Protocol T50

"Five months later... So what?"

Agenda

- 0000 – Once upon a time…
- 0001 – Introduction
- 0010 – Improvements
- 0011 – Protocols
- 0100 – Comparison
- 0101 – Demonstration
- 0110 – Conclusions
- 0111 – Questions and Answers

0000 – Once upon a time…

Denial-of-Service

OVER A DECADE!

0001 – Introduction

Why Denial-of-Service?

- Is there anything more offensive than a DoS, anyways?
  - Bear in mind: DoS means “Stress Testing” for this presentation.
- DoS tools are necessary weapons in a cyber warfare…
- Attacks against the infrastructure are more common than many people might think, and, when they happen, people will certainly be aware of.
- But, what are the real damages? What are the real motivations? Image? Revenge? Financial? Political? Hacktivism?
- DoS attacks are significantly harmful, because they violate one of the three key concepts of security that are common to risk management… Which one?
  - Confidentiality
  - Integrity
  - Availability

T50 shows that some sort of **performance enhancements**, using an ordinary Linux box and programming in user space, can be done.

T50 – The chaos maker

- Primarily, the tool was developed to address my day-by-day needs, and I am sharing with the community, because I always need a tool to perform some "Stress Testing" and that could be launched from my notebook:
  - I do not want to carry/rent/buy a Smartbits, Avalanche, etc.

- The tool was designed to perform "Stress Testing" on a variety of infrastructure network devices (Version 2.45).

- The tool was re-designed to extend the "Stress Testing" (Version 5.3), covering some regular protocols (ICMP, IGMP, TCP and UDP), some infrastructure specific protocols (GRE, IPSec and RSVP) and some routing protocols (RIP, EIGRP and OSPF).

- This new version is focused on internal infrastructure, allowing people to test the availability of its resources.

- Interior Gateway Protocols (Distance Vector Algorithm):
  - Routing Information Protocol (RIP).
  - Enhanced Interior Gateway Routing Protocol (EIGRP).

- Interior Gateway Protocols (Link State Algorithm):
  - Open Shortest Path First (OSPF).

- Quality-of-Service Protocols:
  - Resource ReSerVation Protocol (RSVP).

- Tunneling/Encapsulation Protocols:
  - Generic Routing Encapsulation (GRE).

T50 – The chaos maker

I did not review any third-party codes…
I found my own way to address some challenges!!!

```
#define EIGRP_DADDR_LENGTH(foo) \
   (((foo >> 3) & 3) + (foo % 8 ? 1 : 0))
```

```
if(o.eigrp.type == EIGRP_TYPE_SOFTWARE ||
   o.eigrp.type == EIGRP_TYPE_MULTICAST) goto eigrp_software;
```

```
#define EIGRP_DADDR_BUILD(foo, bar)\
   (foo &= htonl(~(0xffffffff >> ((bar >> 3) * 8))))
```

```
#define TCPOLEN_PADDING(foo) \
   ((foo & 3) ? 4 - (foo & 3) : 0)
```

0010 – Improvements

Also known as "New Features"

License

- Licensed under GNU General Public License version 2:
  - Any piece of code cannot be integrated into proprietary applications and appliances.
  - There is an alternative license to do so.

- Free software and 100% Open Source:
  - You may redistribute and/or modify it under the terms of GPL version 2.
  - Will always be available as an Open Source project to the community.

- Recruiting new coders, hackers and developers to keep the project and add new substantial improvements.

Classless Inter-Domain Routing (CIDR)

- CIDR specifies an IP address range using a combination of an IP address and its associated network mask:
  - 192.168.1.13/24 – 192.168.1.13/255.255.255.0
  - 172.16.0.128/15 – 172.16.0.128/255.254.0.0
  - 10.200.200.1/10 – 10.200.200.1/255.192.0.0

- CIDR for destination address is supported:
  - Allows to simulate both Distributed Denial-of-Service and Distributed Reflection Denial-of-Service in a controlled environment.
  - CIDR network mask supported:
    - Minimum is "/8" (255.0.0.0).
    - Maximum is"/30" (255.255.255.252).

Classless Inter-Domain Routing (CIDR)

```
unsigned int hostid      = 0, counter = 0, rand_addr = 0;
in_addr_t    netmask     = INADDR_ANY, all_bits_on = 0xffffffff,
             __1st_addr = INADDR_ANY, addresses[16777214] = INADDR_ANY;
struct iphdr *ip;

    netmask    = ~(all_bits_on >> bits);
    hostid     = (unsigned int) (pow(2, (32 - bits)) - 2);
    __1st_addr = (ntohl(address) & netmask) + 1;

[...]

    for(counter = 0 ; counter < hostid ; counter++)
        addresses[counter] = htonl(__1st_addr++);

[...]

    rand_daddr = (unsigned int) ((float)(hostid) * rand() / (RAND_MAX + 1.0));
    ip->daddr = addresses[rand_daddr];

[...]
```

Multi-protocol sequential injection

- Version 2.45 (as of November 2010):
  - Support for four protocols: ICMP, IGMPv1, TCP and UDP.
  - Sends all of them sequentially, i.e., almost on the same time.

- Version 5.3 (as of today):
  - Support for the previous four protocols: ICMP, IGMPv1[1], TCP[1] and UDP.
  - Eleven (11) new protocols: IGMPv3[1], EGP[2], RIPv1, RIPv2, DCCP[1], RSVP[1], GRE[3], IPSec (AH/ESP), EIGRP[1] and OSPF[1].
  - Sends all of them sequentially, i.e., almost on the same time.

1 This protocol can be improved to cover additional advanced options.
2 This protocol demands more development efforts to cover advanced options.
3 Very first tool able to encapsulate the protocols within GRE packets.

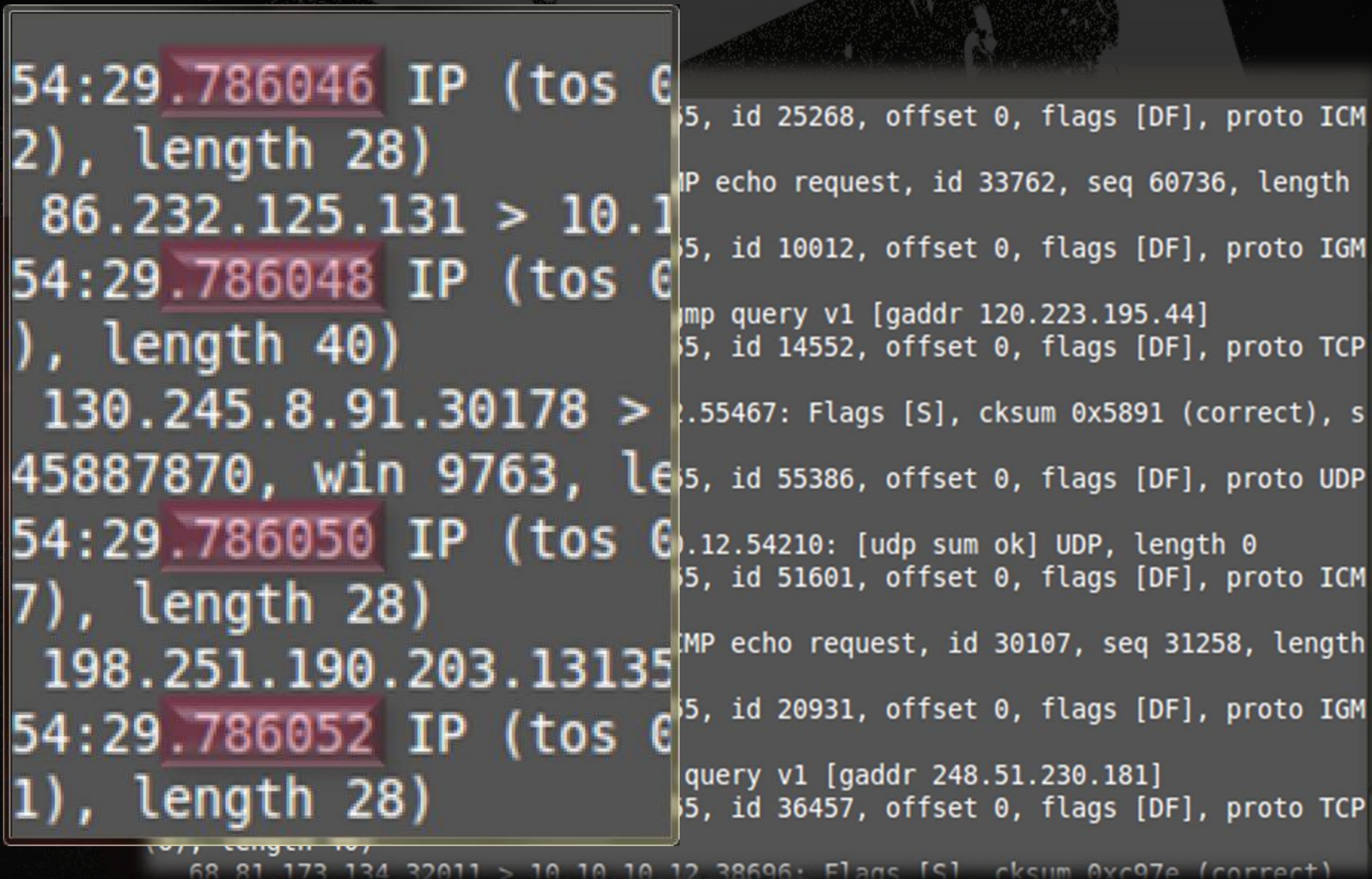
Multi-protocol sequential injection

```
socket_t fd; int flags, n = 1, len, * nptr = &n; fd_set wfds;

[...]

    if((fd = socket(AF_INET, SOCK_RAW, IPPROTO_RAW)) == -1)
        exit(EXIT_FAILURE);

    if(setsockopt(fd, IPPROTO_IP, IP_HDRINCL, nptr, sizeof(n)) < 0)
        exit(EXIT_FAILURE);

[...]
```

Multi-protocol sequential injection

```
struct t50{ int proto; void(*raw)(int, struct options); };

[...]

    while(flood || threshold--){

[...]

        if(protocol != IPPROTO_T50){

[...]

        }else{
            for(module = 0 ; module < modules ; module++){
                protocol = t50[module].proto;
                t50[module].raw(fd, options);
            }

            threshold -= (modules-1);
            protocol = IPPROTO_T50;
        }

[...]
```

Multi-protocol sequential injection



Checksum optimization

- The version 5.3 introduced a new technique to calculate the checksum, consequentially, a new technique to build the packet.

- This technique is **`MEMCPY(3)`**-free, and allows to build the packet byte-by-byte – sometimes bit-by-bit.

- This technique is more flexible, specially when playing with exotic protocol options – sometimes uses GOTO. For example:
  - EIGRP IP Internal Routes TLV destination address.
  - EIGRP IP External Routes TLV destination address.
  - OSPF HELLO Message with multiple NEIGHBOR addresses.
  - RSVP Object SCOPE Class with multiple SCOPE addresses.
  - Etc…

Checksum optimization

```
unsigned int  offset = 0;
unsigned char packet[packet_size], * checksum = NULL;
struct eigrp_hdr * eigrp;

[...]

    offset   = sizeof(struct eigrp_hdr);
    checksum = (unsigned char *)eigrp + offset;

    *((unsigned short *)checksum) = htons(length);
    checksum += sizeof(unsigned short);
    offset   += sizeof(usigned short);
    *((usigned int *)checksum) = htonl(auth_key_id));
    checksum += sizeof(unsigned int);
    offset   += sizeof(usigned int);

[...]

    eigrp->check = cksum((u_int16_t *)eigrp, offset);

[...]
```

Checksum optimization

```
packet[packet_size]
```

Checksum optimization

```
packet_size = sizeof(ip) + sizeof(eigrp) + eigrp_hdr_len();
```

```
packet[packet_size]
```

Checksum optimization

```
packet[packet_size]
```

Checksum optimization

packet[packet_size]

packet 1st step → ip

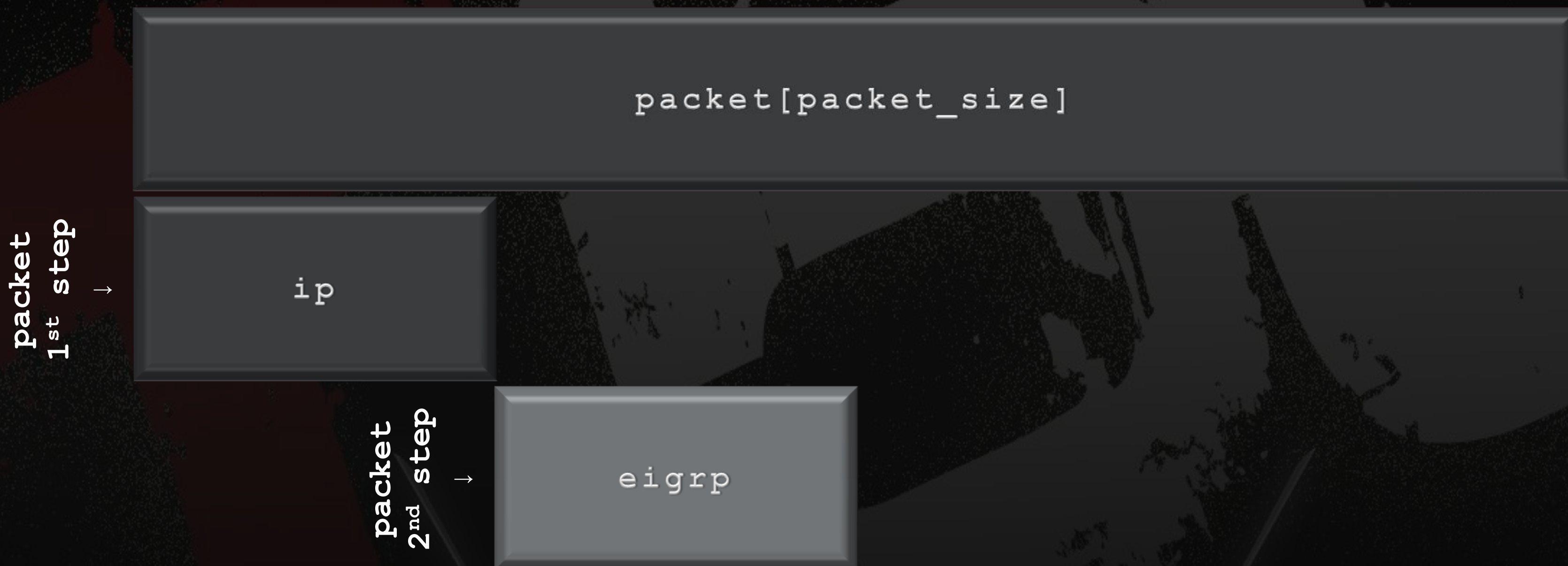
Checksum optimization

`packet[packet_size]`

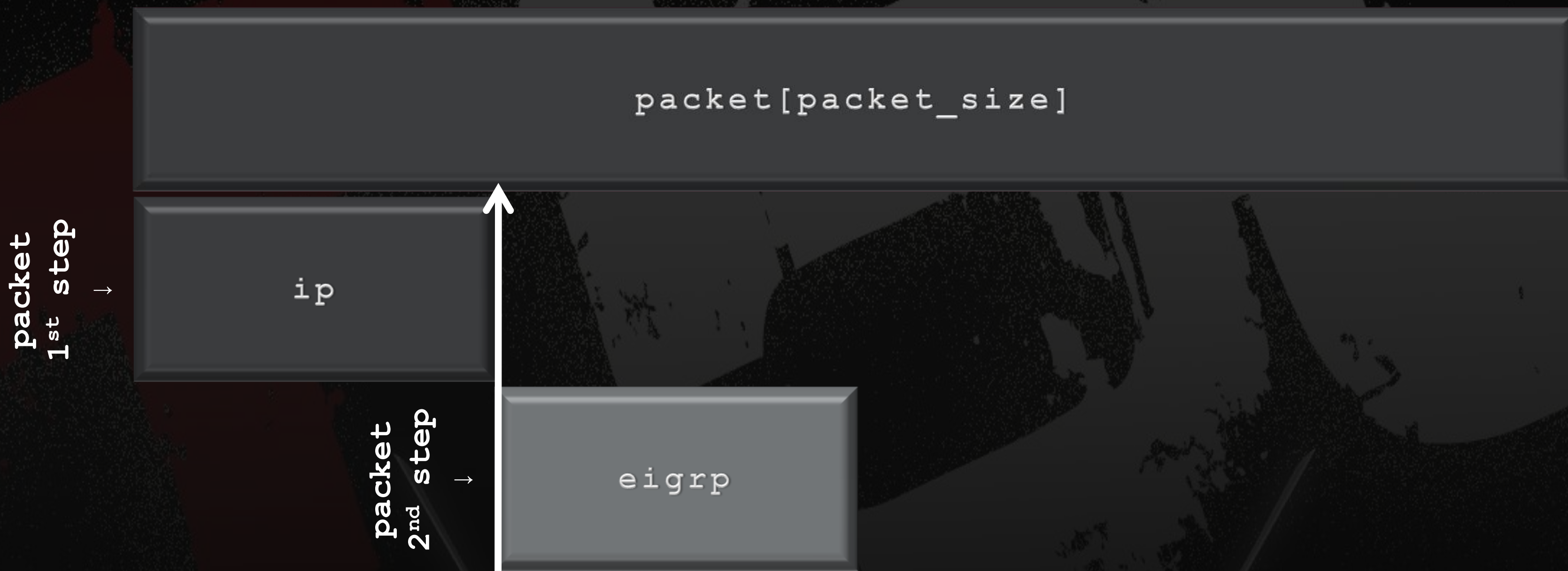
packet
1^{st} step →

`ip`

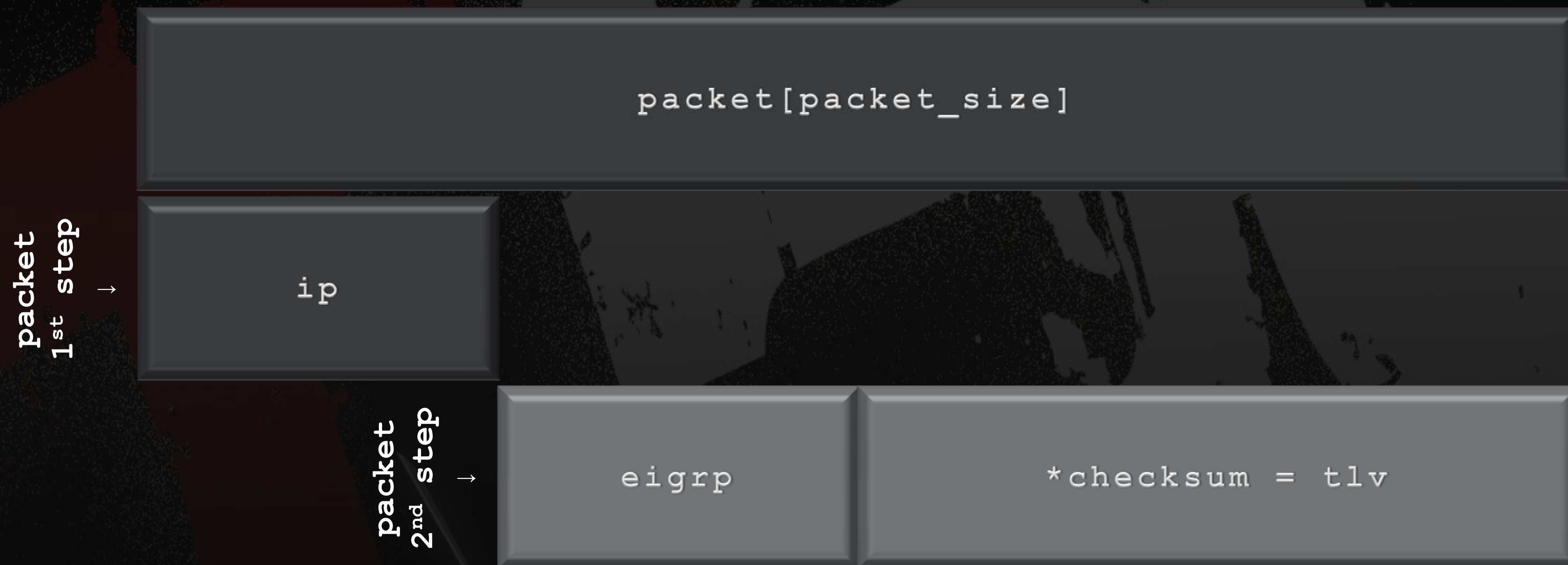
Checksum optimization



Checksum optimization



Checksum optimization



Checksum optimization

packet[packet_size]

packet 1st step → ip

packet 2nd step → eigrp | *checksum = tlv

Checksum optimization

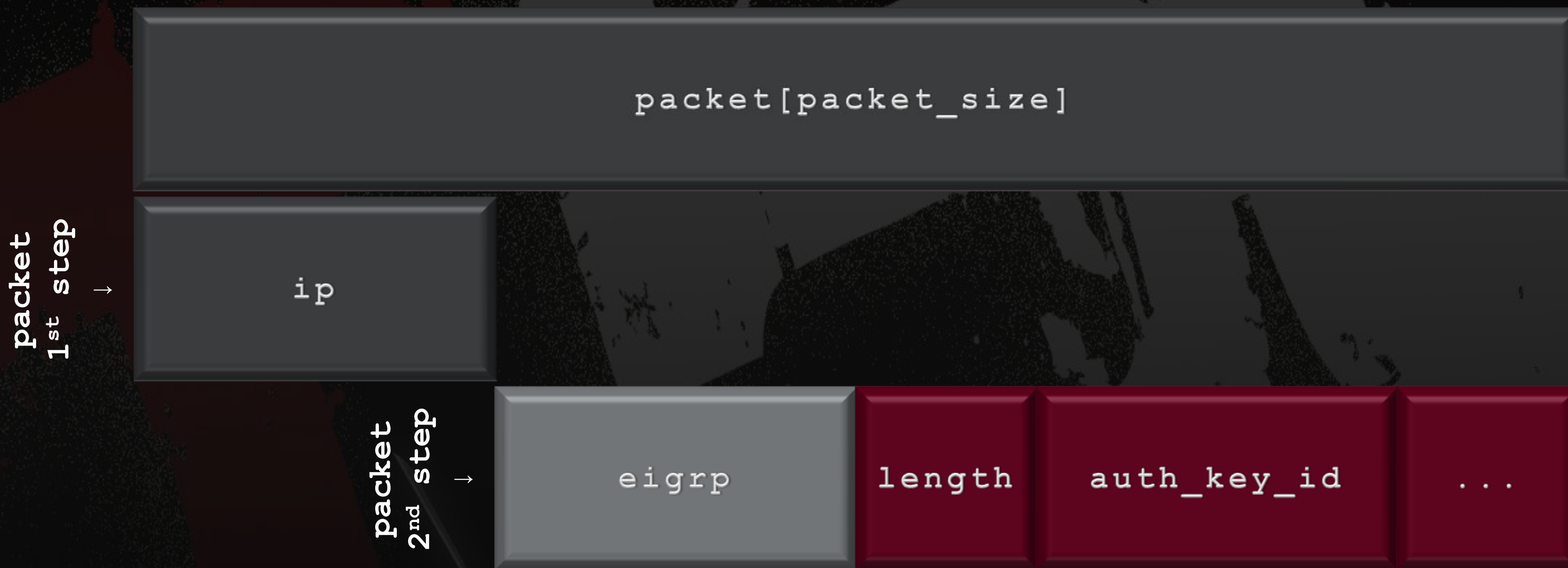
Checksum optimization

packet[packet_size]

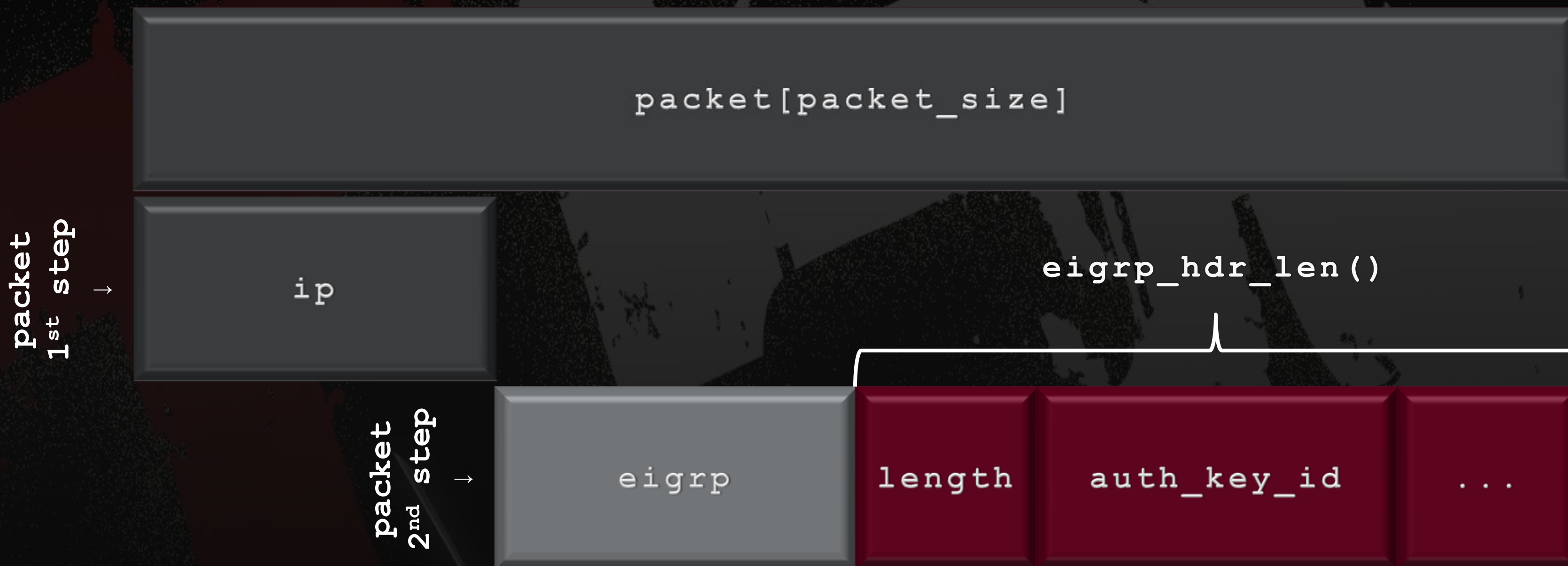
packet 1st step → ip

packet 2nd step → eigrp | length | auth_key_id |

Checksum optimization



Checksum optimization



Checksum optimization

Checksum optimization

packet[packet_size]

packet 1st step → ip

packet 2nd step → eigrp | length | auth_key_id | ...

eigrp->check = cksum(eigrp, offset);

Checksum optimization

```
sendto(fd, &packet, packet_size, 0, &sin, sizeof(sin))
```

RFC 1700, 1918 and 3330 improvements

```
[...]

switch(ntohl(daddr) & 0xff000000){
    case 0x0a000000:        /* Allowing 10/8        (RFC 1918). */
        break;
    case 0x7f000000:        /* Allowing 127/8       (RFC 1700). */
        break;
    case 0xa9000000:        /* Allowing 169.254/16 (RFC 3330). */
        if(((ntohl(daddr) & 0xffff0000) != 0xa9fe0000))
            return(FALSE);
        break;
    case 0xac000000:        /* Allowing 172.16/12  (RFC 1918). */
        if(((ntohl(daddr) & 0xffff0000) < 0xac100000) || \
          ((ntohl(daddr) & 0xffff0000) > 0xac1f0000))
            return(FALSE);
        break;
    case 0xc0000000:        /* Allowing 192.168/16 (RFC 1918). */
        if((ntohl(daddr) & 0xffff0000) != 0xc0a80000)
            return(FALSE);
        break;

[...]
```

0011 – Protocols

Protocols

IGMPv3

- Specific headers for specific types:
  - Membership Query.
  - Membership Report.
- Membership Query options:
  - Max Resp code.
  - Group Address.
  - Suppress Router-processing Flag.
  - Querier's Robustness Variable (QRV).
  - Querier's Query Interval Code (QQIC).
  - Number of Sources.
  - Source Address(es).
- Membership Report options:
  - Group Record Type.
  - Group Record Multicast Address.
  - Number of Sources.
  - Source Address(es).

TCP

- Regular TCP options:
  - Source Port and Destination Port, Sequence Number (also known as ISN), Acknowledgment Number, Data Offset, Window, Urgent Pointer and TCP Flags (FIN, SYN, RST, PSH, ACK, URG, ECE and CWR).
- Supported TCP Options:
  - End of List (EOL), No Operation (NOP), Maximum Segment Size (MSS), Windows Scale (WSopt), Timestamp (TSopt), T/TCP Connection Count (CC, CC.NEW and CC.ECHO), Selective Acknowledgement (SACK), MD5 Signature Option and the brand new TCP-AO (Authentication Option – RFC 5925).
- TCP Authentication Option (as of June 2010):
  - Type (HMAC-MD5).
  - Key ID.
  - Next Key ID.
  - Authentication Data (RANDOM).

Protocols

RIP

- Regular RIPv1 and RIPv2 options:
  - Command.
  - Address Family Identifier.
  - Router IP Address.
  - Router Metric.
- Enhanced RIPv2 options:
  - Routing Domain.
  - Route Tag.
  - Router Network Mask.
  - Router Next Hop.
- RIPv2 Cryptographic Authentication:
  - Type (HMAC-MD5).
  - Key ID.
  - Cryptographic Sequence Number.
  - Authentication Data (RANDOM).

DCCP

- Specific headers for specific types:
  - Request Packets
  - Response Packet.
  - Data Packets
  - Acknowledgment Packet, Data-Ack Packet, Synchronize Packet, Sync-Ack Packet, Close Packet and Close Request Packet.
  - Reset Packet.
- Regular DCCP options:
  - Source Port and Destination Port.
  - Data Offset.
  - HC-Sender CCID (CCVal).
  - Checksum Coverage (CsCov).
  - Extended Sequence Numbers (x).
  - Sequence Numbers (HIGH and LOW).
  - Acknowledgment Numbers (HIGH and LOW).
  - Service Code.
  - Reset Code.

Exotic protocols

RSVP

- Supported RSVP types:
  - Path Message.
  - Resv Message.
  - Path Teardown Message.
  - Resv Teardown Message.
  - Path Error Message.
  - Resv Error Messages
  - Confirmation Message.

- Specific RSVP Objects for specific RSVP type:
  - SESSION Class.
  - RSVP_HOP Class.
  - TIME_VALUES Class.
  - ERROR_SPEC Class.
  - SCOPE Class.
  - STYLE Class.
  - SENDER_TEMPLATE Class.
  - SENDER_TSPEC Class.
  - ADSPEC Class.
  - RESV_CONFIRM Class.

- Regular RSVP options:
  - Flags and Time to Live.

- SESSION Class options:
  - Destination address, Protocol ID, Flags and Destination Port.

- RSVP_HOP Class options:
  - IP Next/Previous Hop (Neighbor) Address and Logical Interface Handle.

- TIME_VALUES Class options:
  - Refresh Period (Interval).

- ERROR_SPEC Class options:
  - IP Error Node Address, Flags, Error Code and Error Value.

- SCOPE Class options:
  - Number of Address and IP Source Address(es).

- Etc... Up to 37 command line interface switches.

Exotic protocols

EIGRP

- Supported EIGRP opcodes:
  - Update Message.
  - Request Message.
  - Query Message.
  - Reply Message.
  - Hello Message.
  - Acknowledgment Message.

- Specific EIGRP TLVs for specific EIGRP types:
  - General Parameter TLV.
  - Software Version TLV.
  - Sequence TLV.
  - Next Multicast Sequence TLV.
  - IP Internal Routes TLV.
  - IP External Routes TLV.

- EIGRP Cryptographic Authentication:
  - Type (HMAC-MD5).
  - Key-ID.
  - Authentication Data (RANDOM).

- Regular EIGRP options:
  - Opcode, Flags, Sequence Number, Acknowledgment Number, Autonomous System (AS) , Type and Length.

- General Parameter TLV options:
  - K1, K2, K3, K4 and K5 Values and Hold Time (Interval).

- Software Version TLV options:
  - IOS Release Version and EIGRP Protocol Release Version.

- IP Internal Routes TLV and IP External Routes TLV options:
  - IP Next Hop Address, Delay, Bandwidth, Maximum Transmission Unit (MTU), Hop Count, Load, Reliability, IP Source Address(es) and IP Address Prefix (CIDR).

- Etc... Up to 33 command line interface switches.

Exotic protocols

OSPF

- Supported OSPF type:
  - Hello Packet.
  - Database Description Packet.
  - Query Message Packet.
  - Link State Request Packet.
  - Link State Update Packet.
  - Link State Acknowledgment Packet.

- Specific LSA Header for specific LSA type:
  - Router LSA Header.
  - Network LSA Header.
  - Summary IP Network LSA Header.
  - Summary ASBR Header.
  - AS External LSA Header. (ASBR).
  - No-so-Stubby Area LSA Header (NSSA).
  - Group Membership LSA Header (Multicast).

- OSPF Cryptographic Authentication:
  - Type (HMAC-MD5).
  - Key ID.
  - Cryptographic Sequence Number.
  - Authentication Data (RANDOM).

- Specific LLS Data Block for specific LLS TLV:
  - Extended Options and Flags TLV.
  - Cryptographic Authentication TLV.

- Regular OSPF options:
  - Type, Router ID, Area ID and Options (Multi-Topology or TOS-Based, External Routing Capability, Multicast Capable, NSSA Supported, LLS Data Block in Contained, Demand Circuits is Supported, Opaque-LSA and Down Bit).

- Etc… Up to 54 command line interface switches.

0100 – Comparison

Methodology

TOOL	ROUND	5 sec	10 sec	15 sec	20 sec	25 sec	30 sec	3[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
T50++	1	1,042,520 pps	1,005,000 pps	1,096,896 pps	1,007,505 pps	884,625 pps	1,127,360 pps	966,[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
	2	1,049,119 pps	990,140 pps	1,017,841 pps	1,015,022 pps	929,380 pps	1,066,516 pps	975,[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
	3	1,006,763 pps	990,749 pps	1,024,613 pps	1,108,781 pps	977,011 pps	935,315 pps	970,[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
	Average	1,032,801 pps	995,296 pps	1,046,450 pps	1,043,769 pps	930,339 pps	1,043,064 pps	970,[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
T50	1	725,418 pps	764,580 pps	737,099 pps	754,883 pps	726,450 pps	775,007 pps	730,[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
	2	717,414 pps	706,533 pps	682,686 pps	773,280 pps	744,215 pps	771,824 pps	693,[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
	3	758,627 pps	750,388 pps	737,385 pps	712,530 pps	673,897 pps	763,482 pps	755,[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
	Average	733,820 pps	740,500 pps	719,057 pps	746,898 pps	714,854 pps	770,104 pps	726,[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
GEMINID	1	687,767 pps	726,631 pps	706,528 pps	664,497 pps	648,750 pps	653,156 pps	700,[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
	2	683,211 pps	671,667 pps	636,244 pps	659,598 pps	692,827 pps	628,294 pps	667,[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
	3	669,697 pps	615,774 pps	633,065 pps	665,430 pps	689,454 pps	654,414 pps	681,[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
	Average	680,225 pps	671,357 pps	658,612 pps	663,175 pps	677,010 pps	645,288 pps	683,[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
C4	1	642,031 pps	610,288 pps	640,826 pps	664,040 pps	651,443 pps	636,698 pps	637,[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
	2	679,068 pps	646,650 pps	603,662 pps	678,486 pps	734,530 pps	633,777 pps	668,[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
	3	680,033 pps	638,640 pps	673,919 pps	717,858 pps	629,936 pps	628,849 pps	677,[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
	Average	667,044 pps	631,859 pps	639,469 pps	686,795 pps	671,970 pps	633,108 pps	660,[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
MAUSEZAHN	1	428,713 pps	429,454 pps	451,012 pps	417,094 pps	492,839 pps	395,440 pps	432,[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
	2	381,298 pps	385,295 pps	383,023 pps	381,767 pps	370,599 pps	352,501 pps	362,[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
	3	393,947 pps	454,822 pps	440,594 pps	458,563 pps	439,603 pps	430,944 pps	485,573 pps	415,073 pps	316,867 pps	428,321 pps	426,430.70 pps
	Average	401,319 pps	423,190 pps	424,876 pps	419,141 pps	434,347 pps	392,962 pps	426,861 pps	444,562 pps	389,192 pps	439,198 pps	419,564.93 pps
[L]OTUS	1	235,631 pps	255,594 pps	249,036 pps	237,238 pps	254,054 pps	249,670 pps	249,588 pps	243,094 pps	264,290 pps	237,910 pps	247,610.50 pps
	2	244,804 pps	253,116 pps	247,638 pps	258,888 pps	232,283 pps	248,570 pps	235,393 pps	250,298 pps	258,258 pps	217,327 pps	244,657.50 pps
	3	248,238 pps	261,346 pps	228,825 pps	246,062 pps	249,342 pps	231,604 pps	247,523 pps	230,368 pps	233,460 pps	247,719 pps	242,448.70 pps
	Average	242,891 pps	256,685 pps	241,833 pps	247,396 pps	245,226 pps	243,281 pps	244,168 pps	241,253 pps	252,003 pps	234,319 pps	244,905.57 pps
F22	1	245,173 pps	260,785 pps	248,110 pps	264,463 pps	250,081 pps	250,203 pps	261,815 pps	248,948 pps	244,840 pps	227,263 pps	250,168.10 pps
	2	249,326 pps	244,757 pps	216,906 pps	261,288 pps	250,084 pps	247,186 pps	232,441 pps	235,442 pps	237,577 pps	233,648 pps	240,865.50 pps
	3	239,308 pps	236,760 pps	243,571 pps	231,162 pps	223,074 pps	240,039 pps	234,427 pps	245,488 pps	244,061 pps	234,593 pps	237,248.30 pps
	Average	244,602 pps	247,434 pps	236,196 pps	252,304 pps	241,080 pps	245,809 pps	242,894 pps	243,293 pps	242,159 pps	231,835 pps	242,760.63 pps
HPING3	1	156,493 pps	151,085 pps	146,048 pps	145,270 pps	147,417 pps	152,256 pps	145,725 pps	144,520 pps	176,051 pps	136,306 pps	150,117.10 pps
	2	153,111 pps	150,119 pps	147,199 pps	172,350 pps	142,788 pps	148,529 pps	148,544 pps	149,045 pps	138,734 pps	139,219 pps	148,963.80 pps
	3	146,603 pps	146,211 pps	143,150 pps	143,566 pps	154,061 pps	147,915 pps	145,588 pps	150,853 pps	167,876 pps	155,560 pps	150,138.30 pps
	Average	152,069 pps	149,138 pps	145,466 pps	153,729 pps	148,089 pps	149,567 pps	146,619 pps	148,139 pps	160,887 pps	143,695 pps	149,739.73 pps

919,062 pps	1,000,337.00 pps
961,701 pps	1,003,714.43 pps
735,146 pps	732,918.30 pps
726,174 pps	727,705.20 pps
799,222 pps	746,154.10 pps
753,514 pps	735,592.53 pps
731,240 pps	688,111.90 pps
733,406 pps	671,461.40 pps
671,188 pps	658,195.40 pps
711,945 pps	672,589.57 pps
654,753 pps	647,260.30 pps

0101 – Demonstration

T50: an Experimental Mixed Packet Injector

Dell Latitude E6400

- Intel® Core™ 2 Duo P8400 (2.26 GHz)
- Memory 4GB RAM
- Ubuntu Desktop Linux 10.04 64-bit
- Intel® 82567LM Gigabit Controller
- 1 Gbps Network
- Cross-over Cable (CAT-5e)

Dell Latitude D620

- Intel® Core™ Duo T5600 (1.83 GHz)
- Memory 2GB RAM
- Microsoft Windows 7 32-bit
- Broadcom NetXtreme 57xx Gigabit Controller
- 1 Gbps Network
- Cross-over Cable (CAT-5e)

0110 – Conclusions

Conclusions

- Can be applied to any DoS:
  - Peer-to-Peer Attacks
  - Application Level Attacks
  - Distributed Attacks
  - Reflected Attacks
  - Level-2 Attacks
  - Degradation-of-Service Attacks
  - DNS Amplifiers Attacks
- Is DoS and DDoS so 1990's?
  - Please, don't be silly, again!!!
- Can be considered a cyber warfare's weapon?
  - Yes, it can be considered like one.
- It is just a matter of time to things get worse on the Internet.
- A DoS can be perpetrated overnight!
- What else?

An attacker does not even need multiples zombies.

0111 – Questions & Answers

Any questions?

THANK YOU!