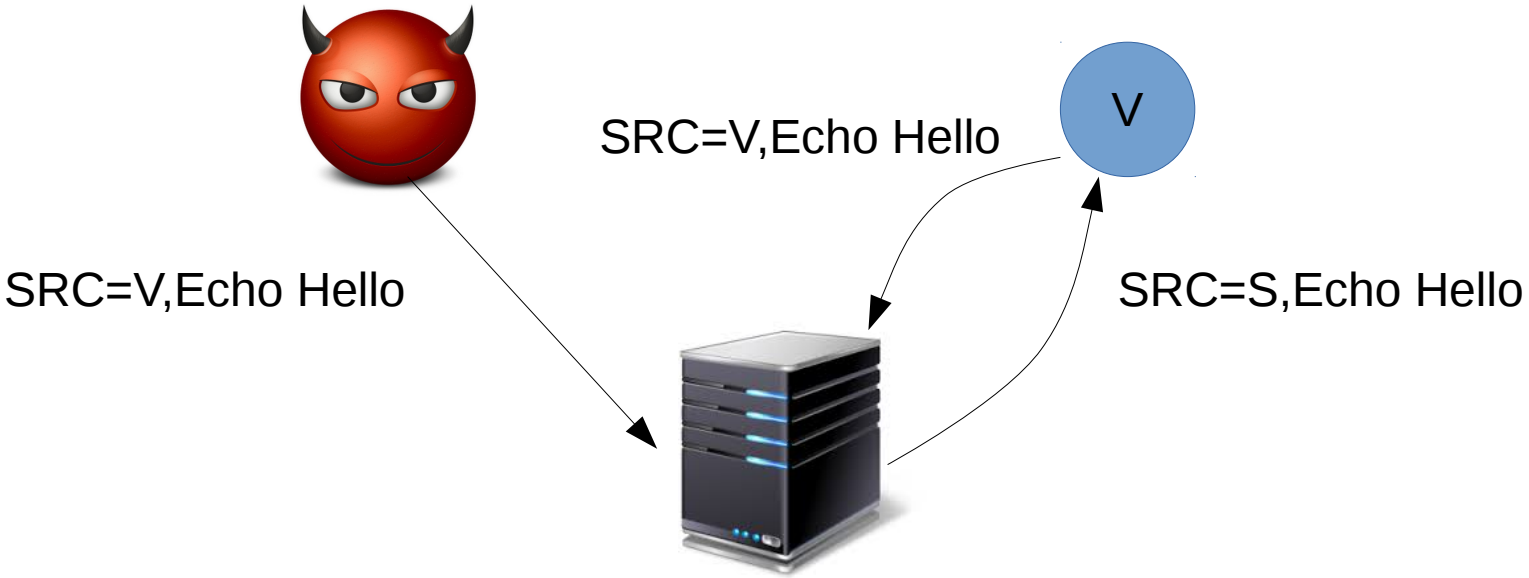
Reflection Attacks

- various protocols e.g. ICMP, UDP or TCP/SYN
- with spoofed source address



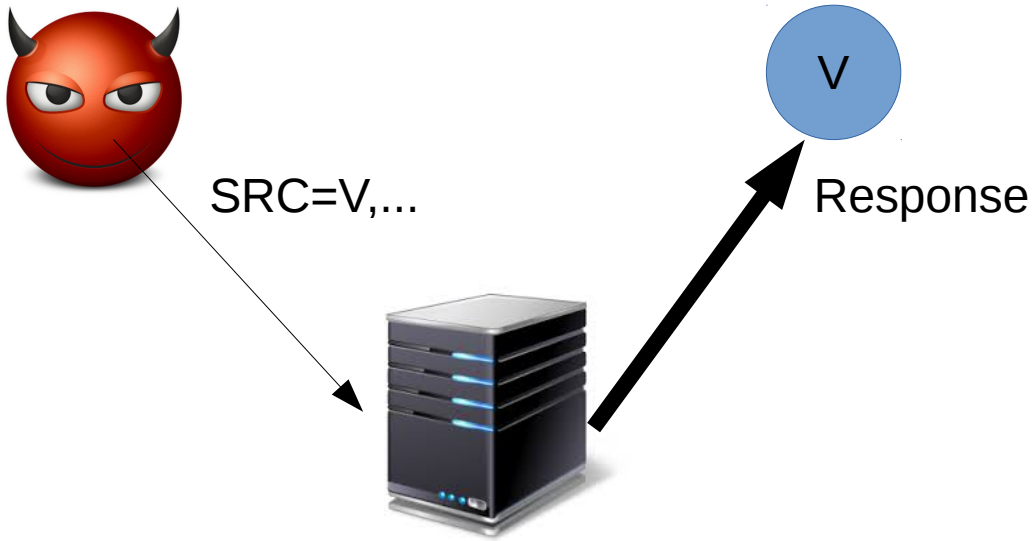
Reflection Attacks

- various protocols e.g. ICMP, UDP or TCP/SYN
- **loops**



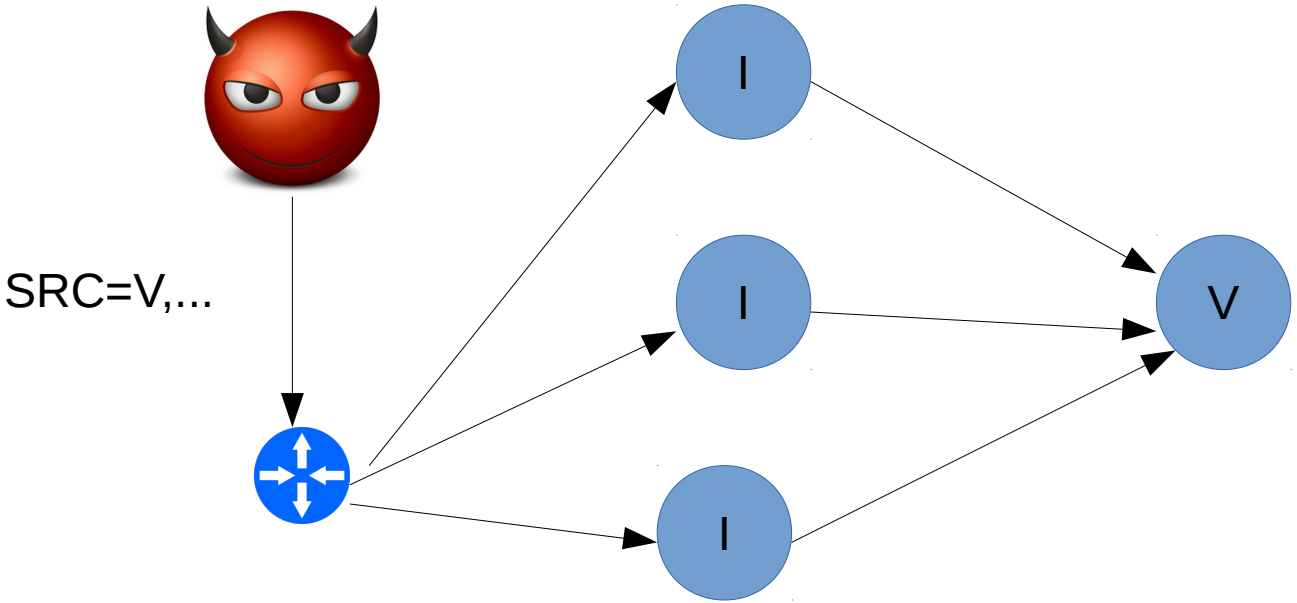
Amplification Attacks

- DNS (60 byte request to 512 - 4000 byte response)



Amplification Attacks

- DNS (60 byte request to 512 - 4000 byte response)
- Broadcast/Multicast



Denial Of Service

- Networked
- **Based on SW vulnerabilities**
- Physical

DoS on **Phones**

- See SMS-o-Death by Collin Mulliner
- http://www.mulliner.org/security/sms

DoS on **e-mail**

Set vacation message

A

Set vacation message

B

DoS on **e-mail**

From: bob@B
To: alice@A
Subj: Hello
...

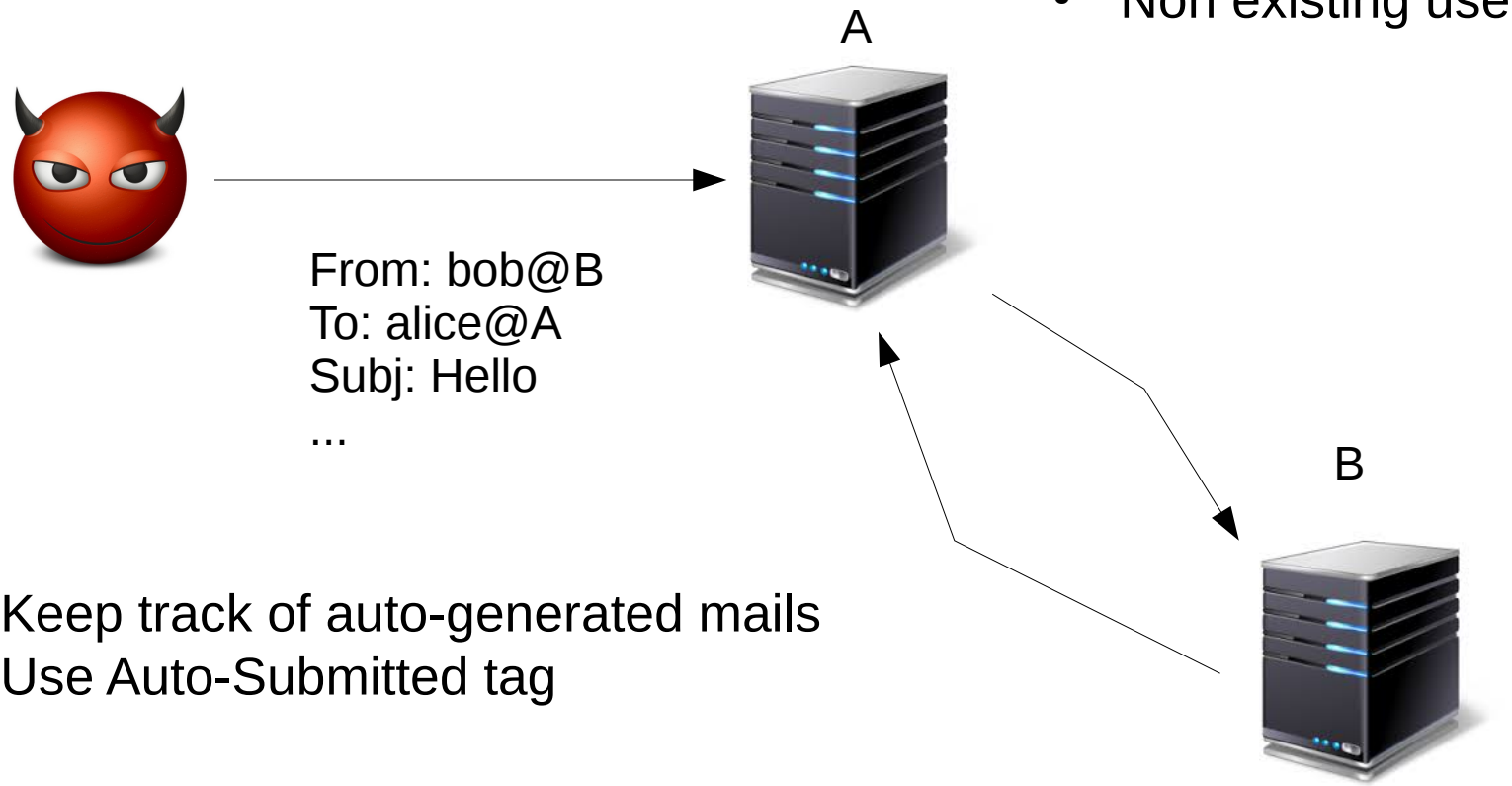
To: alice@A
From: bob@B
Subj: Re-Hello
I'm in Thailand

B

From: bob@B
To: alice@A
Subj: Re-Hello
I'm in Mexico

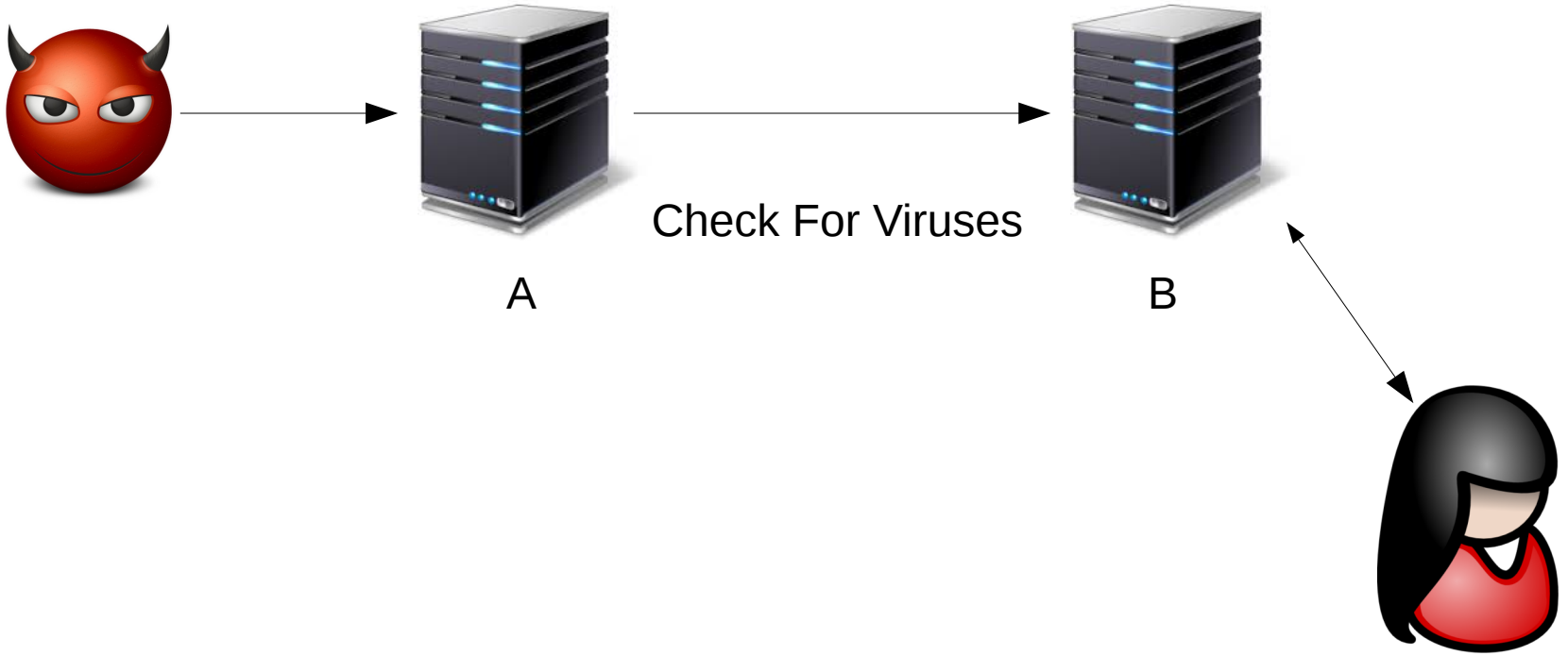
DoS on **e-mail**

- User not allowed e.g. SMTP fax proxy
- Non existing user

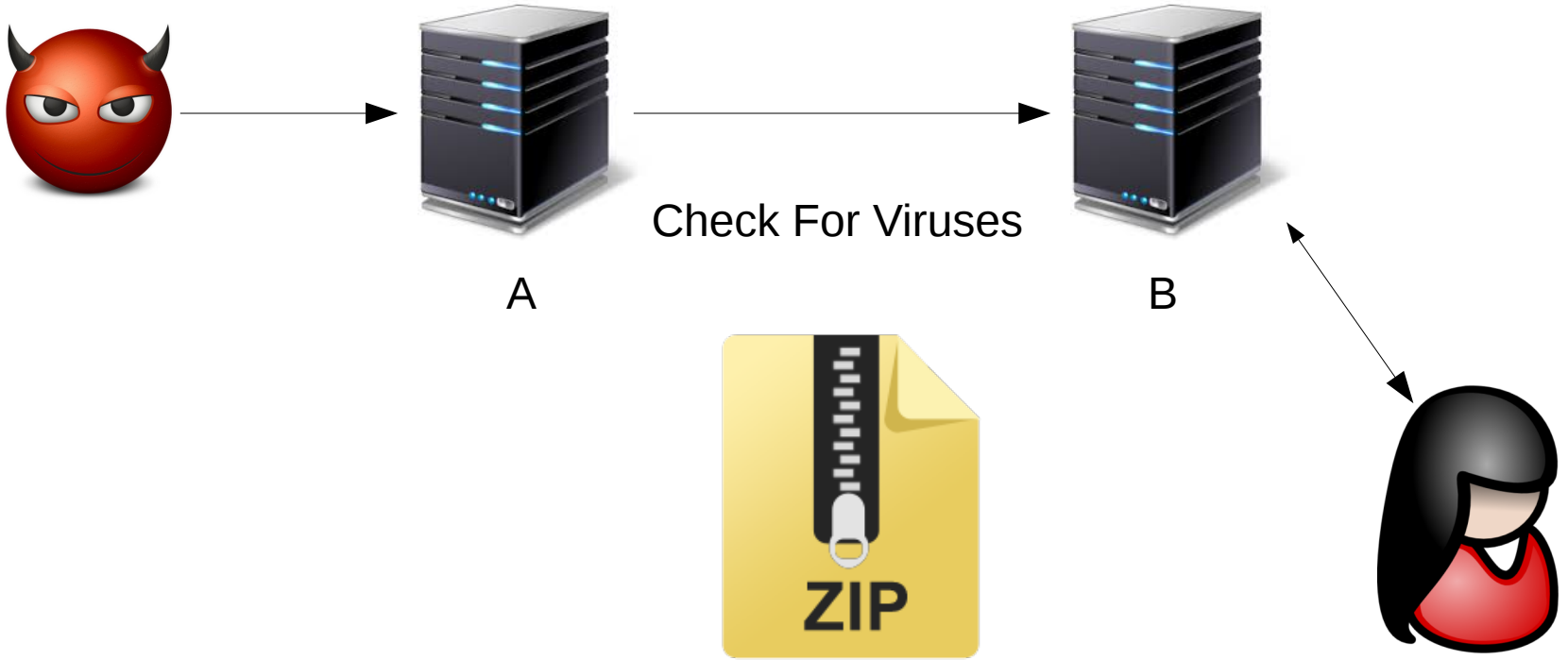


- Keep track of auto-generated mails
- Use Auto-Submitted tag

DoS on **e-mail filters**



DoS on **e-mail filters**

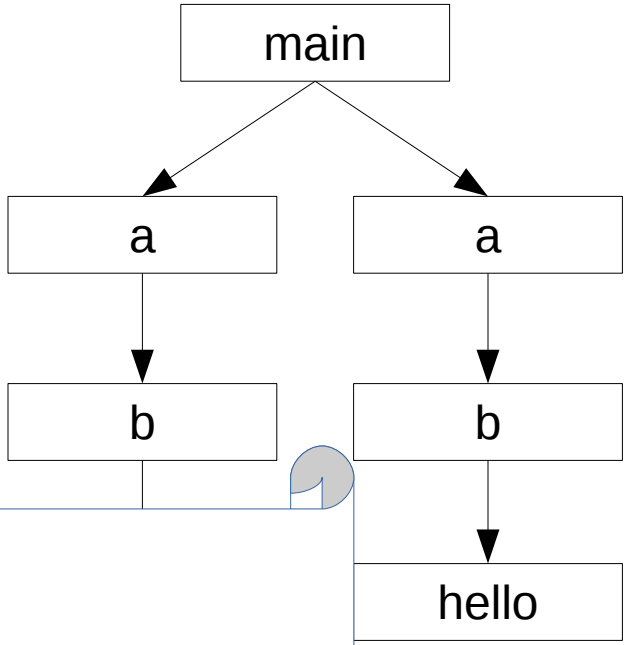


DoS on **e-mail filters**

- Infinite recursive zip
- Bombs (e.g. non recursive files)
  - gzip: 100 GB, 97 MB compressed, 1000:1 ration
  - bzip2: 100 GB, 69 KB, 1.6 ∗ 10 6 :1
  - PNG image: 19000 x 19000, 1-bit (45 MB) expand in 24-bit color to 1 GB, 44 KB compressed, 1000:1

XML bomb

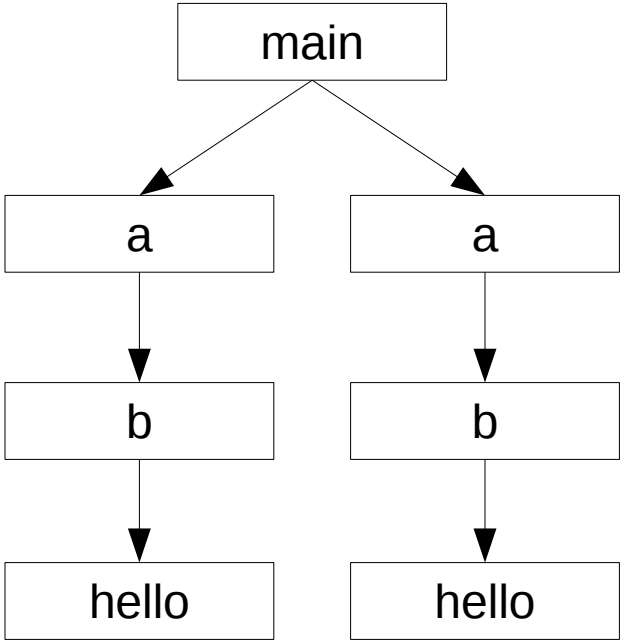
```
<?xml version="1.0"?>
<main>
 <a><b>hello</b></a>
 <a><b>hello</b></a>
</main>
```



Question 4: You knowledge about XML
A) I do not know what XML is
B) I know what XML is
C) I know how a XML DOM parser works
D) I know how a XML DOM parser works and what a DTD is

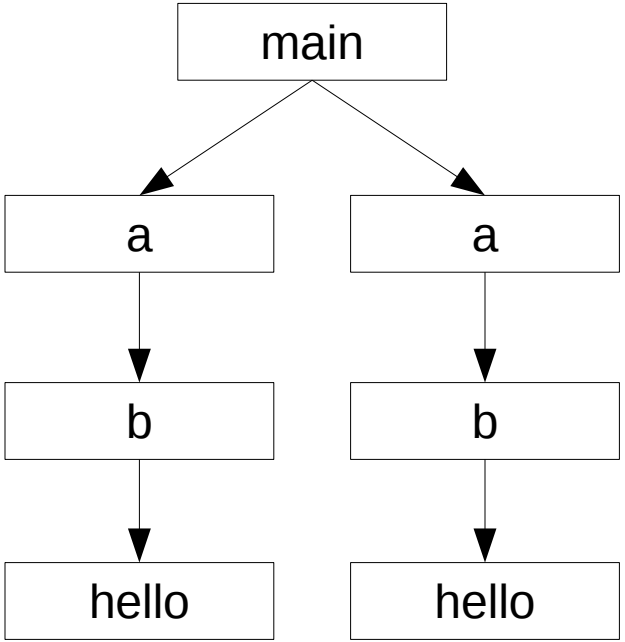
XML bomb

```
<?xml version="1.0"?>
<main>
 <a><b>hello</b></a>
 <a><b>hello</b></a>
</main>
```



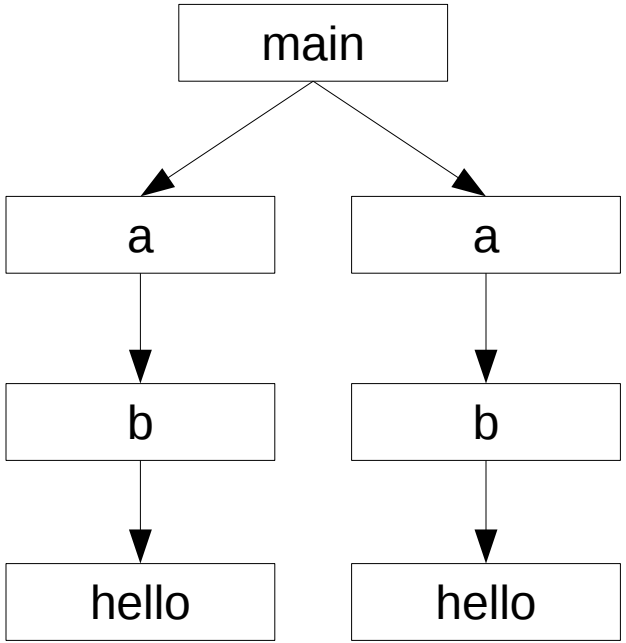
XML bomb

```
<?xml version="1.0"?>
<!DOCTYPE main [
  <!ELEMENT main (a+)>
  <!ELEMENT a (b+)>
  <!ELEMENT b (#PCDATA)>
]>
<main>
 <a><b>hello</b></a>
 <a><b>hello</b></a>
</main>
```



XML bomb

```
<?xml version="1.0"?>
<!DOCTYPE main [
  <!ELEMENT main (a+)>
  <!ELEMENT a (b+)>
  <!ELEMENT b (#PCDATA)>
  <!ENTITY H "hello">
]>
<main>
 <a><b>&H;</b></a>
 <a><b>&H;</b></a>
</main>
```



```
<?xml version="1.0"?>
<!DOCTYPE lolz [
<!ENTITY lol "lol">
<!ELEMENT lolz (#PCDATA)>
<!ENTITY lol1 "&lol;&lol;&lol;&lol;&lol;&lol;&lol;&lol;&lol;&lol;">
<!ENTITY lol2 "&lol1;&lol1;&lol1;&lol1;&lol1;&lol1;&lol1;&lol1;&lol1;&lol1;
<!ENTITY lol3 "&lol2;&lol2;&lol2;&lol2;&lol2;&lol2;&lol2;&lol2;&lol2;&lol2;
<!ENTITY lol4 "&lol3;&lol3;&lol3;&lol3;&lol3;&lol3;&lol3;&lol3;&lol3;&lol3;
<!ENTITY lol5 "&lol4;&lol4;&lol4;&lol4;&lol4;&lol4;&lol4;&lol4;&lol4;&lol4;
<!ENTITY lol6 "&lol5;&lol5;&lol5;&lol5;&lol5;&lol5;&lol5;&lol5;&lol5;&lol5;
<!ENTITY lol7 "&lol6;&lol6;&lol6;&lol6;&lol6;&lol6;&lol6;&lol6;&lol6;&lol6;
<!ENTITY lol8 "&lol7;&lol7;&lol7;&lol7;&lol7;&lol7;&lol7;&lol7;&lol7;&lol7;
<!ENTITY lol9 "&lol8;&lol8;&lol8;&lol8;&lol8;&lol8;&lol8;&lol8;&lol8;&lol8;
]>
<lolz>&lol9;</lolz>
```

1 KB file
3GB XML (DOM worst)

Fork bomb

- PostgreSQL 7.2
- Uses elapsed milliseconds to compute number of thread to spawn
- No limit check
- What happen if you advance the date to next year?
- Network Time Protocol attack

WEB DOS

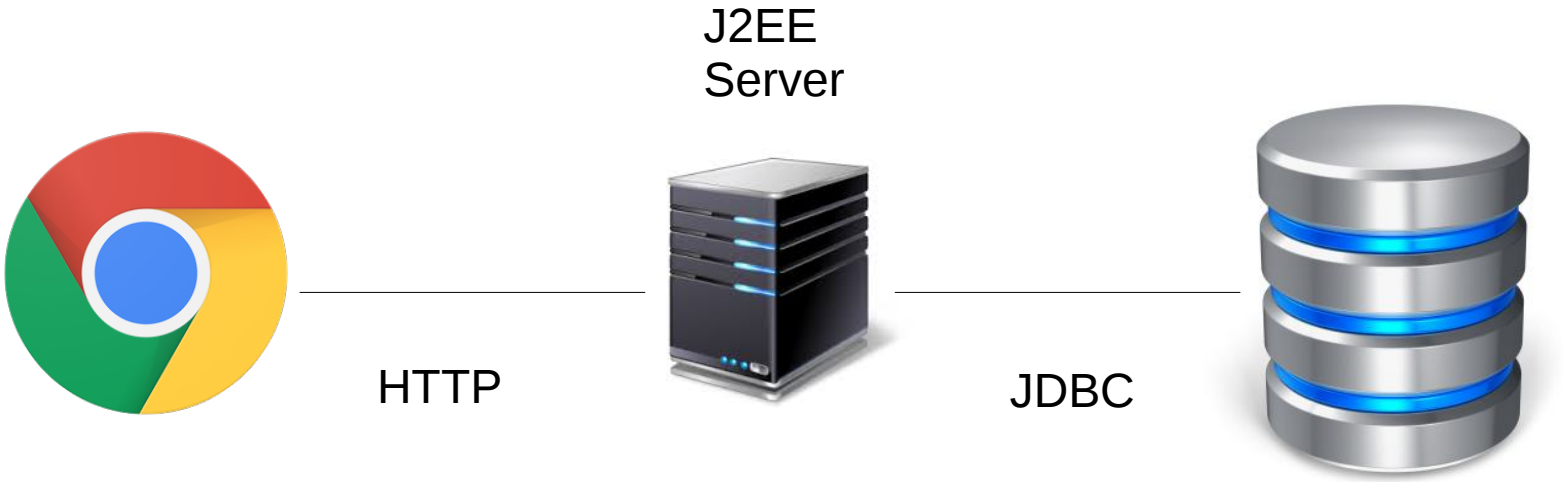
- Involuntary
  - Slashdotted
  - Distributed flood (spiders)
- Voluntary
  - Target supporting services e.g. DNS

WEB DOS

- Slowloris
  - HTTP server
    - one thread/one connection
    - one connection/one request

  1) Open TCP connection
  2) Infinitely (and slowly) send HTTP headers
  3) Consume the thread pool and connections

WEB DOS



J2EE/JDBC

- To communicate with DB you need a transaction
- 1 DB connection can handle 1 transaction at a time
- Work in a transaction can be committed or undone
- Transactions are isolated
- Common pattern
  - One transaction per HTTP request

WEB DOS

```
public class ConnectionFilter implements Filter {

  public void doFilter(ServletRequest request,
                       ServletResponse response,
                       FilterChain chain) {

    Connection connection;
    connection = connectionPool.getConnection();

    //tie it to a request attribute so other servlets can grab it
    request.setAttribute(CONNECTION_ATTR, connection);

    //let other servlets do their work
    chain.doFilter(request, response);

    connection.close();
  }
}
```

WEB DOS

```
public class ConnectionFilter implements Filter {

  public void doFilter(ServletRequest request,
                       ServletResponse response,
                       FilterChain chain) {

    Connection connection;
    connection = connectionPool.getConnection();

    //tie it to a request attribute so other servlets can grab it
    request.setAttribute(CONNECTION_ATTR, connection);

    //let other servlets do their work
    chain.doFilter(request, response);

    connection.close();
  }
}
```

WEB DOS

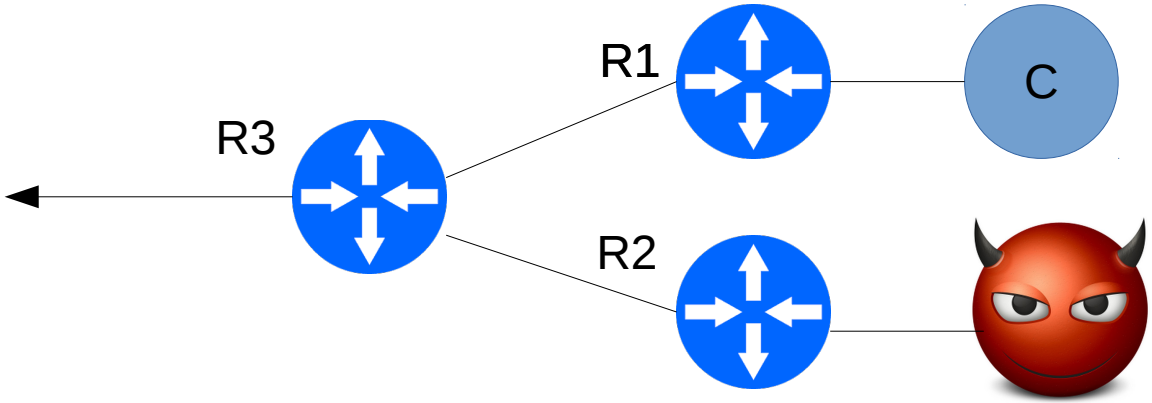
```
Connection connection;
connection = connectionPool.getConnection();

Try {
//tie it to a request attribute so other servlets can grab it
request.setAttribute(CONNECTION_ATTR, connection);

//let other servlets do their work
chain.doFilter(request, response);
}
catch (Exception e) {
  connection.rollback();
  throw e;
//mandatory: insure the connection is closed
} finally {
  if (connection != null) {
    connection.close();
  }
```

DOS **Defenses**

- high traffic volumes may be legitimate
  - result of high publicity, e.g. slash-dotted
  - or to a very popular site, e.g. Olympics etc
- or legitimate traffic created by an attacker
- three lines of defense against (D)DoS:
  - attack prevention and preemption
  - attack detection and filtering
  - attack source traceback and identification

DOS **Prevention**

- block spoofed source addresses
  - on routers as close to source as possible (ingress filtering)
  - still far too rarely implemented
  - alternatively enforce “unicast severse path”



DOS **Prevention**

- Flood: rate controls in upstream
  - on specific packets types
  - e.g. some ICMP, some UDP, TCP/SYN
- TCP/syn-spoof: use modified TCP connection handling
  - use SYN cookies when table full (consumes computational resources)
  - or selective or random drop when table full

DOS **Prevention**

- block external request to broadcasts
- block suspicious services (e.g. echo)
- block suspicious combinations (DNS to echo port)
- manage application attacks with puzzles to distinguish legitimate human requests
- good general system security practices
- use mirrored and replicated servers when high-performance and reliability required

DOS **Response**

- need good incident response plan
  - with contacts for ISP
  - needed to impose traffic filtering upstream
  - details of response process
- have standard filters
- ideally have network monitors and IDS
  - to detect and notify abnormal traffic patterns

DOS **Response**

- identify type of attack
  - capture and analyze packets
  - design filters to block attack traffic upstream
  - or identify and correct system/application bug
- have ISP trace packet flow back to source
  - may be difficult and time consuming
  - necessary if legal action desired

THANKS!

Any questions?

You can find me at robertog@kth.se