TCP/IP 协议的安全问题

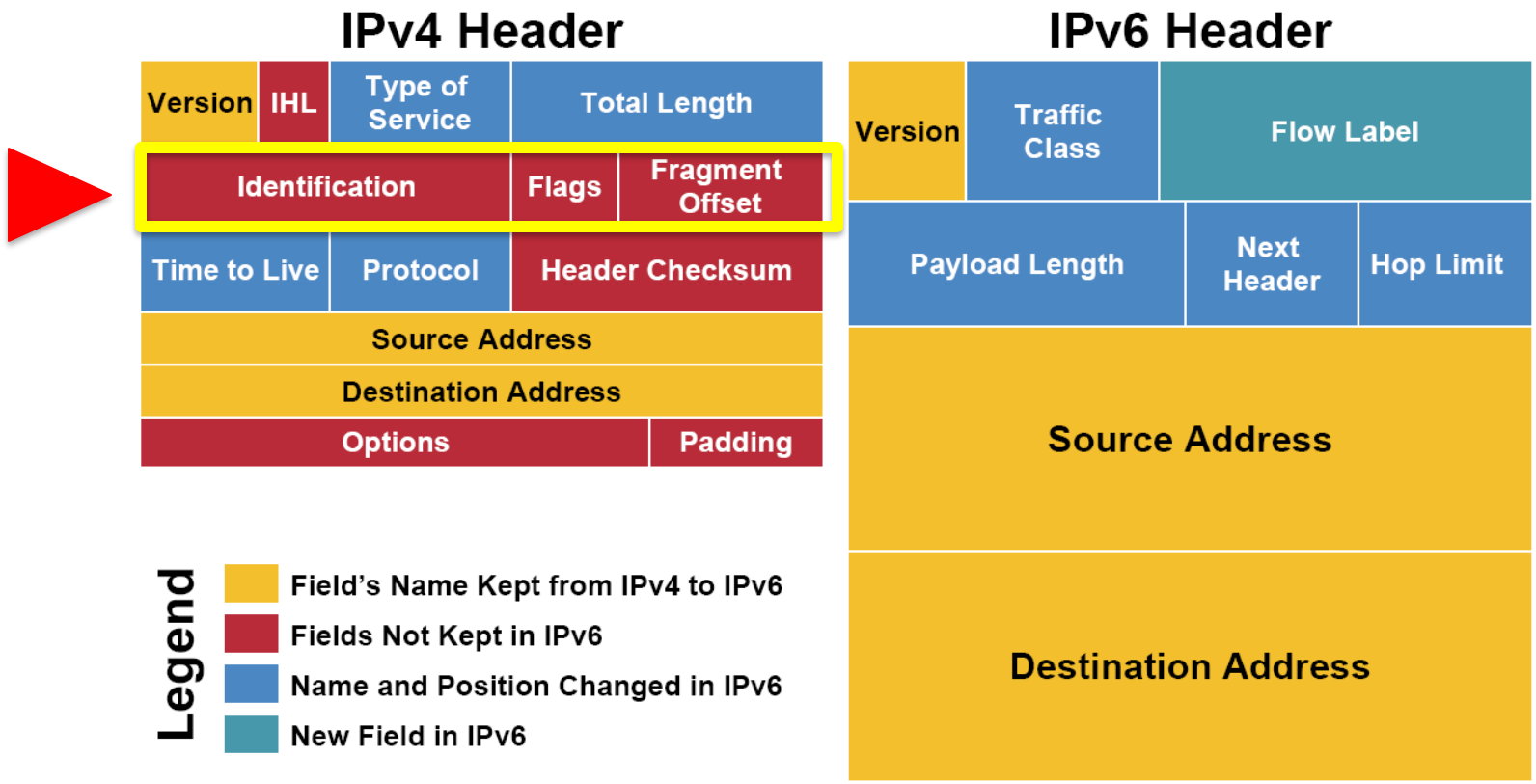
段海新

2024 年 10 月

CONTENTS

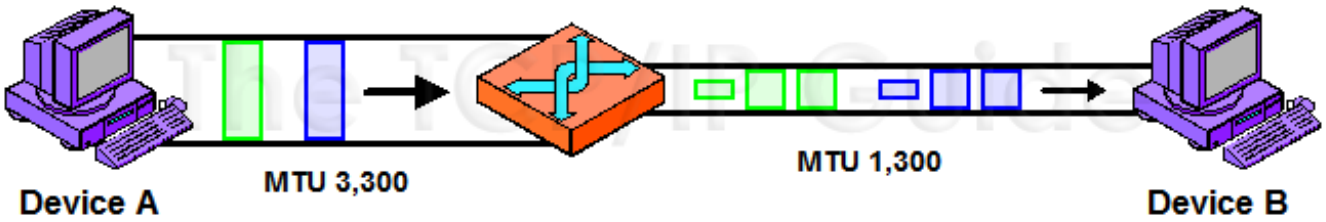
- IP 协议的安全问题
  - Fragment Problems
  - IP ID problems
  - NIDS evasion
- TCP 协议的安全问题
  - TCB maintenance Problems
    - SYN Flood
    - Connection de-synchronize
  - TCP Sequence Number prediction
    - Reset, Hijack

Differences between IPv4 &IPv6 Packet

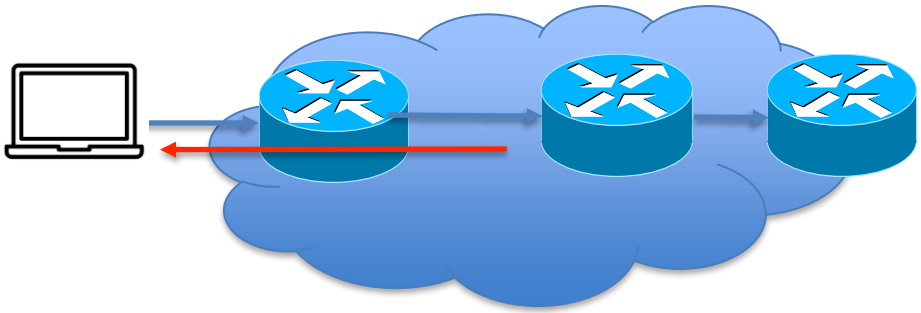


Why Fragment ?

- MTU:
  - link Maximum Transmission Unit (MTU)

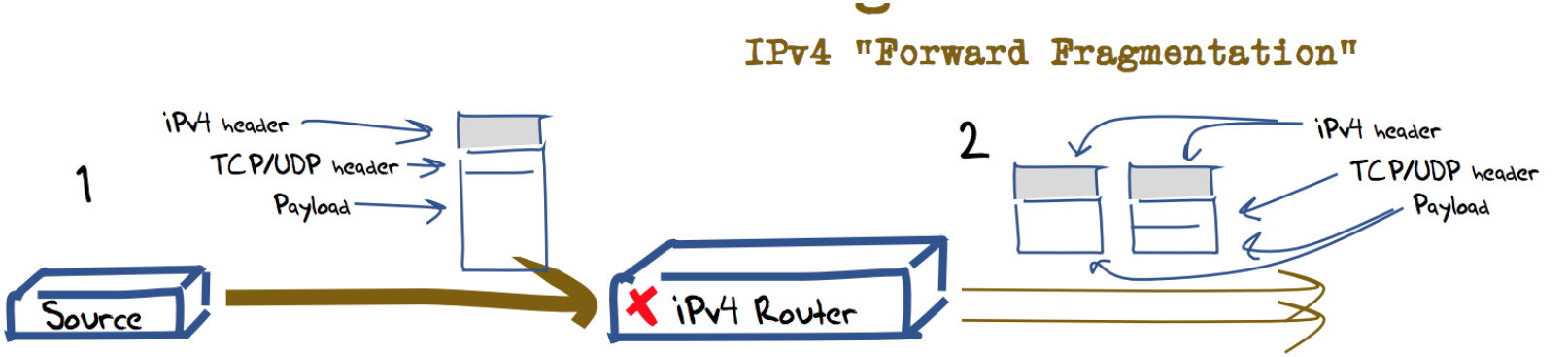


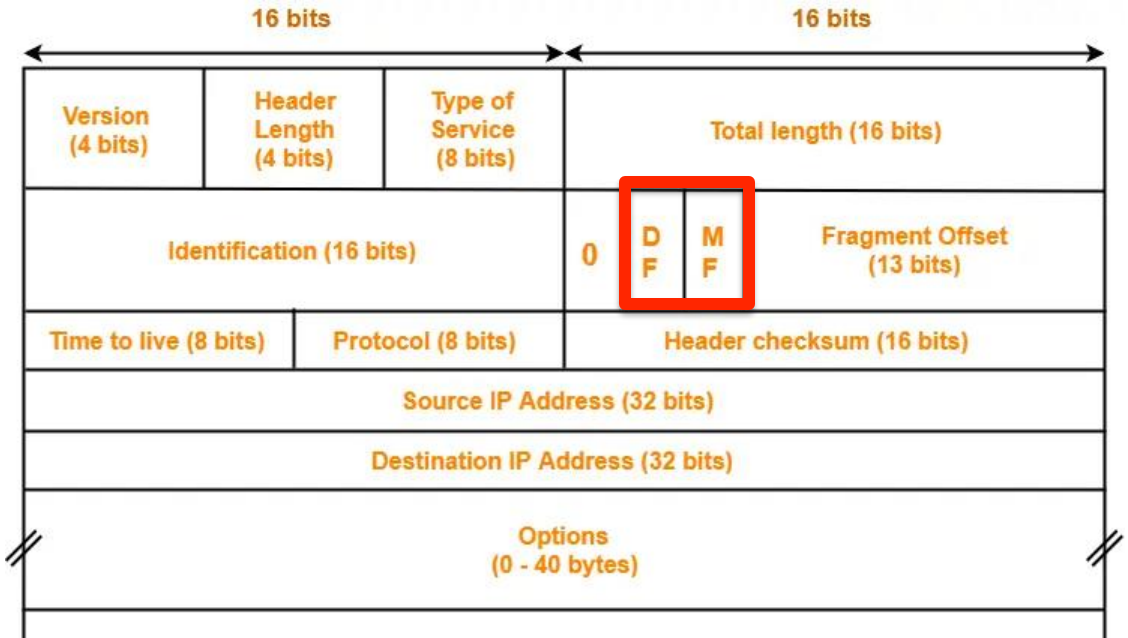
- PMTU(Path MTU)
  - Smallest MTU of the path
  - ICMP PTB(packet too big)
  - PMTU is dynamic



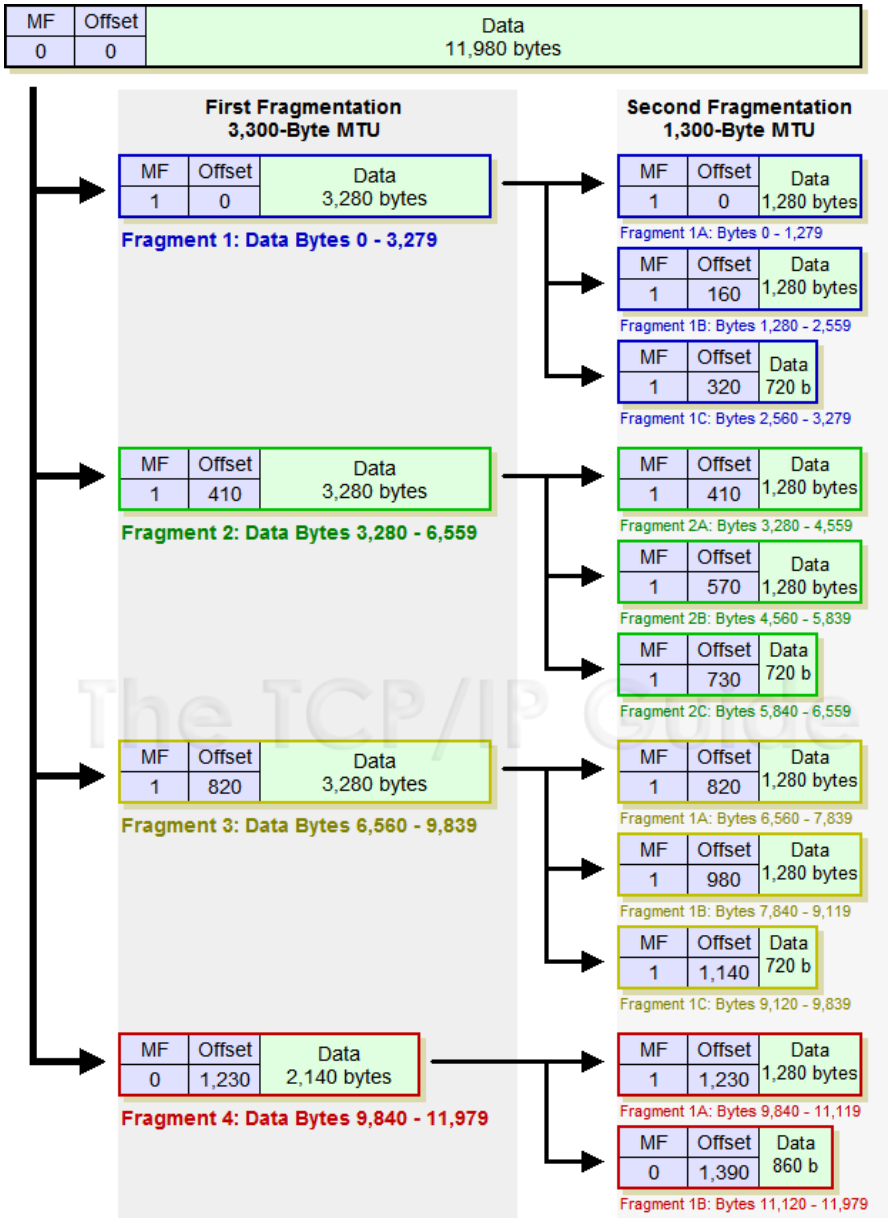
Fragment in IPv4

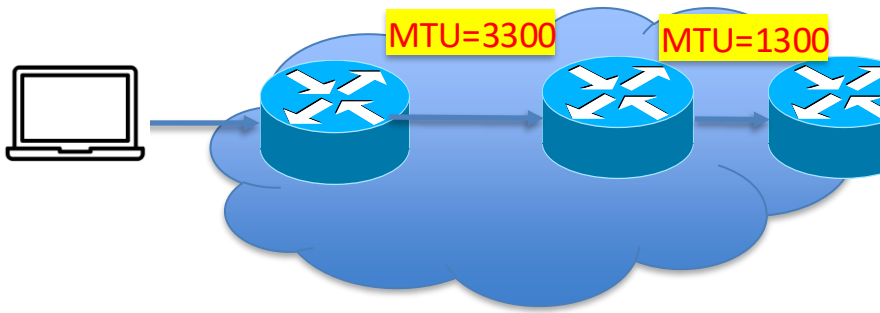
- Router can fragments packet without a DF flag





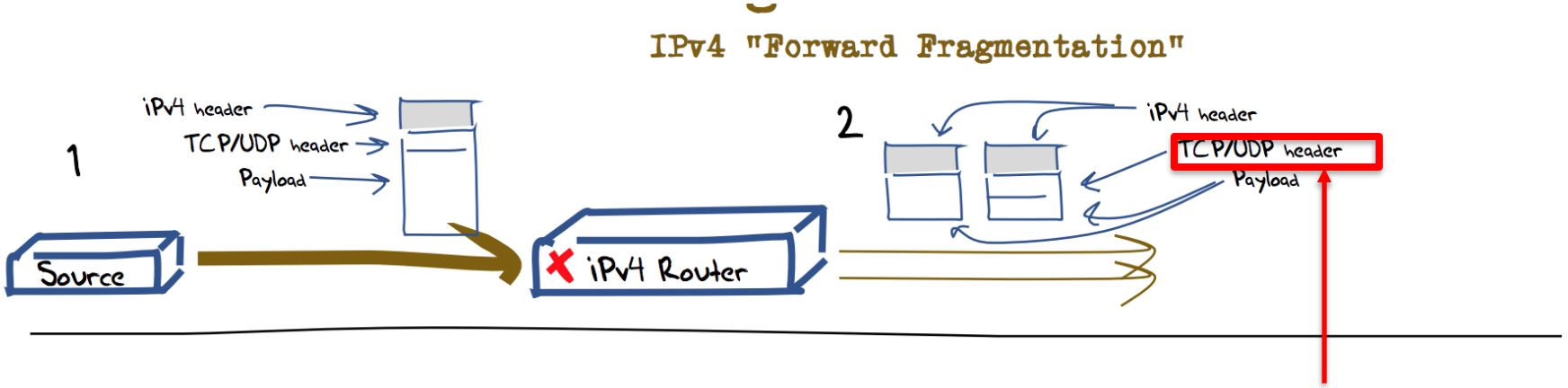
再分片





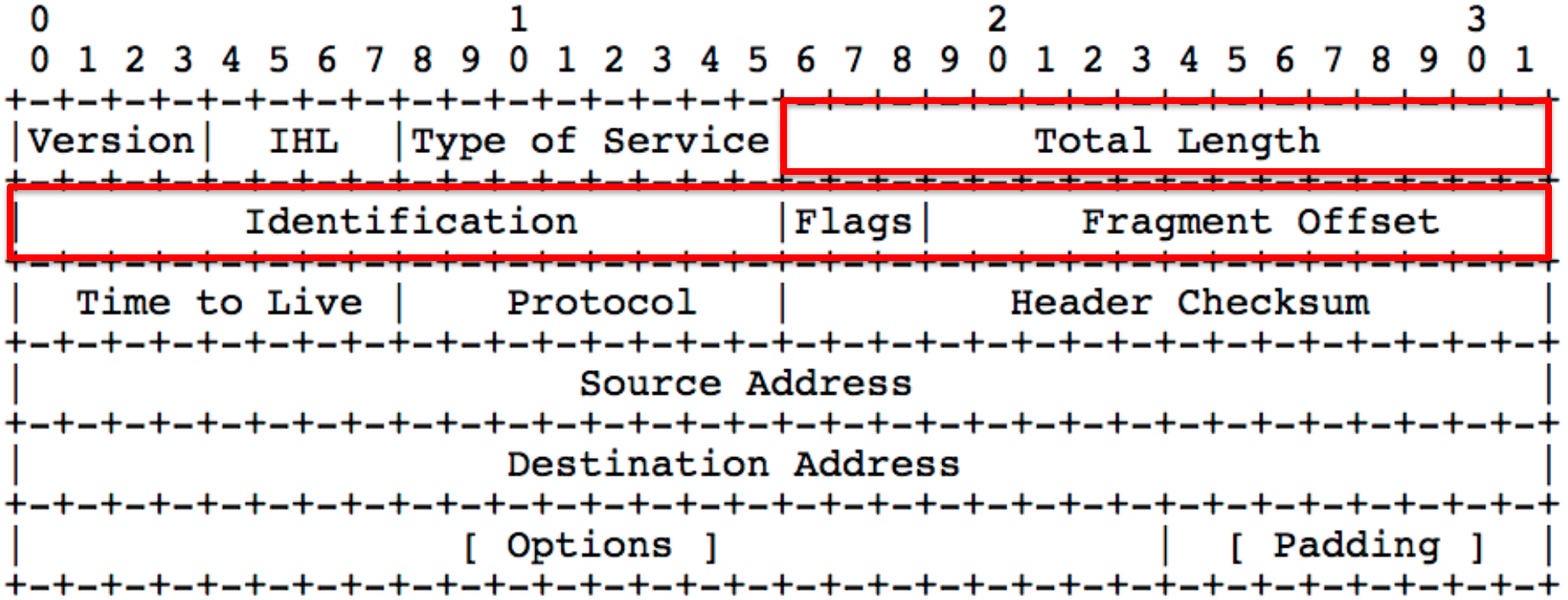
http://www.tcpipguide.com/free/t_IPMessageFragmentationProcess-2.htm

Fragment in IPv4



- Only the first fragment contains up layer headers
  - Firewall or IDS has to assemble all fragmentations
  - Otherwise, pass through or drop them?
- Reassembly is not easy…
  - Bugs, overlaps,

Ping of Death, Malachi Kenney, 1997



Attack: ping -l 65510 your.host.ip.address

TL <=65535（2^16-1）

MTU is typically 1500

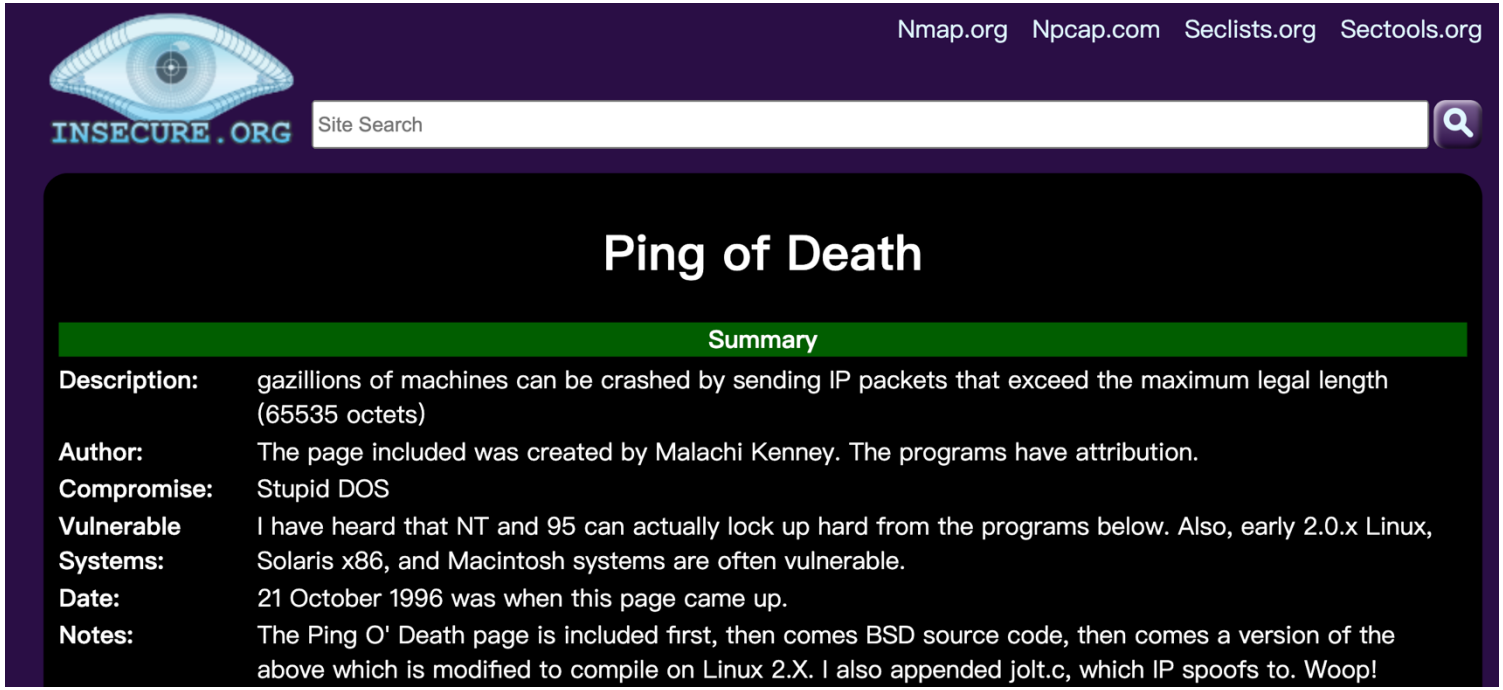
Data size <=65507（65535-20-8）

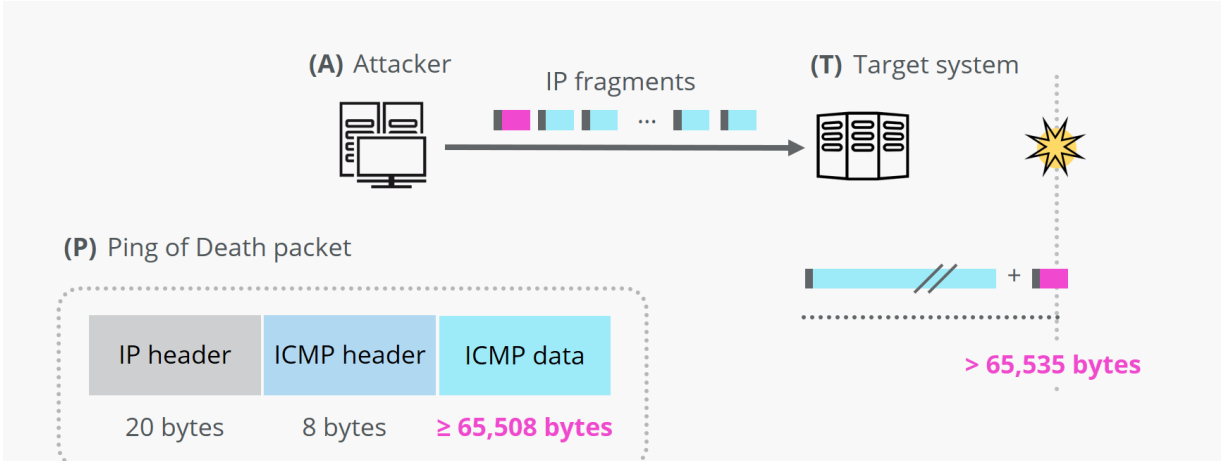
If Offset + size > 65507, overflow, crash

POC：http://insecure.org/sploits/ping-o-death.html

Demo: https://www.youtube.com/watch?v=FzuFYdDUjsQ

Ping of Death, Malachi Kenney, 1997



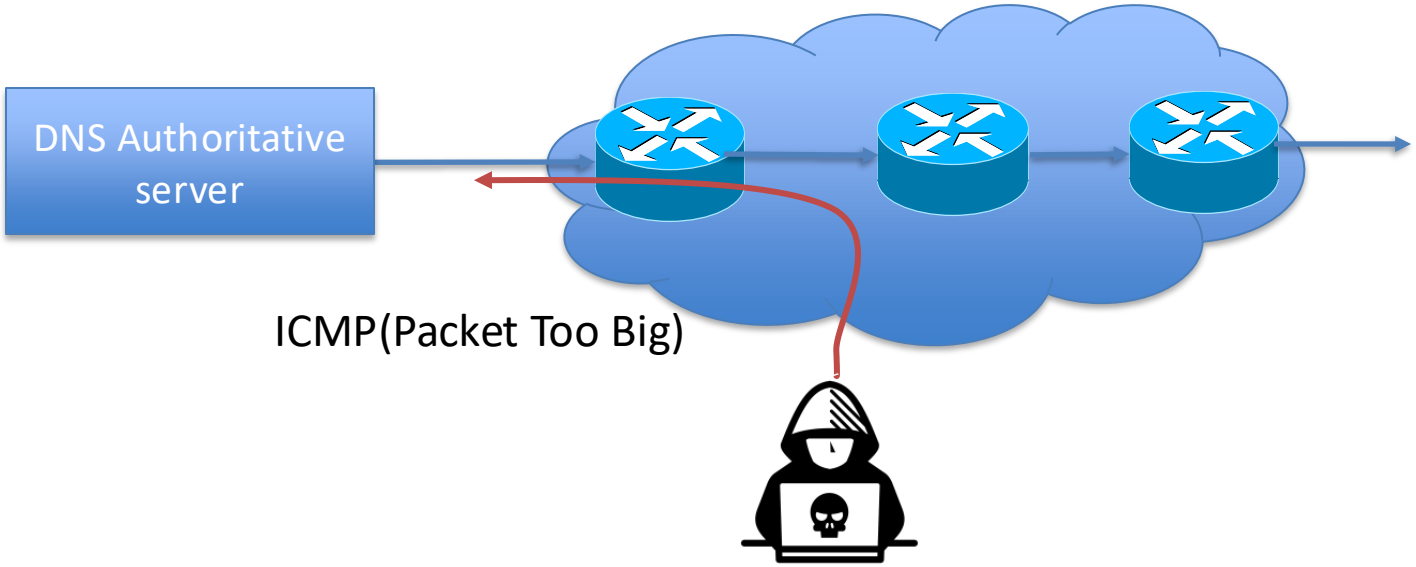


Ping of Death, Malachi Kenney, 1997

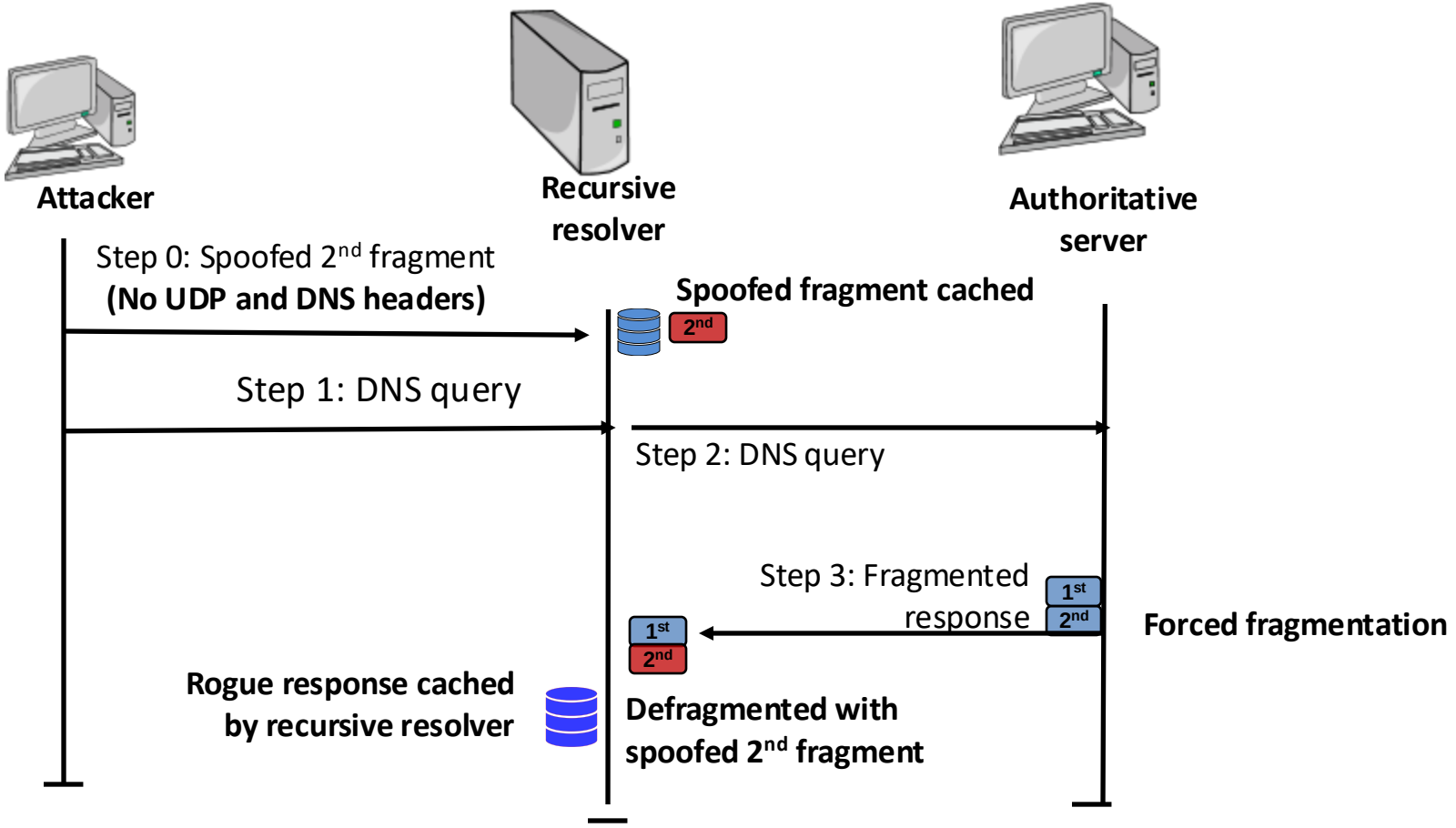
Security Concerns of IP fragmentation

DoS attacks

- Attacker sends fragmented packets to victim
  - Attack flow is optimized to consume resources on victim platform
- Attacker spoofs PTB message to victim's legitimate communication partners (Packet Too Big)
  - Causes legitimate communication partners to fragment packets that don't need to be fragmented

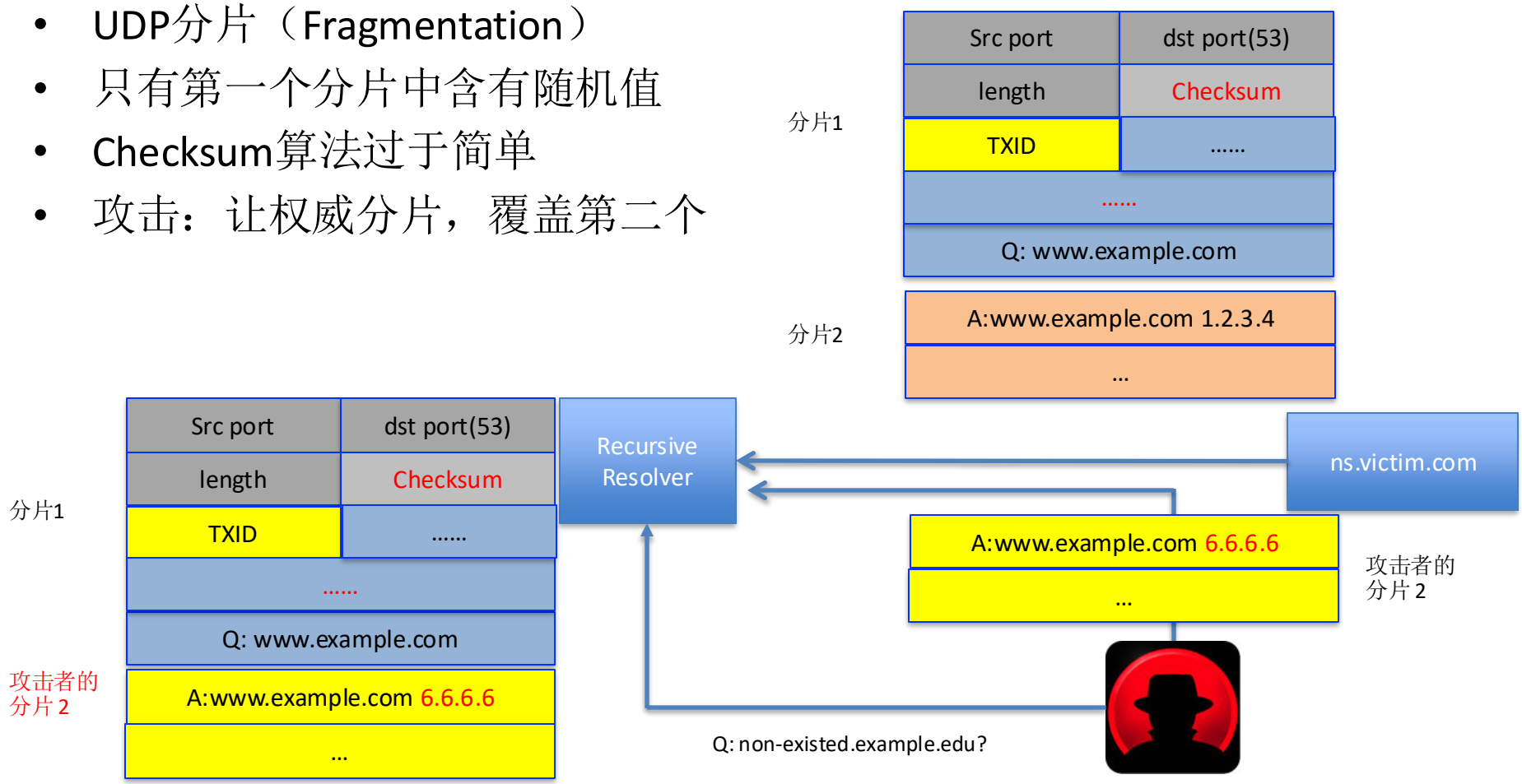


基于分片的域名缓存污染



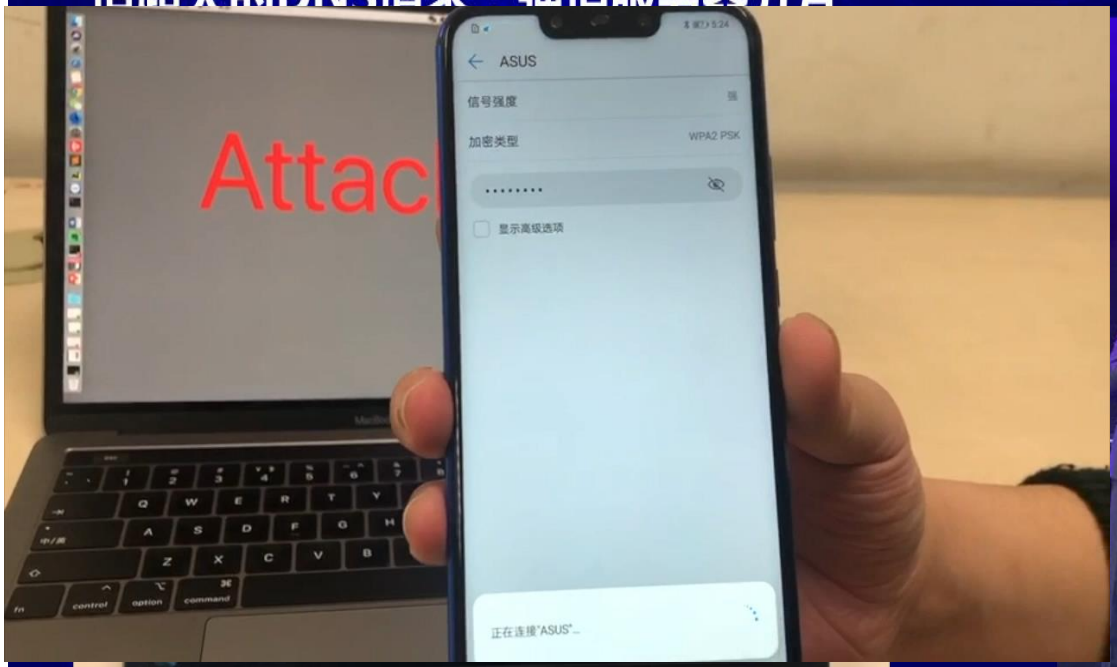
DNS Cache Poisoning by IP fragment

- UDP分片（Fragmentation）
- 只有第一个分片中含有随机值
- Checksum算法过于简单
- 攻击：让权威分片，覆盖第二个



GeekPwn 2018

- 清华-奇安信联合实验室发现的新型DNS缓存污染攻击：构造超大的DNS请求，强迫服务器分片



小米 飞鱼星 ASUS D-Link TP-LINK Tenda 腾达 motorola

HUAWEI MERCURY 水星 MerCruiser LINKSYS WAVLINK 睿因 FAST 迅捷 极路由

USENIX Security 2020

Poison Over Troubled Forwarders:

A Cache Poisoning Attack Targeting DNS Forwarding Devices

Xiaofeng Zheng, Chaoyi Lu, Jian Peng, Qiushi Yang, Dongjie Zhou, Baojun Liu, Keyu Man, Shuang Hao, Haixin Duan and Zhiyun Qian

Fragmentation Attacks

- Fragment overun
- IP fragmentation buffer full, Resource exhaustion attacks
- IP fragment overlap
- IP Fragment Too Small
- IP fragment incomplete packet

Internet Area WG	R. Bonica
Internet-Draft	Juniper Networks
Intended status: Best Current Practice	F. Baker
Expires: April 21, 2019	Unaffiliated
	G. Huston
	APNIC
	R. Hinden
	Check Point Software
	O. Troan
	Cisco
	F. Gont
	SI6 Networks
	October 18, 2018

IP Fragmentation Considered Fragile

draft-ietf-intarea-frag-fragile-02

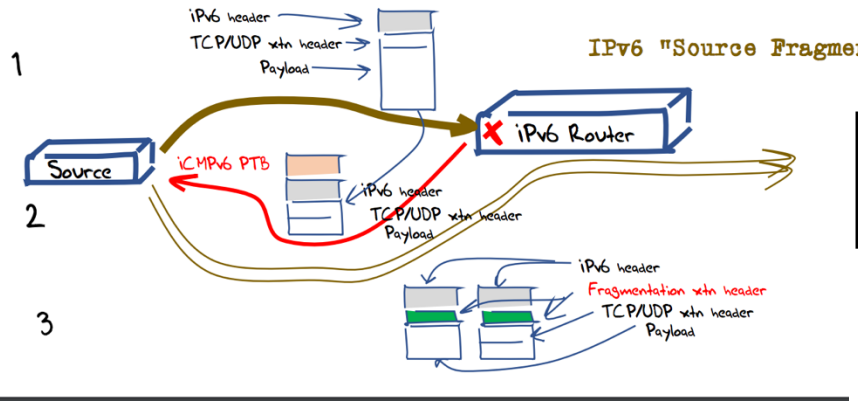
Abstract

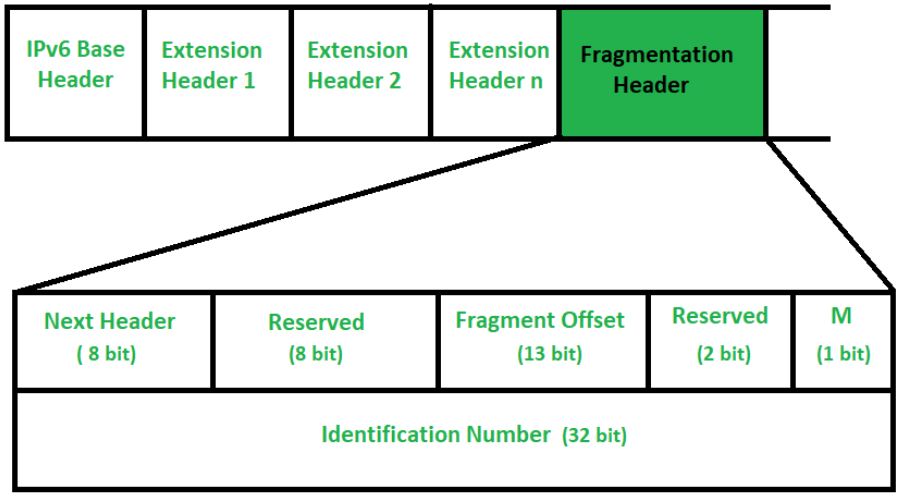
This document describes IP fragmentation and explains how it reduces the reliability of Internet communication.

This document also proposes alternatives to IP fragmentation and provides recommendations for developers and network operators.

Status of This Memo

Fragment in IPv6

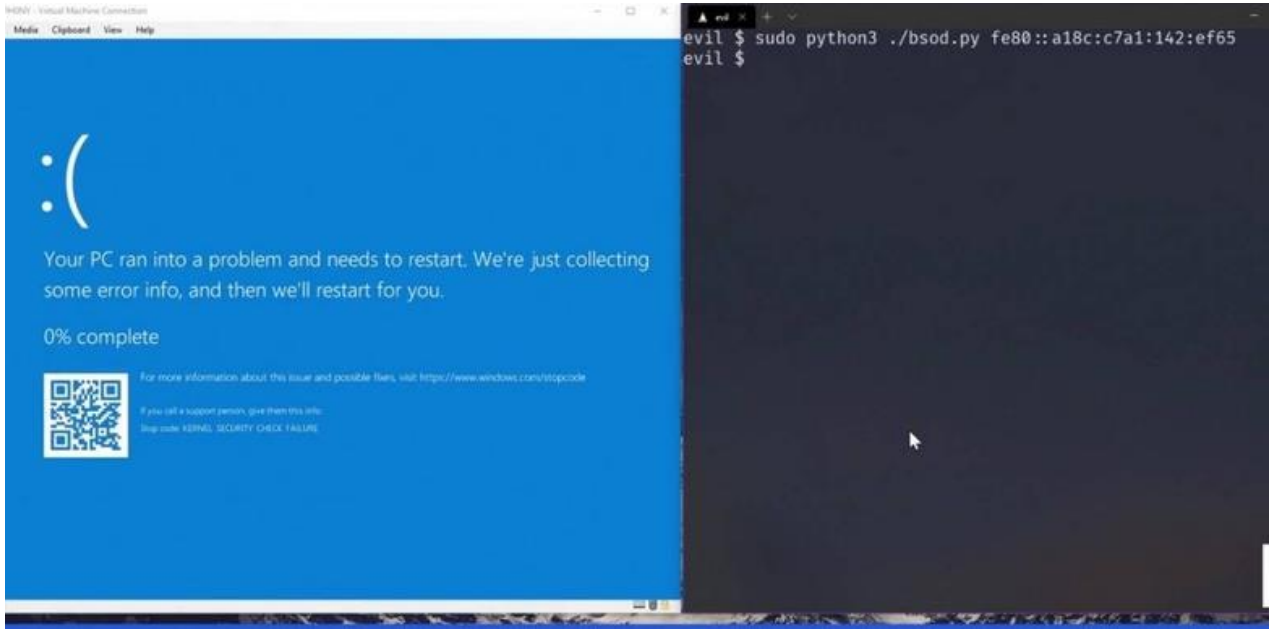




- Routers don’t fragment
- ICMPv6 Packet to Big error
- Source Fragmentation

Ping of Death v2
IPv6 Vulnerability (CVE-2020-16898/9)

- A remote code execution vulnerability exists when the Windows TCP/IP stack improperly handles ICMPv6 Router Advertisement packets. An attacker who successfully exploited this vulnerability could gain the ability to execute code on the target server or client.
- To exploit this vulnerability, an attacker would have to send specially crafted ICMPv6 Router Advertisement packets to a remote Windows computer.



- **Denial of Service(DoS)**
- **Remote Code Execution (RCE)**

Workaround

Workarounds

The following workaround may be helpful in your situation. In all cases, Microsoft strongly recommends that you install the updates for this vulnerability as soon as they become available even if you plan to leave this workaround in place:

Disable ICMPv6 RDNSS.

You can disable ICMPv6 RDNSS, to prevent attackers from exploiting the vulnerability, with the following PowerShell command. This workaround is only available for Windows 1709 and above. See What's new in Windows Server 1709 for more information.

```
netsh int ipv6 set int *INTERFACENUMBER* rabaseddnsconfig=disable
```

Note: No reboot is needed after making the change.

Impact of Workaround

The workaround disables RA-based DNS configuration. It is an alternative in networks where an IPv6 host's address is auto-configured through IPv6 stateless address auto-configuration where there is either no DHCPv6 infrastructure at all or some hosts do not have a DHCPv6 client. Windows still supports DHCPv6 and it takes precedence over 6106-based configuration.

Before applying the workaround, customers need to consult with their IT admin to confirm that their network infrastructure doesn't rely on RA-based DNS configuration. Refer to RFC 8106 for further detail.

How to undo the workaround

You can disable the workaround with the following PowerShell:

```
netsh int ipv6 set int *INTERFACENUMBER* rabaseddnsconfig=enable
```

Note: No reboot is needed after disabling the workaround.

https://msrc.microsoft.com/update-guide/en-US/advisory/CVE-2020-16898

Internet Engineering Task Force (IETF)
Request for Comments: 8021
Category: Informational
ISSN: 2070-1721

F. Gont
SI6 Networks / UTN-FRH
W. Liu
Huawei Technologies
T. Anderson
Redpill Linpro

January 2017

Generation of IPv6 Atomic Fragments Considered Harmful

Abstract

This document discusses the security implications of the generation of IPv6 atomic fragments and a number of interoperability issues associated with IPv6 atomic fragments. It concludes that the aforementioned functionality is undesirable and thus documents the motivation for removing this functionality from an upcoming revision of the core IPv6 protocol specification (RFC 2460).

https://www.rfc-editor.org/rfc/rfc8021.txt

So,

Disable fragmentation ?

IP ID, information leakage

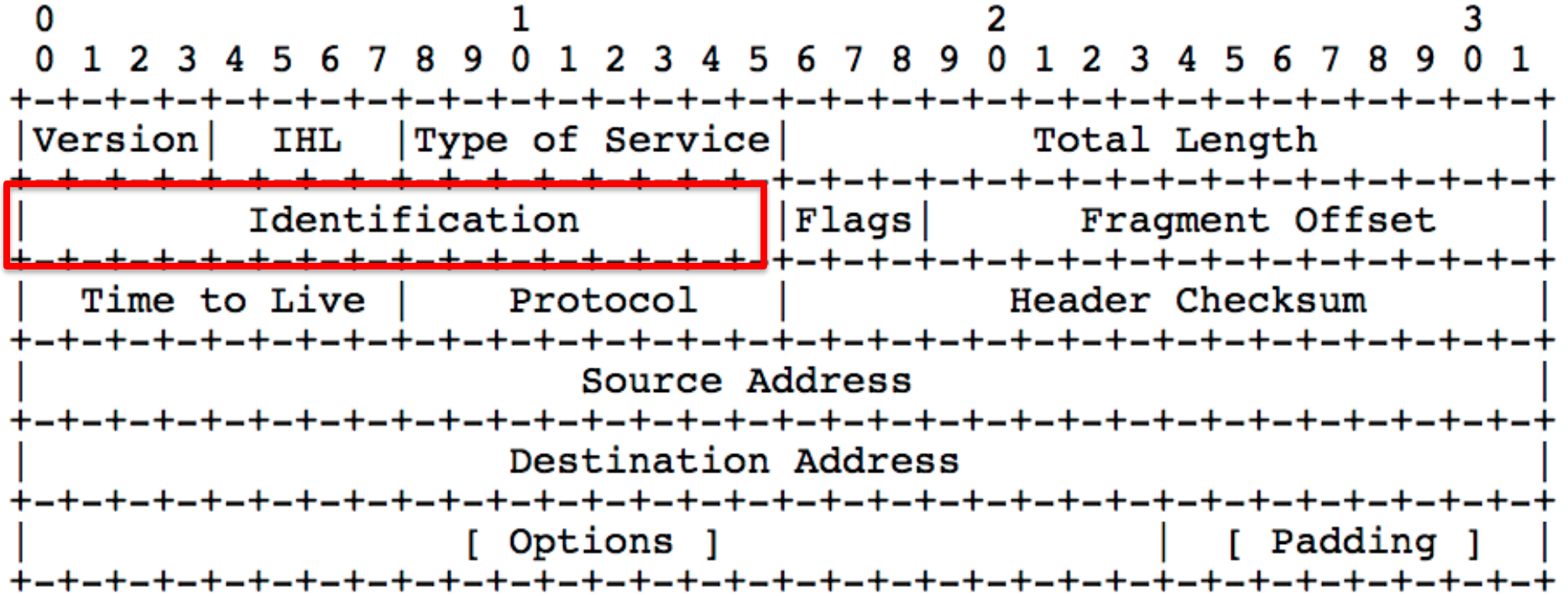
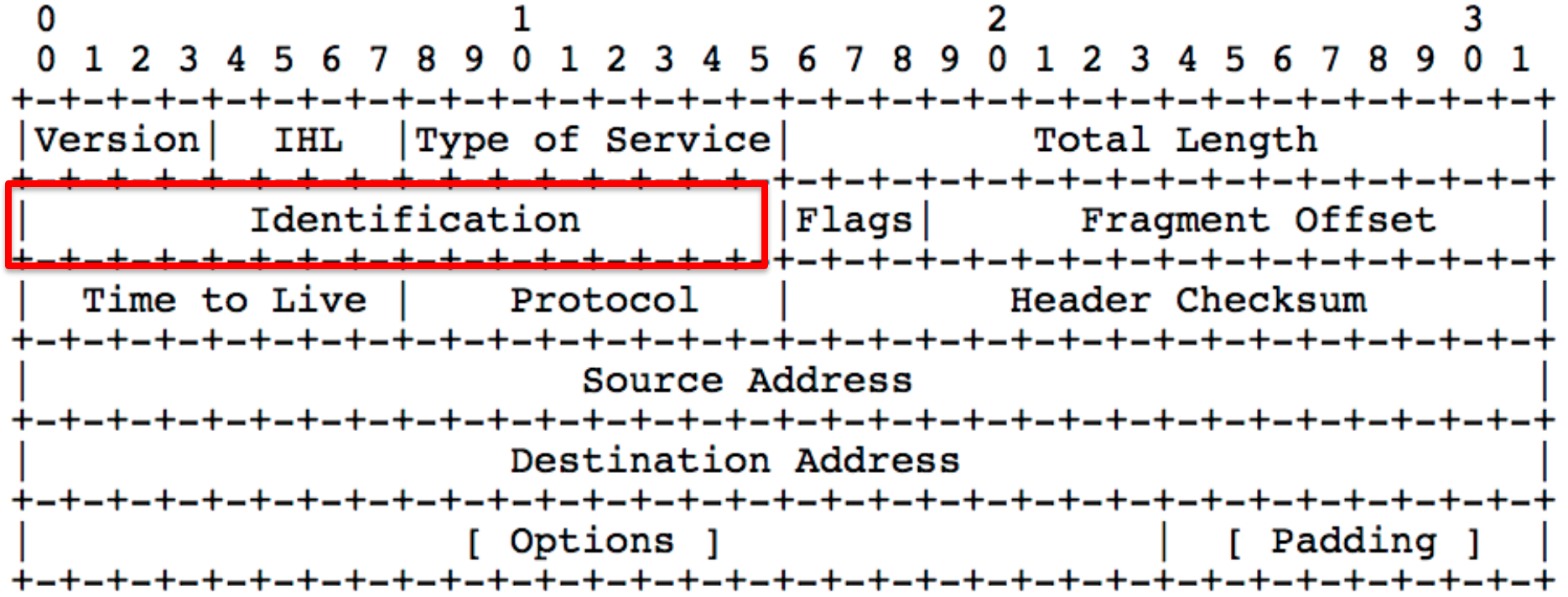


Figure 1: Internet Protocol Header Format

so what ?

- Scanning…
- TCP hijacking
- UDP data Injection
- …

IP ID, information leakage



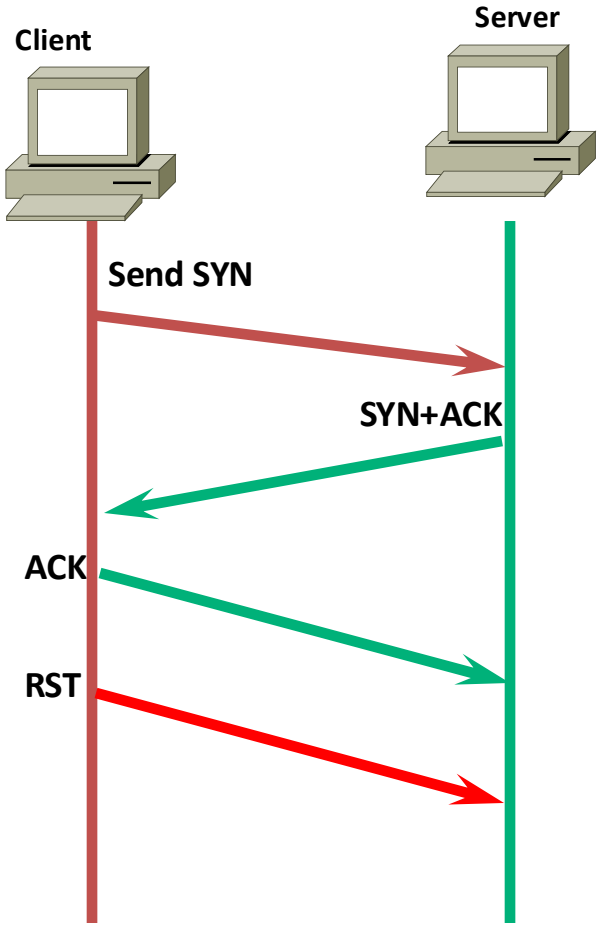
many unix, **increase the ID** by one for each packet sent. It is possible to approximate the number of outgoing packets from a remote host by performing the following:

- send one packet that presuppose a replies from target host
- record reply's ip_hdr id in X
- send another packet on second after the first.
- record reply's ip_hdr in Y
- if Y >= X then sent packets = Y - X
- if Y < X then sent packets = (65535 - X) + Y /* id rewind */

IP ID as a Side Channel

Port Scan Technologies

- TCP connection



A Review of Port Scanning Techniques

Marco de Vivo mdevivo@reacciun.ve
Apartado Postal 68274
Caracas, Venezuela.

Eddy Carrasco ecarrasco@ciens.ucv.ve

Germinal Isern gisern@ciens.ucv.ve

Gabriela O. de Vivo gdevivo@reacciun.ve

LACORE U.C.V.

Abstract *

This paper reports the most important techniques used by **TCP *port scanners***. TCP port scanners are specialized programs used to determine what TCP ports of a host have processes *listening on* them for possible connections. Since these ports characterize, in part, the amount of exposure of the hosts to potential external attacks, knowing their existence is a fundamental matter for network and/or security administrators. Moreover, as scanners are also used by hackers, administrators need to know how they work and what possible weakness they exploit to be able to prevent unwanted scanning or at least to record each scanning attempt.

Keywords: TCP/IP, UDP, Three-way Handshake, SYN Scanning, FIN Scanning, Stealth Scanning, Indirect Scanning, Decoy Scanning, Fingerprinting, and Coordinated Scanning.

Introduction.

In the first section we review the TCP connection establishment process (often called the *three-way handshake*), and discuss some implementation details that are relevant to the design of scanner programs.
Next, classical scanners (*full TCP connection*) are reviewed as well as the so-called ***SYN*** (or "*half open*") scanners.
A third section is devoted to study ***indirect*** and ***stealth*** scanning; techniques developed to conceal scanning attacks and/or their origin.

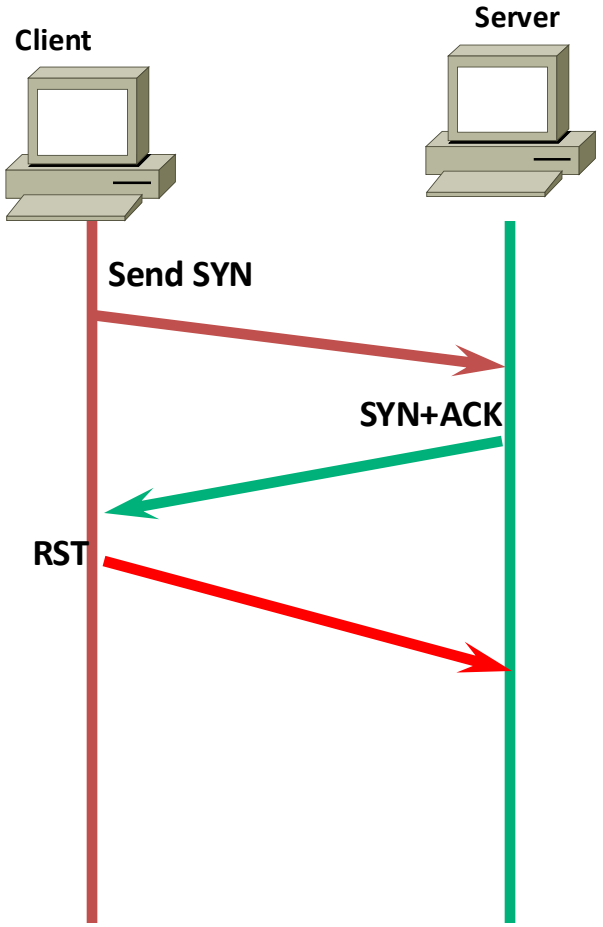
Stealth scanners are based on the use of ***FIN*** segments. The idea is that in most implementations, closed TCP ports reply to a FIN segment with a ***RST***, while open ports usually discard the segment with no reply at all. Indirect scanning, as will be explained, is realized with the (usually involuntary) help of a spoofed host.
In the next section several scanning techniques developed to bypass *firewalls* analysis and/or filtering are discussed. We first explain the use of IP fragmentation to split up the TCP header of a SYN or FIN probe over several packets. Then ***decoy*** scanning is presented and finally several forms of ***coordinated*** scanning are analyzed.
The fifth section is dedicated to examine the basics of UDP scanning.
The sixth section presents several forms of scanning related to specific *application level protocols*. Usually these scanners exploit weak and/or faulty implementations as well as permissive features. The ***ident*** (or *reverse ident*) scanning is explained as well as the ***proxy*** based scanning approach.
In the last section, additional scanning techniques are reviewed. Scanning tools not based only on TCP *port* abstraction or designed mainly for ***security scanning*** are considered (e.g., **SATAN**). Also the use of scanners for ***stack fingerprinting*** is analyzed. Stack fingerprinting solves the problem of operating systems identification in a unique way, by probing a host's TCP and comparing the answer (or the lack of answer) against the results expected from known operating systems TCP/IP stacks.

* This work was partially supported by C.D.C.H. (U.C.V.) under Grant No. 03-13-4051-97

1.Vivo, M. de, Carrasco, E., Isern, G. & Vivo, G. O. de. A review of port scanning techniques. *ACM SIGCOMM Comput. Commun. Rev.* **29**, 41–48 (1999).

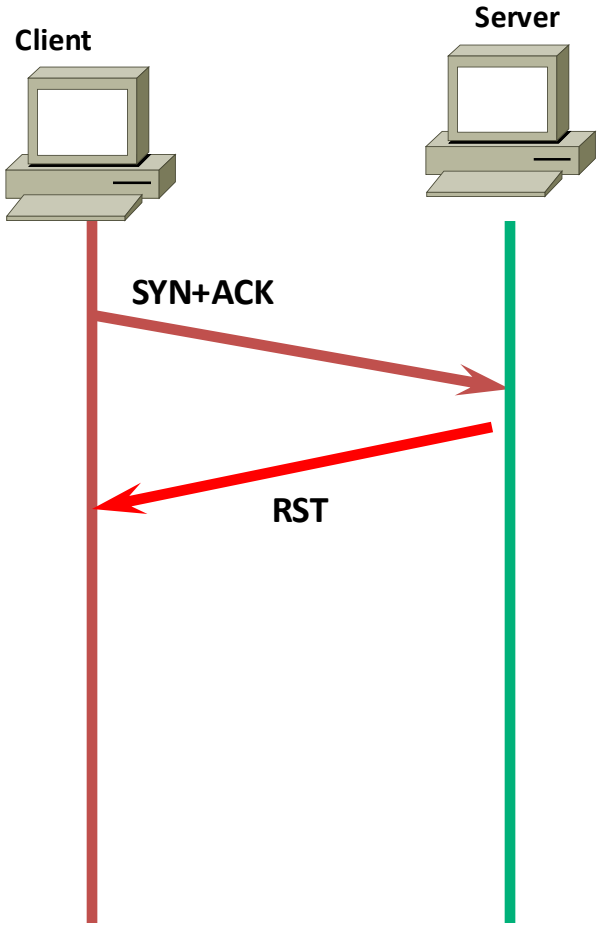
Port Scan Technologies

- TCP connection
- TCP SYN scan



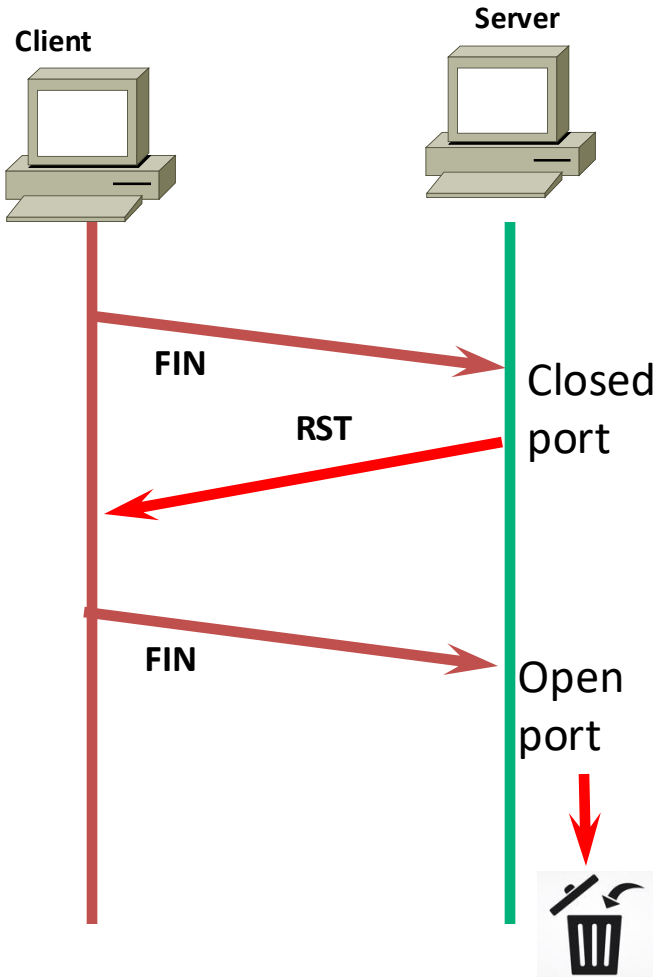
Port Scan Technologies

- TCP connection
- TCP SYN scan
- SYN-ACK scan



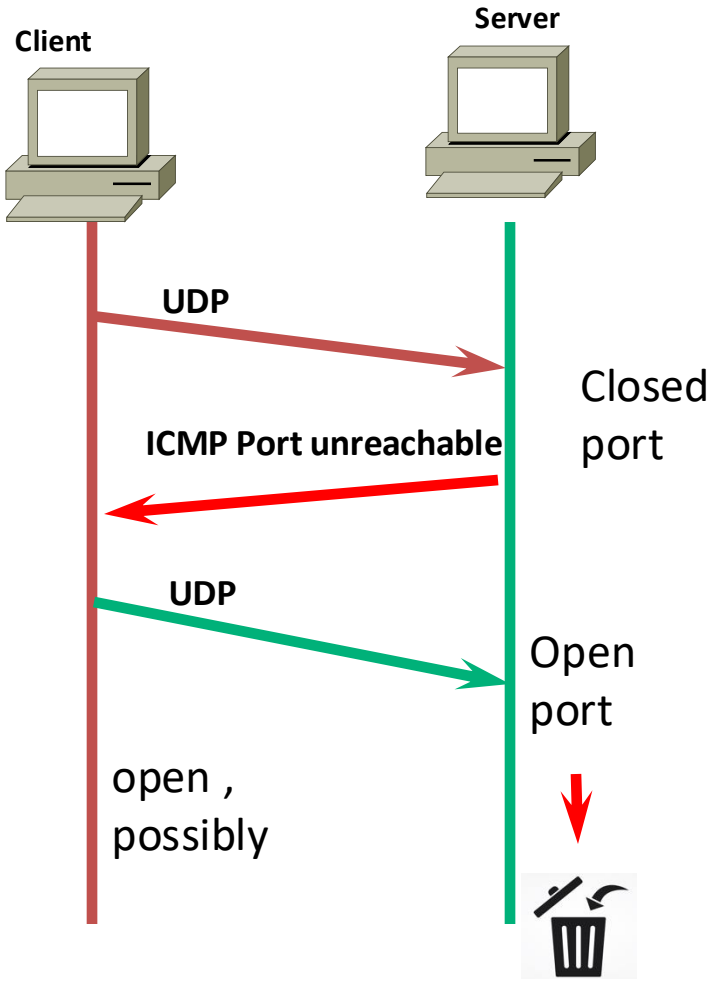
Port Scan Technologies

- TCP connection
- TCP SYN scan
- SYN-ACK scan
- FIN scan
- Xmas Tree, Null scan
  - FIN/PSH/URG
  - No flag



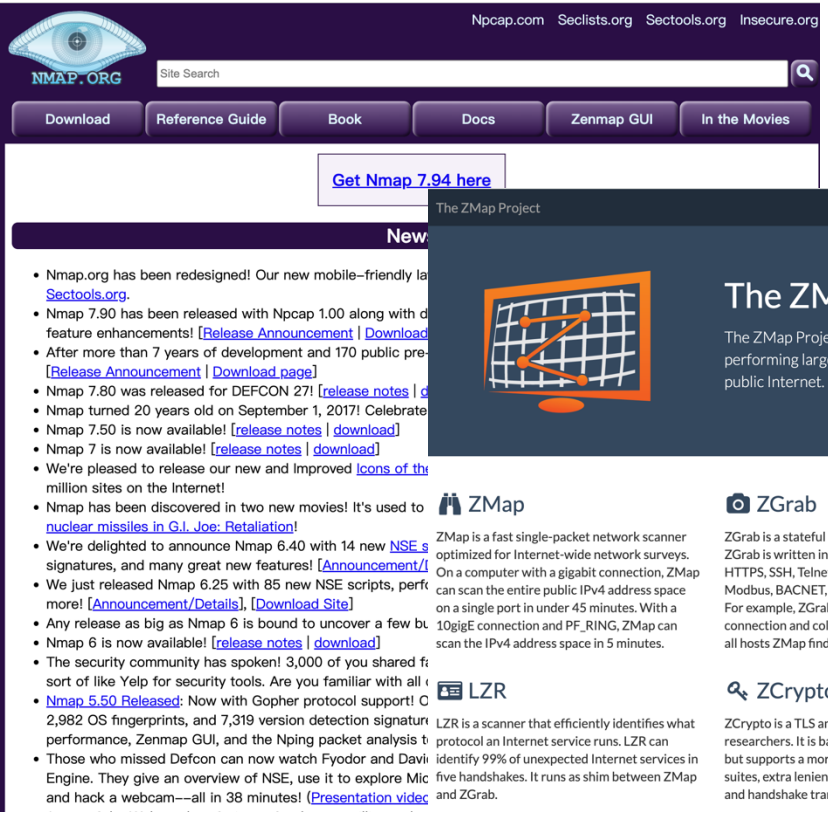
Port Scan Technologies

- TCP connection
- TCP SYN scan
- SYN-ACK scan
- FIN scan
- Xmas Tree, Null scan
  - FIN/PSH/URG
  - No flag
- UDP scan

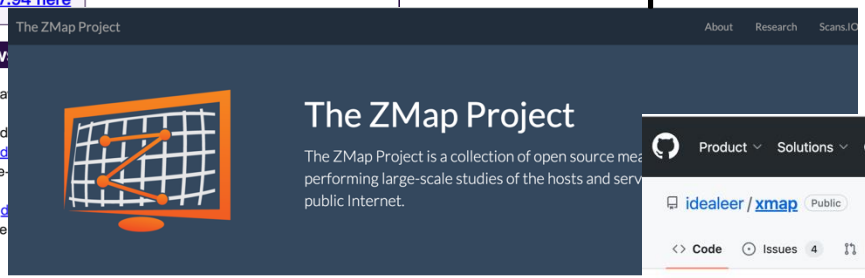


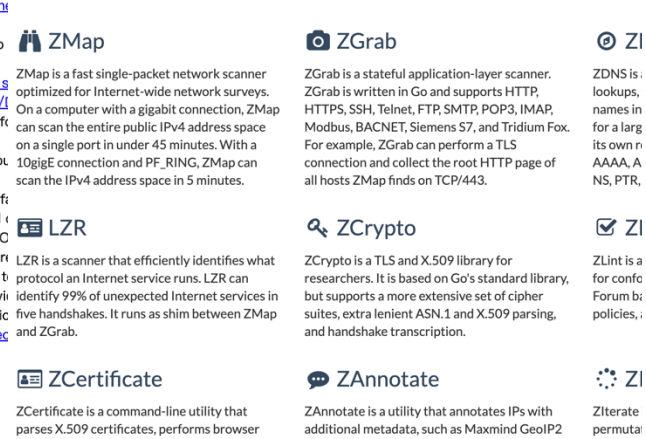
Internet Scanners

Nmap.org

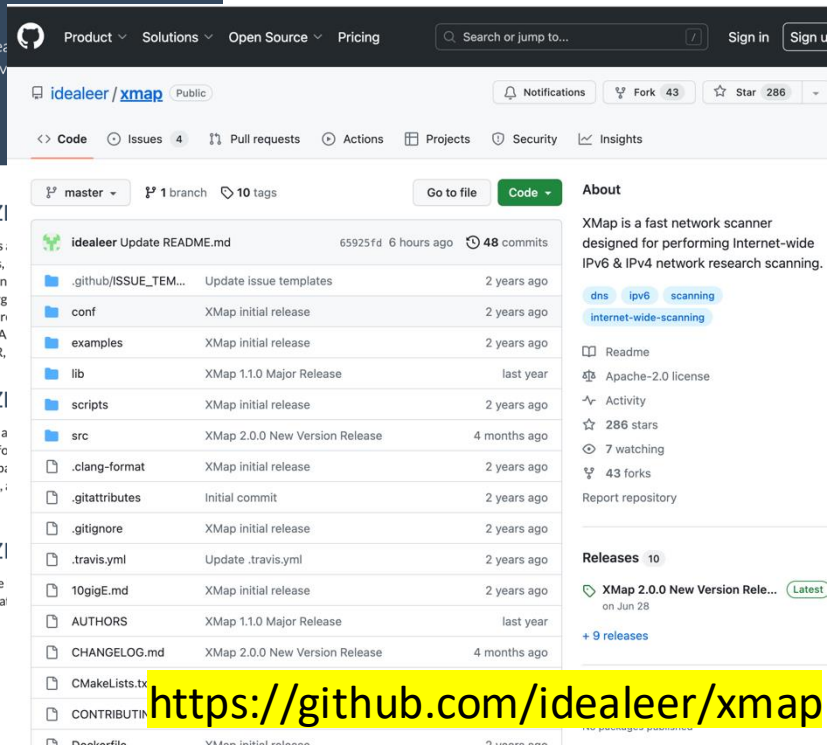


zmap.io



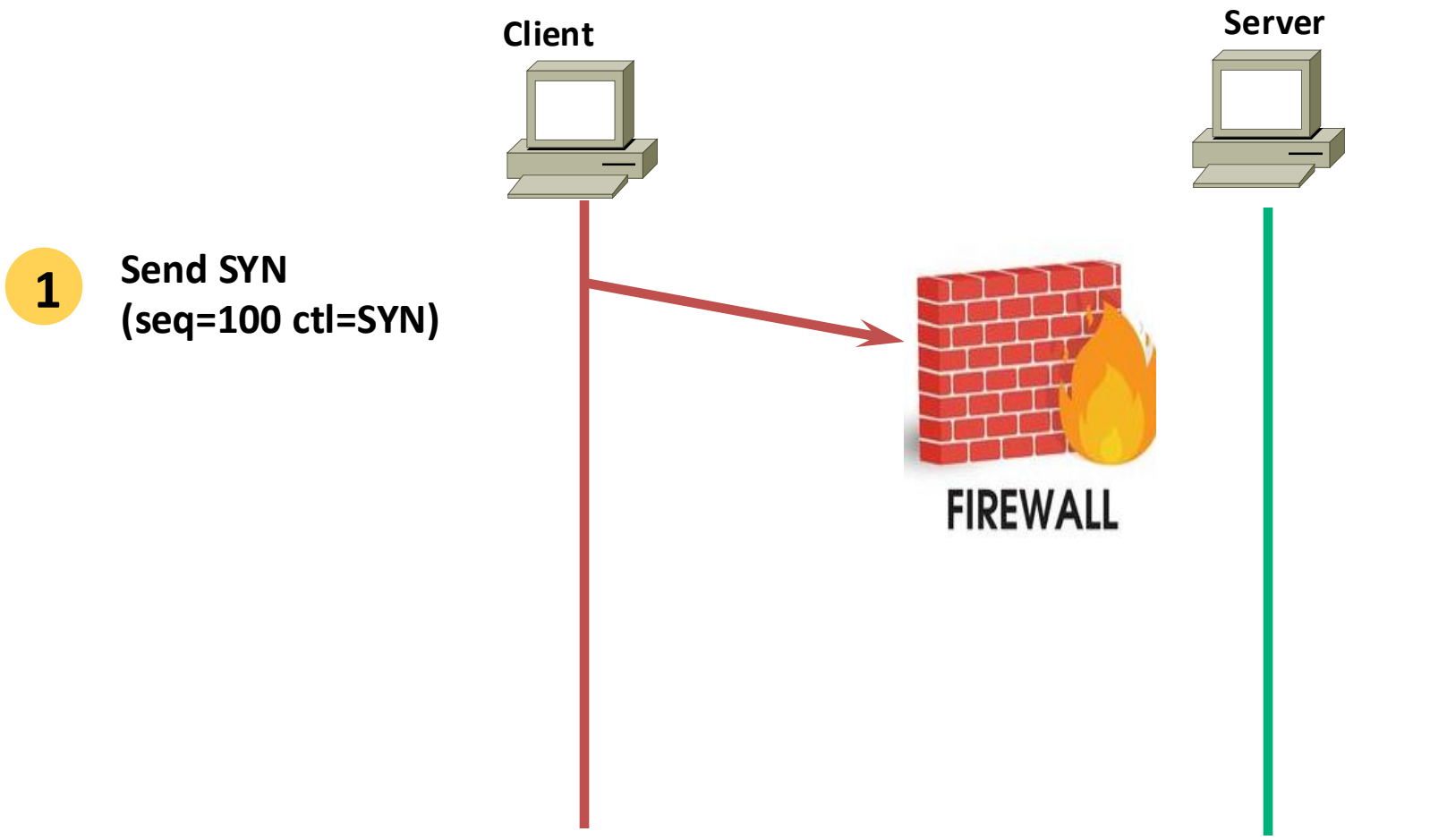


Xmap



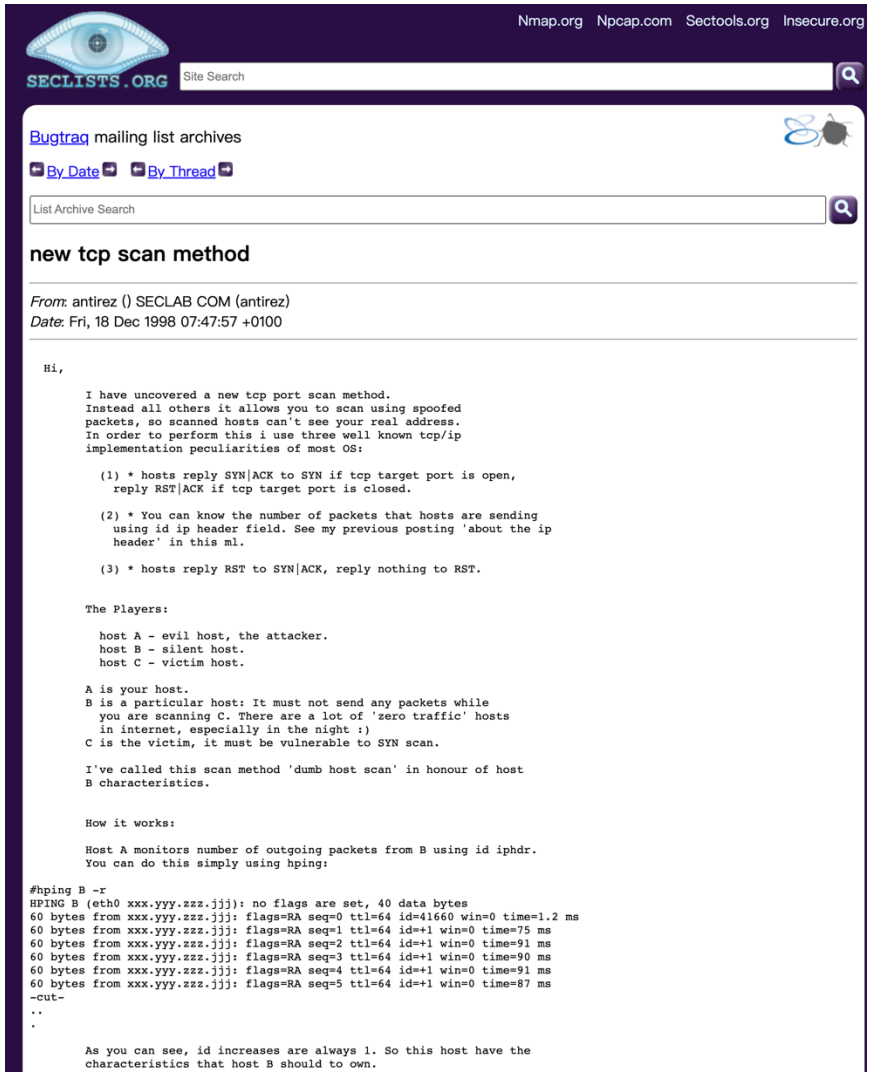
https://github.com/idealeer/xmap

Hosts behind firewalls



How to know whether a port is open or not?

Antirez, **new tcp scan method, 1998**



Nmap.org Npcap.com Sectools.org Insecure.org

SECLISTS.ORG Site Search

Bugtraq mailing list archives

By Date By Thread

List Archive Search

new tcp scan method

From: antirez () SECLAB COM (antirez)
Date: Fri, 18 Dec 1998 07:47:57 +0100

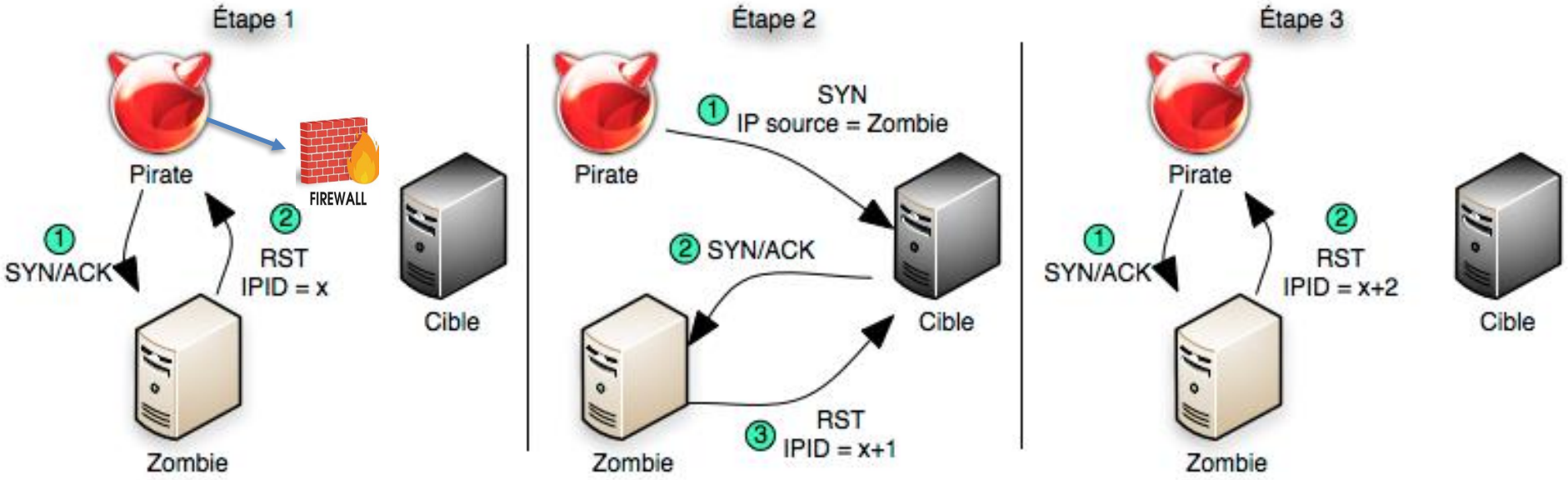
```
Hi,

      I have uncovered a new tcp port scan method.
      Instead all others it allows you to scan using spoofed
      packets, so scanned hosts can't see your real address.
      In order to perform this i use three well known tcp/ip
      implementation peculiarities of most OS:

        (1) * hosts reply SYN|ACK to SYN if tcp target port is open,
          reply RST|ACK if tcp target port is closed.

        (2) * You can know the number of packets that hosts are sending
          using id ip header field. See my previous posting 'about the ip
          header' in this ml.

        (3) * hosts reply RST to SYN|ACK, reply nothing to RST.


      The Players:

        host A - evil host, the attacker.
        host B - silent host.
        host C - victim host.

      A is your host.
      B is a particular host: It must not send any packets while
        you are scanning C. There are a lot of 'zero traffic' hosts
        in internet, especially in the night :)
      C is the victim, it must be vulnerable to SYN scan.

      I've called this scan method 'dumb host scan' in honour of host
      B characteristics.


      How it works:

      Host A monitors number of outgoing packets from B using id iphdr.
      You can do this simply using hping:

#hping B -r
HPING B (eth0 xxx.yyy.zzz.jjj): no flags are set, 40 data bytes
60 bytes from xxx.yyy.zzz.jjj: flags=RA seq=0 ttl=64 id=41660 win=0 time=1.2 ms
60 bytes from xxx.yyy.zzz.jjj: flags=RA seq=1 ttl=64 id=+1 win=0 time=75 ms
60 bytes from xxx.yyy.zzz.jjj: flags=RA seq=2 ttl=64 id=+1 win=0 time=91 ms
60 bytes from xxx.yyy.zzz.jjj: flags=RA seq=3 ttl=64 id=+1 win=0 time=90 ms
60 bytes from xxx.yyy.zzz.jjj: flags=RA seq=4 ttl=64 id=+1 win=0 time=91 ms
60 bytes from xxx.yyy.zzz.jjj: flags=RA seq=5 ttl=64 id=+1 win=0 time=87 ms
-cut-
..
.

      As you can see, id increases are always 1. So this host have the
      characteristics that host B should to own.

      Now host A sends SYN to port X of C spoofing from B.
      (using hping => 0.67 is very easy, http://www.kyuzz.org/antirez)
      if port X of C is open, host C will send SYN|ACK to B (yes,
      host C don't know that the real sender is A). In this
      case host B replies to SYN|ACK with a RST.
      If we send to host C a few of SYN it will reply to B with a few
      of SYN|ACK, so B will reply to C a few of RST... so
      we'll see that host B is sending packets!

.
..
-cut-
60 bytes from xxx.yyy.zzz.jjj: flags=RA seq=17 ttl=64 id=+1 win=0 time=96 ms
60 bytes from xxx.yyy.zzz.jjj: flags=RA seq=18 ttl=64 id=+1 win=0 time=80 ms
60 bytes from xxx.yyy.zzz.jjj: flags=RA seq=19 ttl=64 id=+2 win=0 time=83 ms
60 bytes from xxx.yyy.zzz.jjj: flags=RA seq=20 ttl=64 id=+3 win=0 time=94 ms
60 bytes from xxx.yyy.zzz.jjj: flags=RA seq=21 ttl=64 id=+1 win=0 time=92 ms
60 bytes from xxx.yyy.zzz.jjj: flags=RA seq=22 ttl=64 id=+2 win=0 time=82 ms
-cut-
..
.

      The port is open!

      Instead, if port X of C is closed sending to C a few
      of SYN spoofed from B, it will reply with RST to B, and
      B will not reply (see 3). So we'll see that host B is not sending
      any packet:

.
..
-cut-
60 bytes from xxx.yyy.zzz.jjj: flags=RA seq=52 ttl=64 id=+1 win=0 time=85 ms
60 bytes from xxx.yyy.zzz.jjj: flags=RA seq=53 ttl=64 id=+1 win=0 time=83 ms
60 bytes from xxx.yyy.zzz.jjj: flags=RA seq=54 ttl=64 id=+1 win=0 time=93 ms
60 bytes from xxx.yyy.zzz.jjj: flags=RA seq=55 ttl=64 id=+1 win=0 time=74 ms
60 bytes from xxx.yyy.zzz.jjj: flags=RA seq=56 ttl=64 id=+1 win=0 time=95 ms
60 bytes from xxx.yyy.zzz.jjj: flags=RA seq=57 ttl=64 id=+1 win=0 time=81 ms
-cut-
..
.

      The port is closed.

      All this can appear complicated to perform, but using two sessions
      of hping on Linux virtual consoles or under X makes it more simple.
      First session listen host B: hping B -r
      Second session send spoofed SYN: hping C -a B -S

      Sorry if my english is not so clear.
      However this posting is not adeq
      this scan method, so i'll write
      about how to implement this in a
      about players characteristics an

happy new year,
antirez

--
Salvatore Sanfilippo
Intesis SECURITY LAB            Phone: +
Via Settembrini, 35             Fax: +39-
I-20124 Milano  ITALY           Email: a
```

Idle Scan:

- Idle host
- Global IP ID
- Incremental IP ID

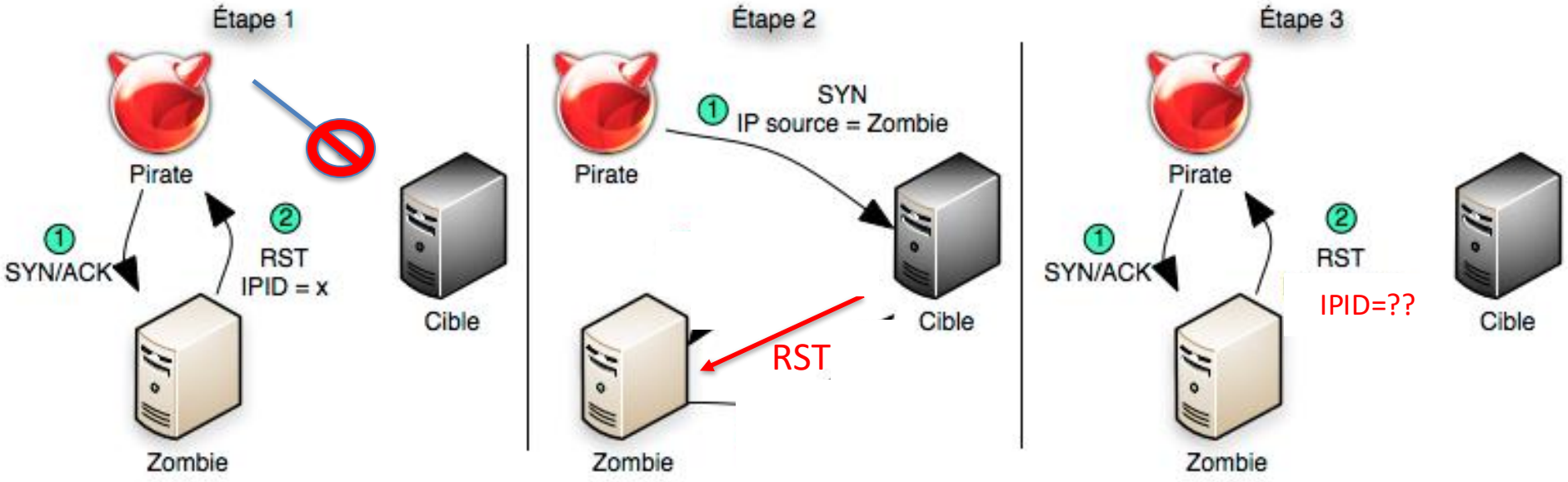
https://seclists.org/bugtraq/1998/Dec/79

Idle Scan: Is the port on Cible OPEN ?



Xu Zhang, Jefferey Knockel, Jed. Crandall. "Original SYN: Finding Machines Hidden Behind Firewalls," INFOCOM, 2015.

Idle Scan: Is the port on Cible CLOSED ?



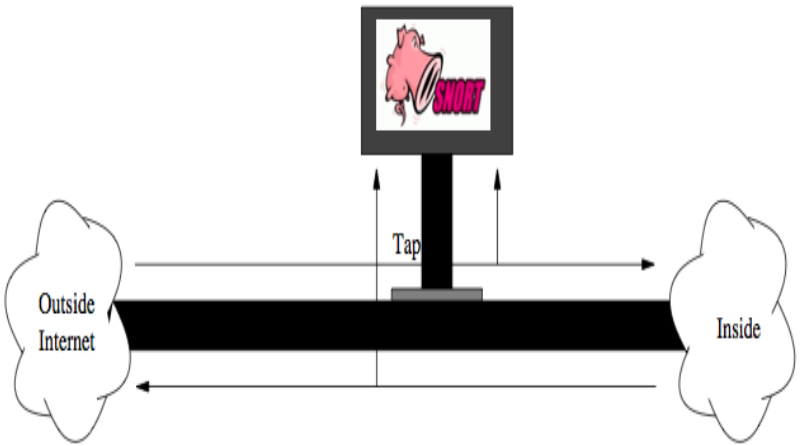
Xu Zhang, Jefferey Knockel, Jed. Crandall. "Original SYN: Finding Machines Hidden Behind Firewalls," INFOCOM, 2015.

IP ID 的生成方法

表 2　IPID 生成算法对比

生成算法	特点	优势	劣势
全局计数器	所有地址共用一个计数器	简单，高效，便于管理，性能高	容易产生分段覆盖的问题，隐蔽性弱，暴露出严重的安全问题
针对单目的地址的初值随机计数器	为每个地址维护一个 IPID 计数器，仅仅初值随机生成	简单，发生 IP ID 冲突的几率低，实现代价低，性能高	需要对每个地址的 IP ID 缓存，管理比较复杂，长连接下可以被预测
伪随机生成器	所有地址 IPID 都是随机生成的	基于伪随机数，基本不可预测	随机算法会增加 IP ID 冲突
基于 Hash 值的 IPID 生成器	每个地址初值随机，利用 Hash 保证唯一性	IPID 发生重用的概率较低，性能较高	在目的端，IPID 可预测

NIDS



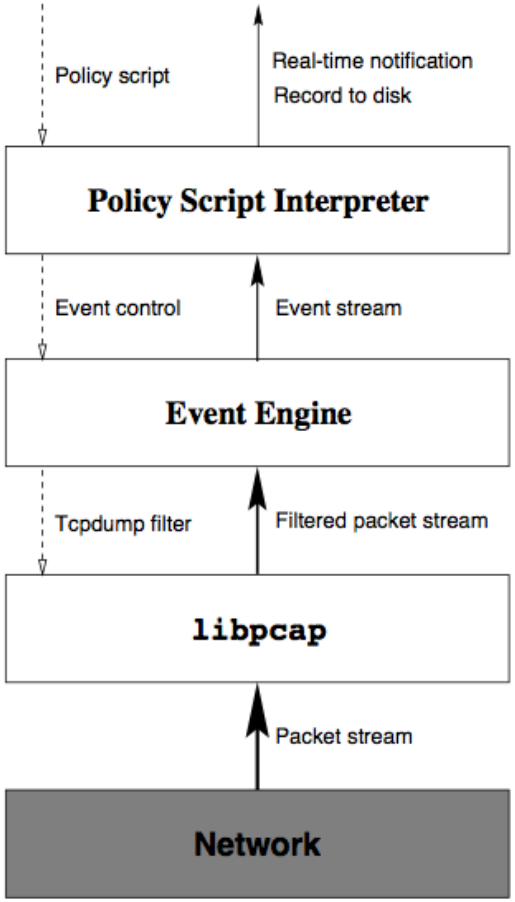
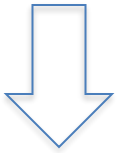


Figure 1: Structure of the Bro system

Bro



NIDS Cluster

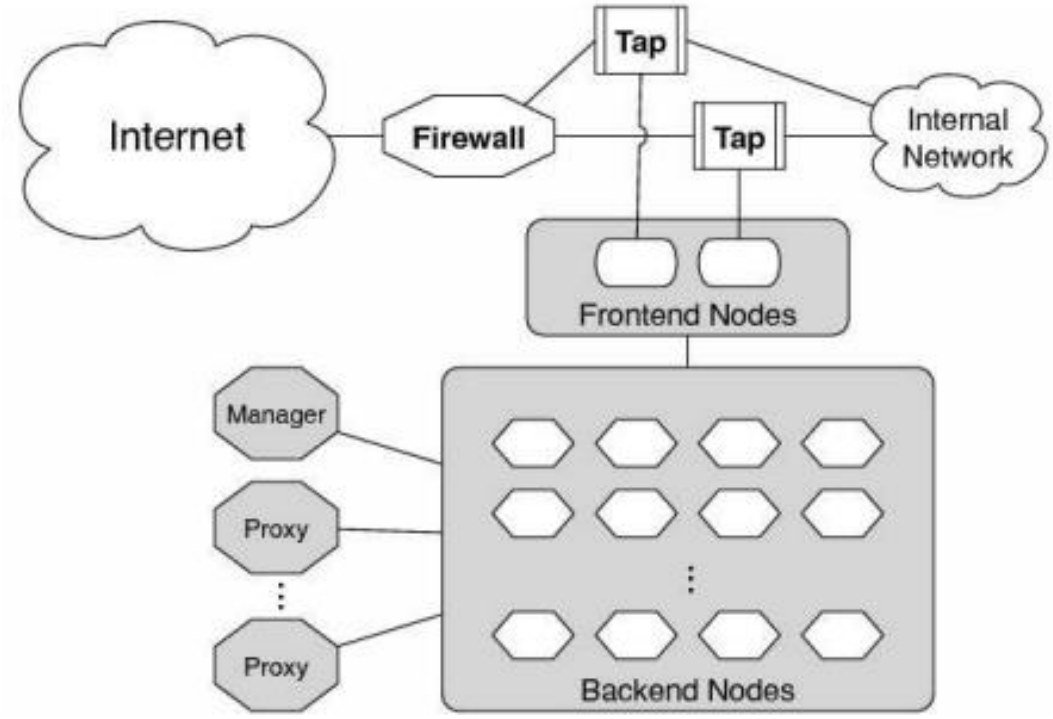


Fig. 1. Cluster architecture.

M. Vallentin, R. Sommer, J. Lee, C. Leres, V. Paxson, B. Tierney
The NIDS Cluster: Scalable, Stateful Network Intrusion Detection on Commodity Hardware, Proc. Symposium on Recent Advances in Intrusion Detection, 2007

BNIDS

[1] W. YANG, B. Fang, X. C. YUN, and H. ZHANG, "A parallel cluster intrusion detection system for backbone network [J]," *Journal of Harbin Institute of Technology*, vol. 3, 2004.

How Can an IDS Find Attacks?

Misuse detection (aka signature-/rule-based)

Searching for what we *know* to be bad.

- Blacklist

Snort Signature Example

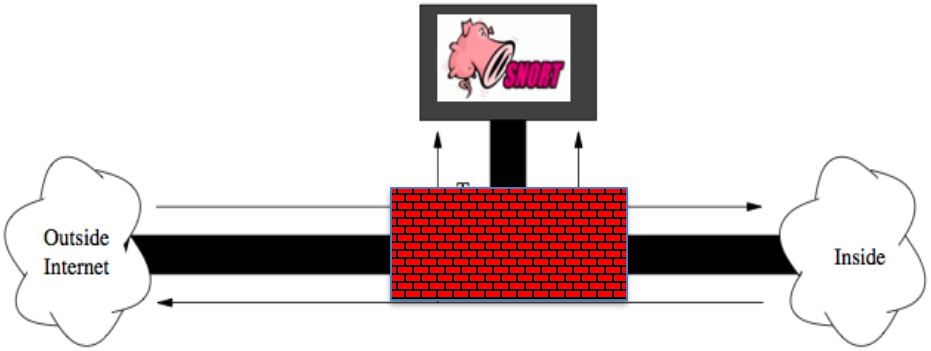
```
alert tcp $EXTERNAL_NET any -> $HOME_NET 139
    flow:to_server,established
    content:"|eb2f 5feb 4a5e 89fb 893e 89f2|"
    msg:"EXPLOIT x86 linux samba overflow"
    reference:bugtraq,1816
    reference:cve,CVE-1999-0811
    classtype:attempted-admin
```

- 普遍部署的成熟模式
- 问题：依赖经验和特征，无法检测未知攻击，加密流量...

Sample *Snort* Signature

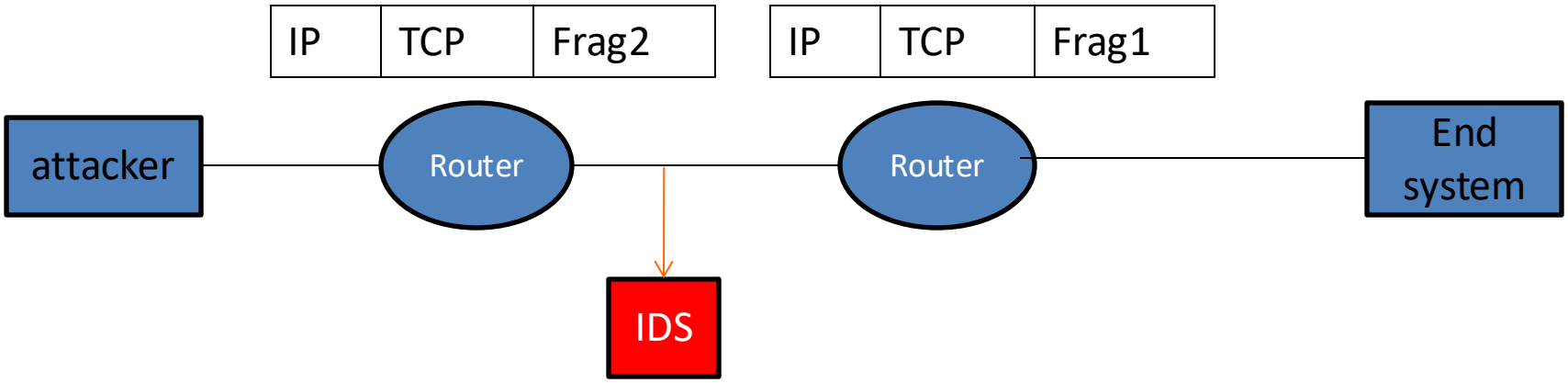
```
alert tcp $EXTERNAL_NET any ->
  $HOME_NET 139
  flow:to_server,established
content:"|eb2f 5feb 4a5e 89fb 893e
  89f2 "
msg:"EXPLOIT x86 linux samba overflow"
reference:bugtraq,1816
reference:cve,CVE-1999-0811
classtype:attempted-admin
```

Challenges of NIDS / Firewall



- Insufficiency of Information on the Wire
  - To predict the behavior of end hosts in real time, according to only packets, but, ambiguities exist in protocols and end hosts 歧义，二义性
- Vulnerability to Denial of Service
  - DoS against end host, to consume its resource
  - DoS against NIDS, to disable it(Fail-Open)

Case Study: IP Fragment



3.2.1.4 Fragmentation and Reassembly: RFC-791 Section 3.2

The Internet model requires that every host support reassembly. See Sections 3.3.2 and 3.3.3 for the requirements on fragmentation and reassembly.

https://tools.ietf.org/html/rfc112

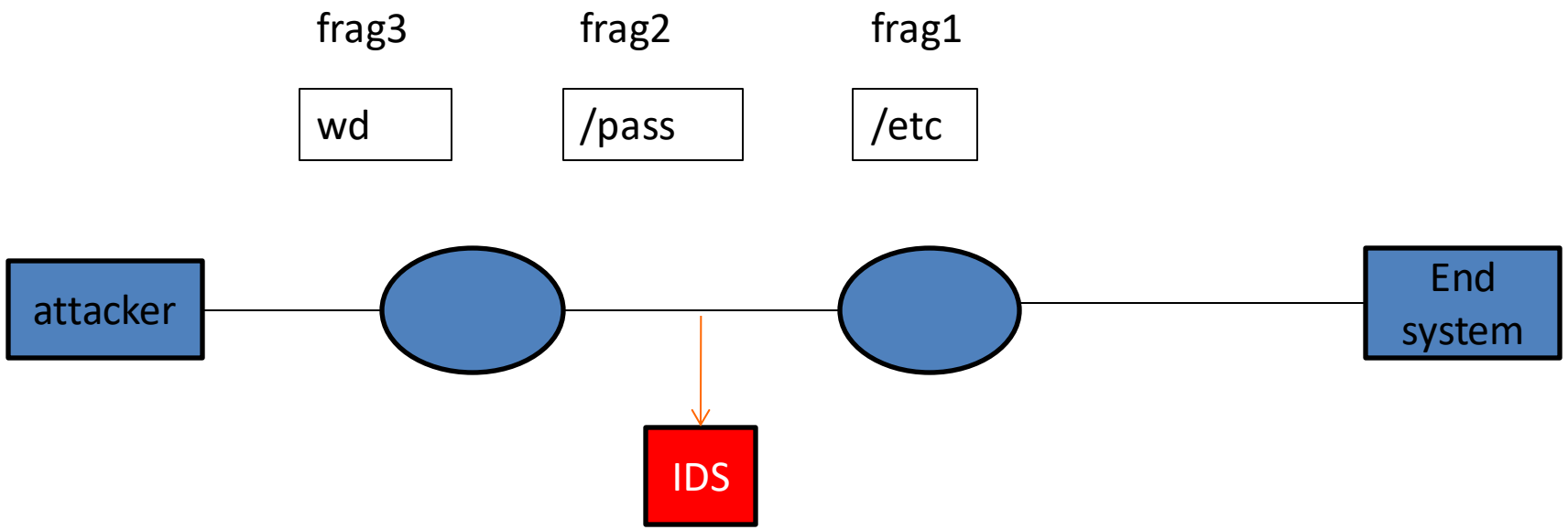
3.3.3 Fragmentation

Optionally, the IP layer MAY implement a mechanism to fragment outgoing datagrams intentionally.

If no assembling…

- snort rules to detect some web shell

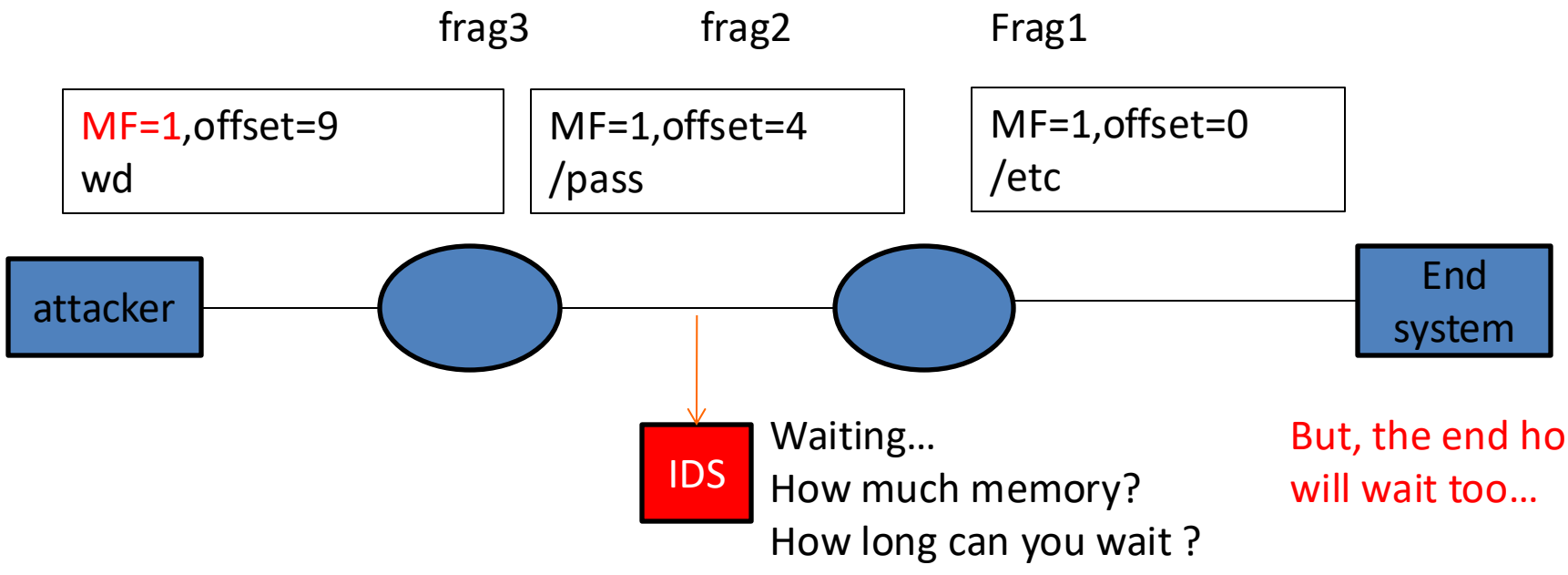
```
alert tcp $ EXTERNAL_NET any -> $ HTTP_SERVERS 80 (msg:
"WEB-MISC / etc / passwd”;flags: A +; content:
”/etc/passwd"; nocase; classtype: attempted-recon; sid:
1122; rev: 1)
```



If assembling…

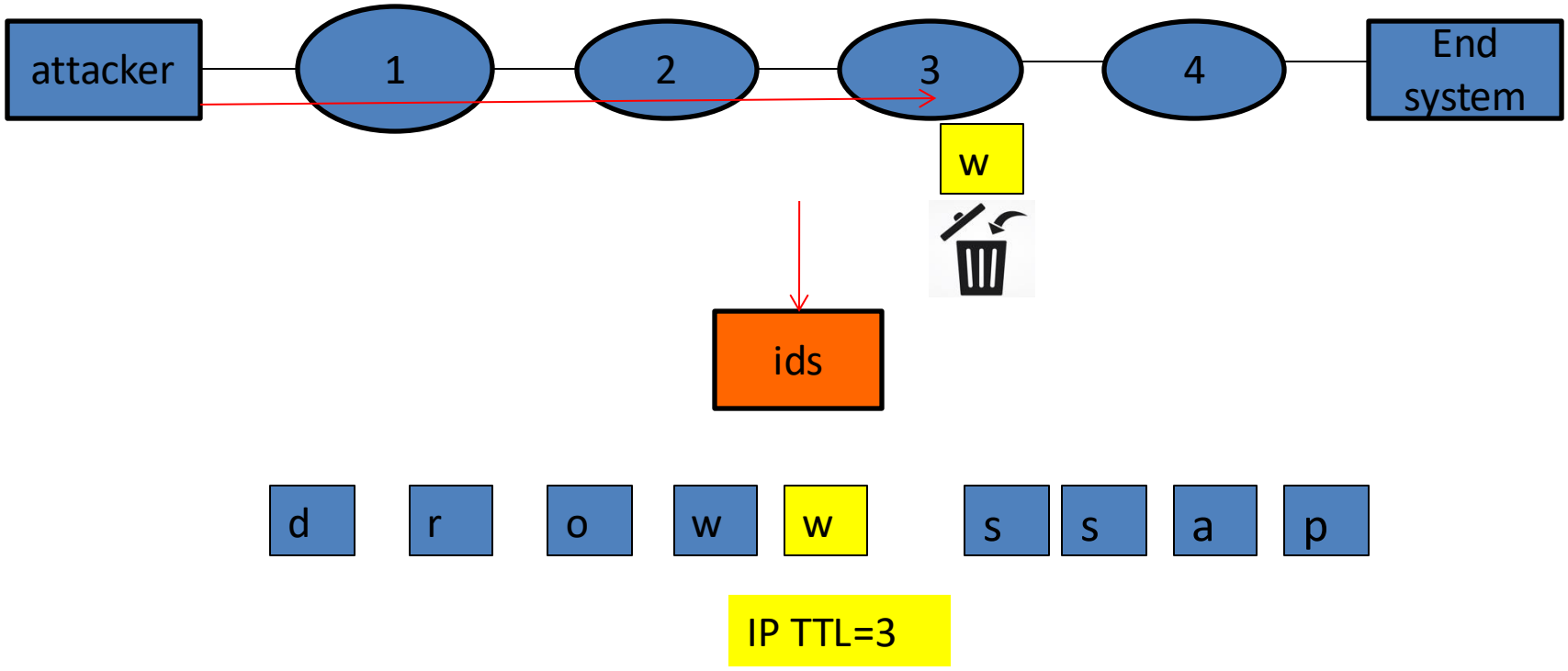
- snort rules to detect some web shell

```
alert tcp $ EXTERNAL_NET any -> $ HTTP_SERVERS 80 (msg:
"WEB-MISC / etc / passwd";flags: A +; content:
"/etc/passwd"; nocase; classtype: attempted-recon; sid:
1122; rev: 1)
```



IP TTL

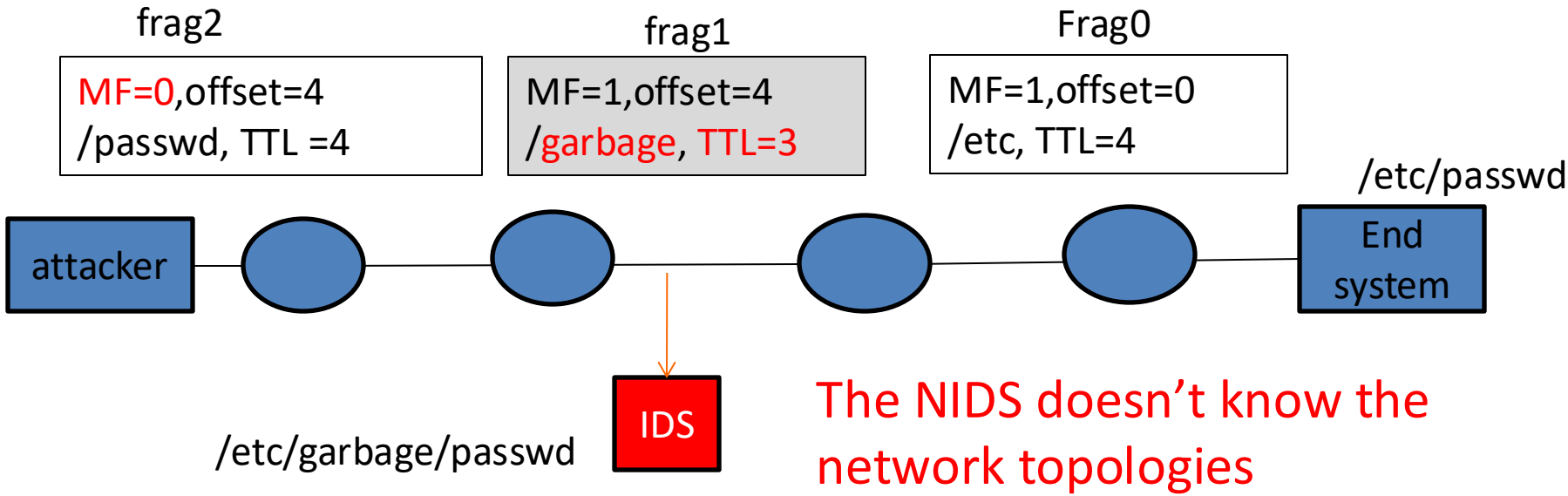
- The length of path is 4
- What if the attacker sends a packet with TTL=3?



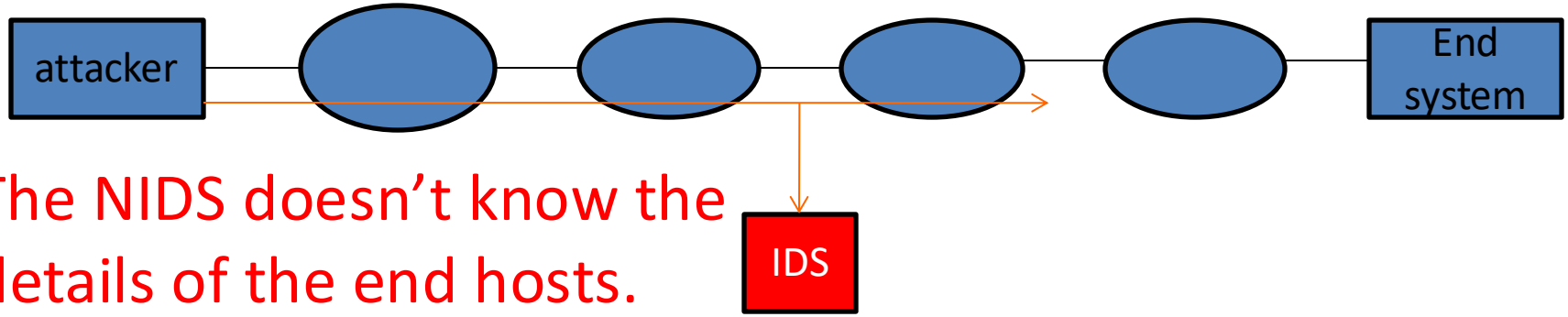
If assembling…

- snort rules to detect some web shell

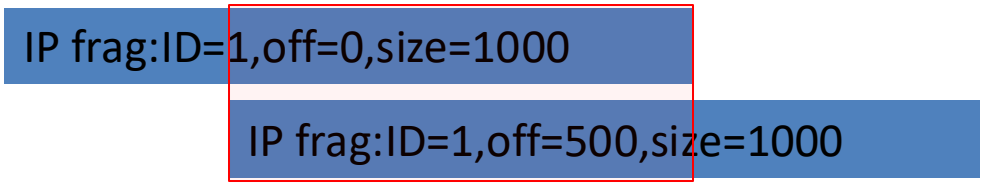
```
alert tcp $ EXTERNAL_NET any -> $ HTTP_SERVERS 80 (msg:
"WEB-MISC / etc / passwd";flags: A +; content:
"/etc/passwd"; nocase; classtype: attempted-recon; sid:
1122; rev: 1)
```



IP fragment Overlap



The NIDS doesn't know the details of the end hosts.

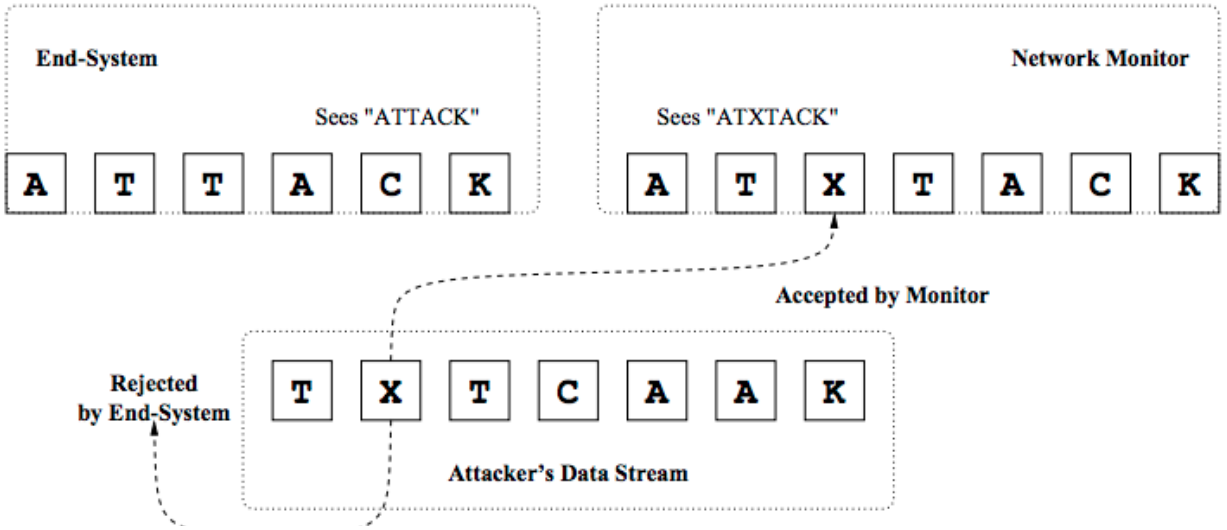


OS	Overlap Behavior
Windows NT 4.0	Favors old data
BSD 4.4	Favors new data
Linux	New..

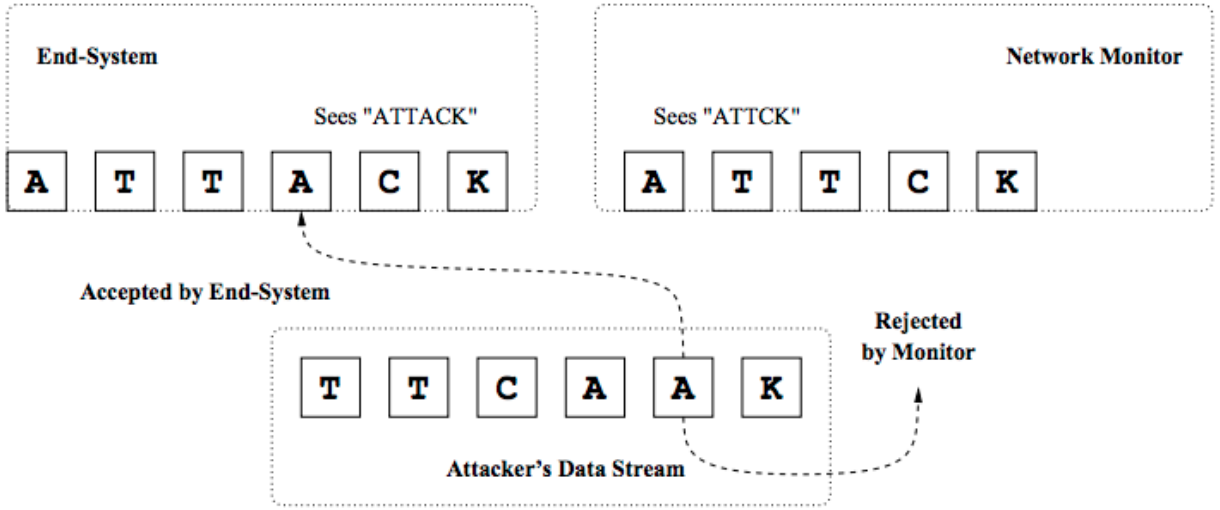
Postel' Law:
"an implementation should be conservative in its sending behavior, and liberal in its receiving behavior"

--RFC 791, 1981

Insertion of ‘X’

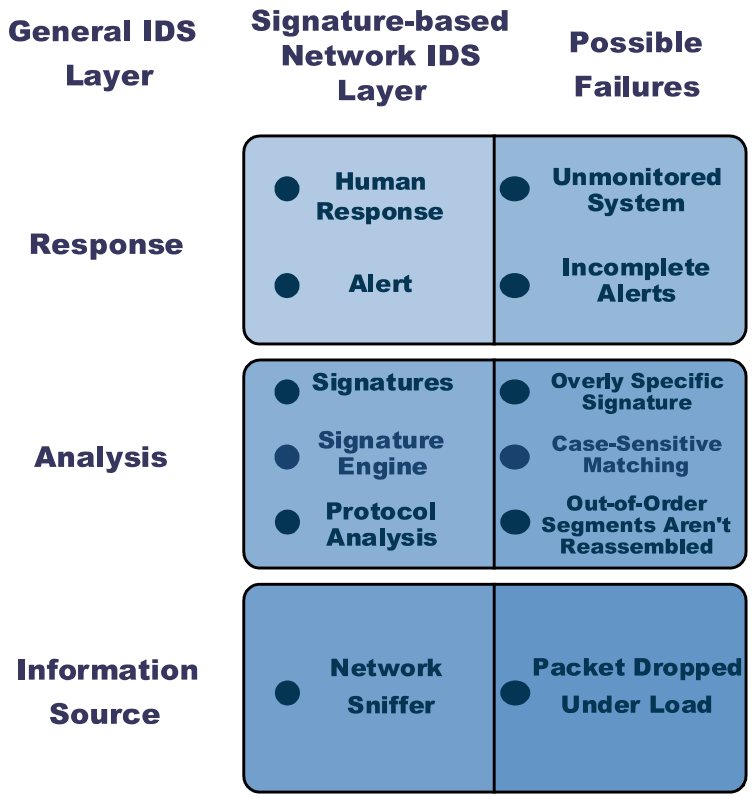


Deletion of ‘A’



Ptacek, T. H., & Newsham, T. N. Insertion, evasion, and denial of service: Eluding network intrusion detection. 1998

NIDS Evade/attack tools



- Evasion Tools
  - Fragroute
  - Whisker
  - Mendax

Fig. 1. Expanded IDS Process Model and Example Failure Cases

A. S. Gorton and T. G. Champion, "Combining Evasion Techniques to Avoid Network Intrusion Detection Systems," 2004.
https://pdfs.semanticscholar.org/0214/553521a7d57eeb08b71838d4f56732612f3c.pdf

Challenges for a NIDS

- Ambiguities in protocol
- Lack of network topology information
- Lack of host information (OS, version, app)

References

1. F. Gont, Security Assessment of the Internet Protocol Version 4,RFC 6274 https://datatracker.ietf.org/doc/rfc6274/
2. S.M. Bellovin*, Security Problems in the TCP/IP Protocol Suite, Computer Communication Review, Vol. 19, No. 2, pp. 32-48, April 1989
3. Steven M. Bellovin, A Look Back at “Security Problems in the TCP/IP Protocol Suite”, 20th Annual Computer Security Applications Conference,2004
4. M. Handley, V. Paxson, and C. Kreibich, “Network intrusion detection: Evasion, traffic normalization, and end-to-end protocol semantics,” *Proceedings of the 10th conference on USENIX Security Symposium-Volume 10*, pp. 9–9, 2001.
5. U Shankar, V Paxson. Active mapping: Resisting NIDS evasion without altering traffic, - Security and Privacy, 2003. Proceedings …, 2003 - ieeexplore.ieee.org

小结

- IP 层网络协议设计问题
  - Fragment 带来的困扰
  - IP ID 导致的侧信道
- NIDS /Firewall设计的困难
  - 协议存在大量歧义
  - 端系统的多样性
  - 从IP Packet 恢复端系统的状态非常困难
  - 网络拓扑结构复杂
- HTTP, Email 等应用层协议面临类似的问题