A DoS Attack Against the Integrity-Less ESP (IPSec)

Ventzislav Nikov

Philips TASS and AppTech,
Leuven, Belgium
ventzislav.nikov@philips.com

Abstract. This paper describes a new practical DoS attack that can be mounted against the "encryption-only" configuration (i.e. without authenticated integrity) of ESP as allowed by IPSec.
This finding can serve as a strong argument to convince those in charge of the IPSec standardization to improve it by banning the "encryption-only" configuration from the standard.

Keywords: IPSec (ESP) Standard, Denial of Service Attack.

1 Introduction

As illustrated in [3, 4, 6, 7, 15, 28], it is, today, common knowledge in the cryptographic community that encryption without authenticated integrity opens the door to various attacks. More recently, the authors of [22] illustrate this argument on the encryption-only configuration allowed by IPSec for the secure payload encapsulation (ESP) [11].

IPSec encryption-only configuration was shown to be exposed to variety of "initialization vector attacks" [15]. The authors of [15] analyze the security risks posed on encryption-only IPSec when it is used as intermediate layer of the protocol stack. But their analysis has a limited scope - just the implication on the next (possible) upper-layer protocol (e.g. TCP, UDP, IP). We develop further their idea of properly modifying certain fields in the payloads of the whole protocol stack (e.g. IP/UDP/RTP or IP/TCP). In this way we build a more powerful attack.

The authors of [22] have exposed even more serious weakness of the encryption-only configuration on IPSec. It has been shown that the "encryption-only" configuration is subject to a variety of ciphertext-only attacks, i.e. revealing the encryption key. The attack consists of two phases: in the first phase the attacker modifies certain fields in the first few blocks of the ciphertext such that the upper-layer protocol submits either an error message or the decryption of the modified datagram directly to the attacker's machine. The second phase proceeds with further recovery of the plaintext. The attacks presented in [22] are efficient and have been proven practical by the authors, i.e. against an implementation of the IPSec stack in the Linux kernel. Our attack is completely different, since it is not a chipertext-only attack, the attacker goal is to mount with minimum efforts a Denial of Service Attack (DoS) against the IPSec gateway.

One of the purposes of [22] was to help closing the gap between the currently known cryptography theory and the current practices of standard bodies, implementers and users. Among others, providing practical efficient attacks against existing (or in development) standards or products helps the various communities (theoreticians and practicians) to work together to establish cryptographically sound but workable standards and products.

The main contribution of this paper is to present yet another attack against the "encryption-only" configuration for ESP. This attack has been unearthed years ago but hasn't lead to a publication because at that time it was perceived as of no theoretical interest. However, following the example of [22], we hope to contribute with the publication of our finding to help improving the IPSec standard (and the corresponding implementations).

Our attack is a very practical Denial of Service (DoS) attack against the IPSec gateway providers. It only requires a minimal effort from the attacker while having a devastating potential impact on the gateways performance. We think that this attack is an even stronger and convincing argument than those already published to support the banishment of the "encryption-only" configuration of ESP.

Recall that the ESP configuration is not integrity protected, thus the option provided in the ESP for replay protection makes no sense since it requires obviously data integrity. We stress that our attack is not

just a reply attack since the attacker can deviate/multiply the flow to different targets. We also assume, as in [15, 22], that the datagrams are not checked after IPSec processing to see if the correct IPSec policies were applied; this is the case in the Linux kernel implementation (as pointed out in [22]).

Although it is commonly known that encryption without authenticated integrity is dangerous and more specifically that ESP without data authentication leads to all kind of problems there are still security IPSec specialists that claim "when IPSec gateway is protected in addition with a stateful firewall which does filter the packets, most of the described above attacks will not work." We think that such an approach is just an attempt to shift the problem, instead of eradicate it.

The rest of this paper is organized as follows: After an introduction to the IPSec ESP in section 2, we present the DoS attack itself in section 3.

2 Preliminaries

2.1 IPSec

IPSec provides security at the level of the IP layer. The secure encapsulation of the payload of IP datagrams is part of the IPSec standard and is provided by ESP (Encapsulating Security Payload, [11]). ESP provides integrity, authentication and encryption of the IP datagrams (as an option, it is possible to add a replay protection).

The encapsulation is connectionless, i.e. it is performed on a per-datagram basis. One common use of IPSec is to build virtual private networks where IPSec is configured to use ESP in tunnel mode. The way ESP modifies the datagrams in tunnel mode is shown in Fig. 1.

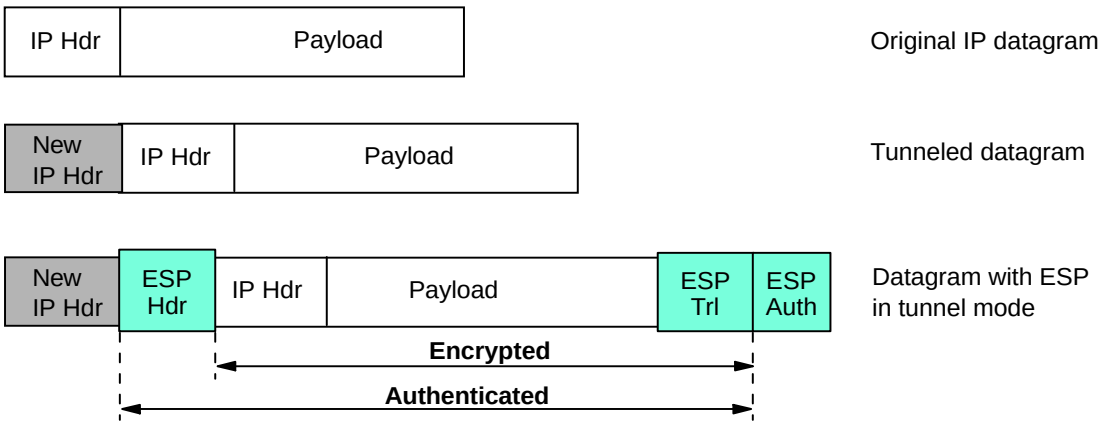

Fig. 1. ESP in Tunnel Mode

Fig. 1 shows that given a datagram to protect, ESP creates a new datagram made of, in sequence, a new IP header, an ESP header, the original datagram (IP header and Inner Payload), an ESP trailer and an ESP authentication payload. The detailed description of the ESP header, trailer and authentication payload is not provided since they are outside of the scope of our attack.

IPSec supports different encryption algorithms: AES [8, 19] is the most commonly used but other block ciphers as DES [16], 3DES [17], CAST128 [1, 2], RC5 [24], IDEA [12], and Blowfish [26] are also allowed. All block ciphers are used in CBC mode [23, 9]. Additionally, AES in CTR mode is a valid alternative [10] but in this case the standard states that "*AES-CTR implementations MUST employ a non-NULL ESP authentication method, since it is trivial to forge an AES-CTR ciphertext*". As a consequence, the hereafter described attack targets only the block ciphers in CBC mode. The attack depends only on the block size n of the cipher, where $n = 64$ for all block ciphers except for AES where $n = 128, 192, 256$.

2.2 CBC mode

If the to be transmitted data after ESP padding is made of q blocks, if the plaintext blocks are denoted by $P_1, P_2, \ldots, P_q$, and if $e_K(\cdot)$ ($d_K(\cdot)$) denotes the encryption (decryption) of blocks using an n-bit key K then the CBC mode [18] works as follows. After a random n-bit IV (Initialization Vector) is generated, the ciphertext blocks are computed according to the equations:

$$C_0 = IV, \quad C_i = e_K(C_{i-1} \oplus P_i), \quad (1 \leq i \leq q).$$

At the receiver side the plaintext is recovered according to the equations:

$$P_i = C_{i-1} \oplus d_K(C_i), \quad (1 \leq i \leq q).$$

As pointed out in [14, 22] a well known weakness of the CBC mode is the "*bit flipping attack*". An attacker can flip (invert) a specific bit in the ciphertext block C_{i-1}, then this specific bit in the recovered plaintext block P_i is also flipped (since $P_i = C_{i-1} \oplus d_K(C_i)$). This allows an attacker to introduce controlled changes into the recovered plaintext block P_i, but the previous block P_{i-1} is randomized. Hence the integrity of the IV in CBC should be protected otherwise uncontrolled change on the first recovered plaintext block P_1 is possible.

2.3 IP, TCP, UDP and RTP Datagrams

The hereafter presented attack depends on the way the IP stack is structured. The presentation is limited to IPv4 headers as specified in [27].

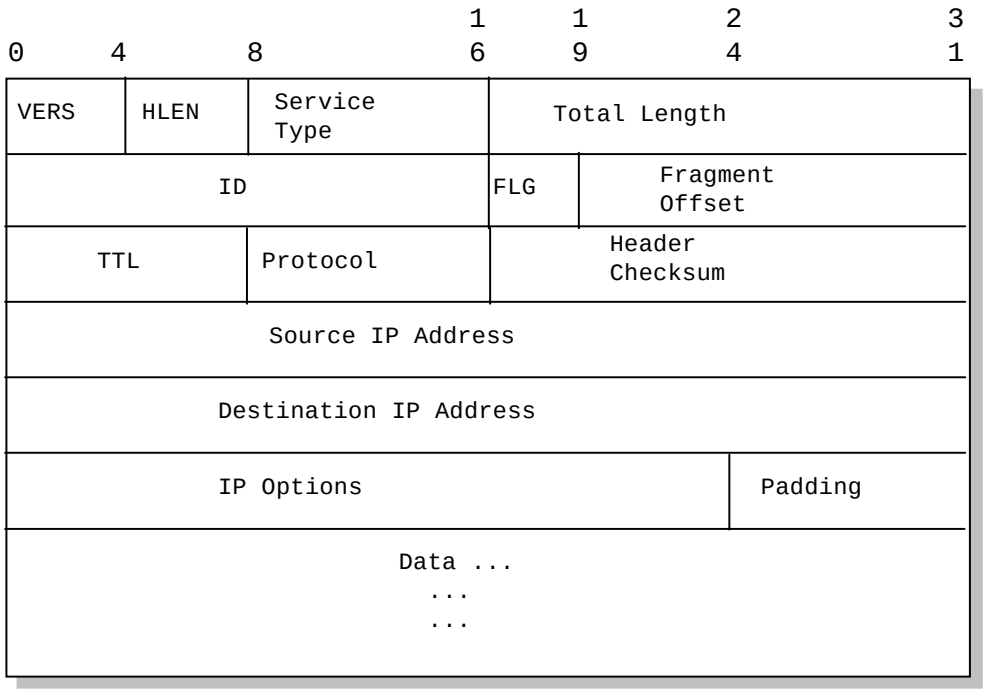

Fig. 2. Format of an IP Datagram Header

The layout of the IP header is illustrated in Fig. 2. The potential targets of the attack are the source and the destination IP addresses (32 bits each) and the header checksum (16 bits). The first 80 bits bits are

disregarded and the last optional fields are assumed to be absent. Note that, in the absence of the optional fields, there is no padding.

There are two commonly used protocols in the transport layer, a connectionless one - the User Datagram Protocol (UDP) [20] and a connection-oriented one - the Transmission Control Protocol (TCP) [21]. Thus an IP datagram may contain either UDP or TCP embedded datagrams. This gives additional targets for the attack.

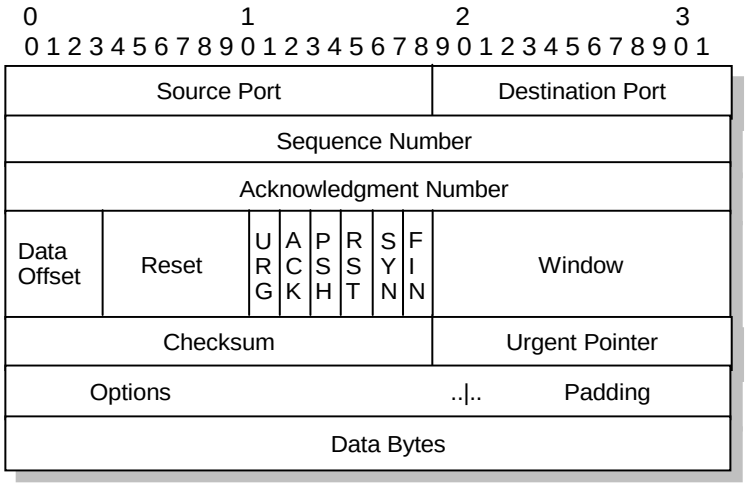

Fig. 3. TCP Segment Format

The TCP header [21] is illustrated in Fig. 3. The source and the destination ports (16 bits each), the sequence and acknowledgement number (32 bits each), which together form the first 96 bits of the TCP header as well as the checksum located between 128-th and 144-th bits are additional important targets of the attack.

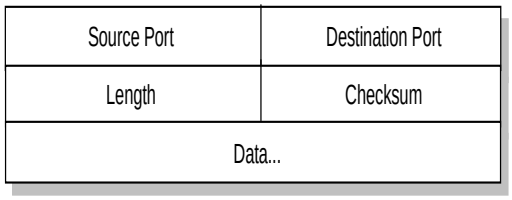

Fig. 4. UDP Segment Format

The UDP header [20] is illustrated in Fig. 4. In this case the source and destination ports as well as the checksum (16 bits each) are additional targets of the attack.

Sometimes, another protocol is encapsulated in the UDP datagrams, e.g. the Real-Time Transport Protocol (RTP) [25]. A description of the RTP header is given in Fig. 5, giving yet another target for the attack: the sequence number field located between 16-th and 32-th bits.

This list of protocols and targets is not exhaustive, the attack is applicable to other protocols. The rationale behind the choice of those targets is that these are the fields that are consulted or manipulated by the IP gateways in an IP network.

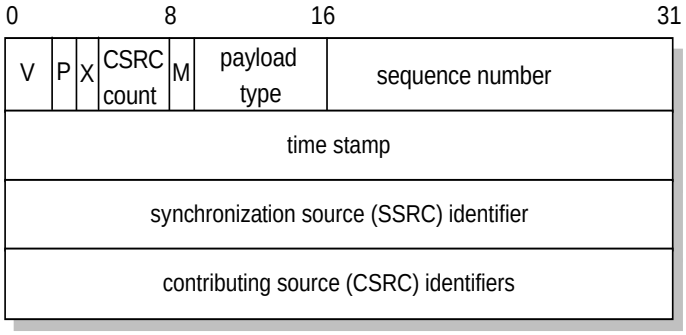

Fig. 5. RTP Header Format

2.4 Internet Checksum

The (internet) checksum [5] is used to discover errors while datagrams are transmitted. The targets chosen in this paper to illustrate the attack: the IP header, the TCP and UDP datagrams, all use checksums to protect their content. The purpose of the internet checksum is to provide an efficient protection against transmission errors but not to provide cryptographic integrity protection of the content.

The outline of the internet checksum is very simple: the data is presented as a sequence of 16 bit integers, then the 1's complement sum of these integers is computed and the checksum is the 1's complement of this sum.

The way this checksum is computed allows the fast incremental update of the checksum when the data over which it is computed is changed. For example, to update the checksum C in the IP header m to the new IP header m', but without recomputing it over the entire new IP header, the updated checksum C' can be easily computed as $C' = C + (m - m')$, as shown in [13]. Note that, to compute the new checksum, one needs to know $m - m'$, but neither m, m' nor the old checksum C are to be known.

3 A DoS Attack

3.1 Fields Manipulation

The choice made by some implementers or users for the "encryption-only" ESP configuration (i.e. without the "costly" authenticated integrity) is based on the false belief that confidentiality protection together with the structure of IP, TCP (or IP, UDP, RTP) datagrams is enough to detect any data change (malicious or not). This amounts at making the hypothesis that any modification to the encrypted datagrams will be detected during the parsing of the decrypted datagrams (i.e. they are supposedly malformed) or during the verification of the checksums.

This is not true; an attacker can, in fact very easily, modify the encrypted (in CBC mode) datagrams in such a way that they are still acceptable for the embedded protocols such as IP, TCP, UDP, RTP. In the attack presented here, the IV is changed in such a way that the first decrypted block of plaintext P'_1 is changed in a predictable way (see section 2.2). More precisely, the IV is replaced with a new one which differs only in the positions to be altered in the first block of plaintext.

Consider a confidentiality protection with a block cipher with length $n = 128$ (see Fig. 6). In this case, P_1, the first block of data, contains the IP header checksum and the source IP address (see section 2.3). Now an attacker can change the source IP address to whatever she/he wants, then using the difference between the old and the new source IP addresses she/he can compute the new correct checksum (see section 2.4) and finally she/he can compute the new IV corresponding to the altered ESP datagram (the rest of the data remains the same). When a gateway receives such a datagram, it accepts it as a valid datagram (i.e. the IP header is valid) and forwards it further to the corresponding destination address.

In the case where $n = 192$ (see Fig. 7), the attacker has more choices. Assuming that TCP/IP (or UDP/IP) datagrams are sent, the attacker can also change the destination IP address, the source and

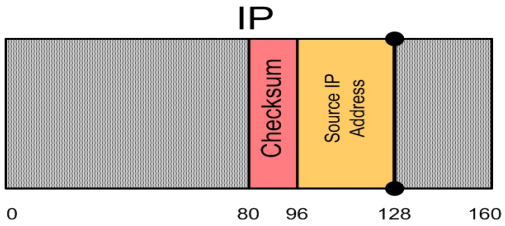

Fig. 6. n=128

destination ports. But since the TCP (UDP) checksum is in block P_2, the modifications brought by the attacker must be such that the TCP (UDP) checksum does not change, i.e. it is still a valid TCP (UDP) datagram.

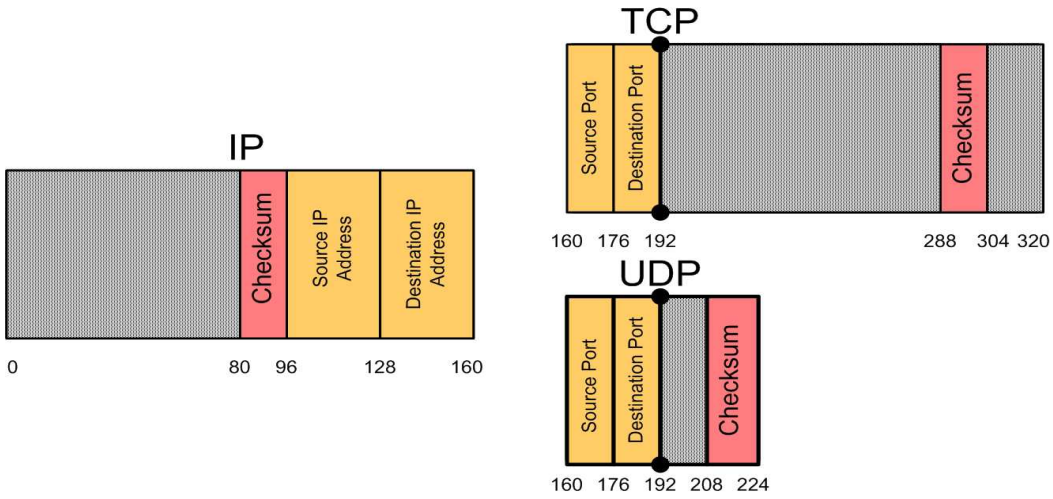

Fig. 7. n=192

In the case where $n = 256$, the attacker has even more choices (see Fig. 8). Assuming that TCP/IP datagrams are sent, the attacker can also change the source and destination ports from the TCP datagram as well as the sequence number and the acknowledgement number but again she/he should do that in such a way that the TCP checksum is not changed since it is still in block P_2. If RTP/UDP/IP datagrams are sent, the attacker can change the source and destination ports from the UDP datagram without being restricted anymore to keep unchanged their UDP checksum, since the checksum is in P_1. In addition the sequence number in the RTP datagram can also be changed.

The attack cannot be applied when $n = 64$ because the first 64 bits of the IP header do not contain any interesting target fields (see Fig. 9).

3.2 Mounting the Attack

Assume a source device communicates with a destination device through IPSec gateways A and B (see Fig. 10), using "encryption-only" IPSec (ESP). Consider the $n = 128$ case. As shown in section 3.1, an attacker can intercept a legitimate ESP datagram, modify the source address, such that the altered ESP datagram and the embedded IP header are still valid and re-inject the new ESP datagram in the network. Gateway B will accept this modified ESP datagram, extract the IP header and embedded payload (see Fig. 1) and forward them to the destination device. Notice that, for any legitimate ESP datagram, the attacker can

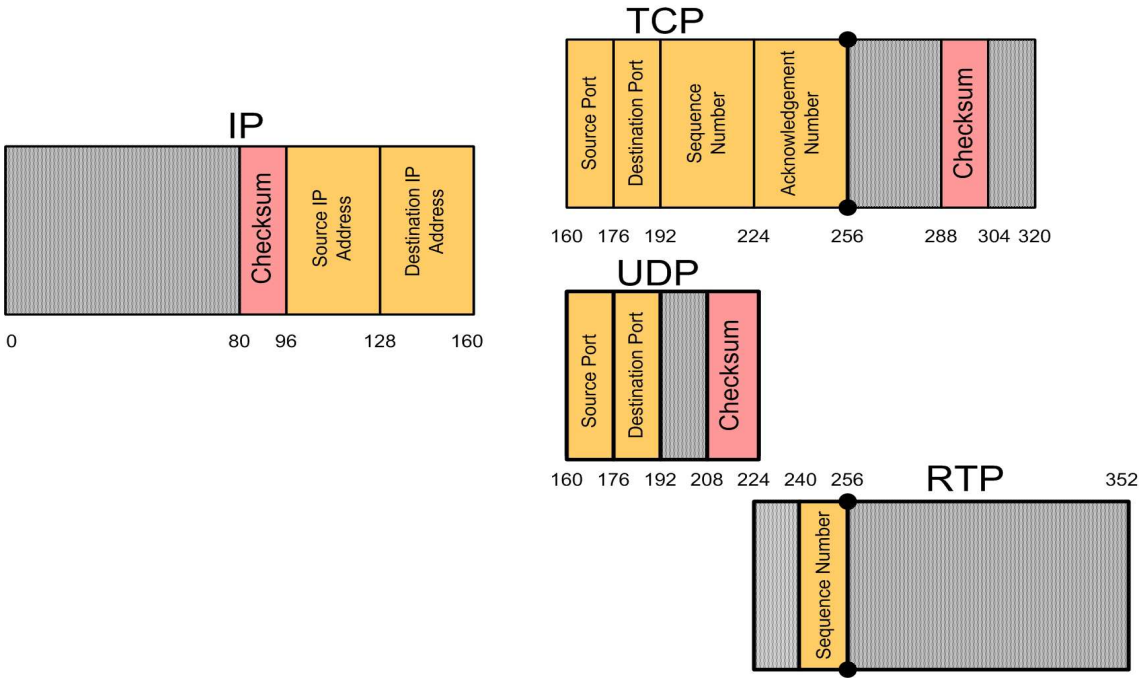

Fig. 8. n=256

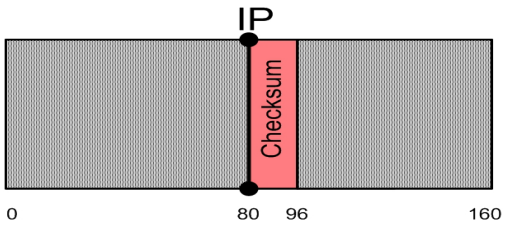

Fig. 9. n=80

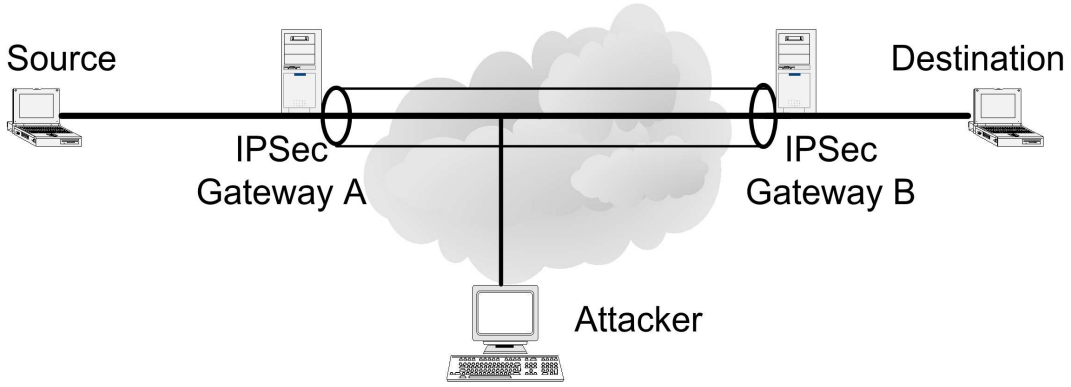

Fig. 10. IPSec

generate up to 2^{32} false source addresses and hence so many false ESP datagrams, which will have the same content as the legitimate one but will claim to come from different sources. All of them will be accepted by

the gateway B and forwarded to the destination device. In this way, the gateway as well as the destination device are overloaded with junk datagrams. The attacker can mount this attack on the fly in real-time.

If we assume (as in [22]) that the attacker knows several source IP addresses of machines which legitimately communicate through the IPSec gateway, then even stateful firewall could not defeat the attack. Moreover it is enough that the attacker knows a significant portion of this IP addresses, which might be more realistic since it may be that the host is on a network with network prefix known to the attacker, or because it is widely used address prefix.

Consider now the $n = 192$ and $n = 256$ cases. In addition to the source address, the attacker can also alter the destination address along with the source and destination ports. The rest of the attack is similar to the $n = 128$ case, except that the junk datagrams can be targeted to any destination address. So the attacker can mount a DoS attack on any selected server in the Internet.

4 Conclusions

This paper presents a very strong and practical DoS attack against the "encryption-only" configuration for ESP (IPSec). It also shows that, contrary to the intuition, increasing the cipher block size jeopardizes the security of the protocol even more, i.e. the attack becomes stronger. We sincerely hope that this attack will serve as a strong argument to improve the existing standard by banning the "encryption-only" configuration and will be a compelling example for the implementors/users of the standard convincing them not to use such a configuration.

5 Acknowledgements

This work has been prepared in conjunction with the design and development of the IPSec communication suite for the Philips Semiconductors Nexperia Mobile platform. The author would like to thank Marijke De Soete, Xavier Serret, Marc Vauclair and Bruno Keymolen for the valuable discussions.

References

1. C. Adams. The CAST-128 Encryption Algorithm, *RFC 2144*, 1997.
2. C. Adams. Constructing Symmetric Ciphers Using the CAST Design Procedure, *Designs, Codes, and Cryptography*, 12(3) 1997, pp. 283–316.
3. M. Bellare, T. Kohno, C. Namprempre. Breaking and provably repairing the SSH authenticated encryption scheme: A case study of the Encode-then-Encrypt-and-MAC paradigm, *ACM Transactions on Information and System Security (TISSEC)*, 7(2), 2004, pp. 206–241.
4. S. Bellovin. Problem Areas for the IP Security Protocols, *Usenix Unix Security Symposium*, 1996, pp. 1–16.
5. R. Braden, D. Borman, C. Partridge. Computing the Internet Checksum, *RFC 1071*, September 1988.
6. N. Borisov, I. Goldberg, D. Wagner. Intercepting Mobile Communications: The Insecurity of 802.11, *MOBICOM'01*, 2001, pp. 180–189.
7. B. Canvel, A. Hiltgen, S. Vaudenay, M. Vuagnoux. Password Interception in a SSL/TLS Channel, *CRYPTO'03*, LNCS 2729, 2003, pp. 583–599.
8. J. Daemen, V. Rijmen. The Design of Rijndael: AES - The Advanced Encryption Standard, *LNCS*, 2002.
9. S. Frankel, R. Glenn, S. Kelly. The AES-CBC Cipher Algorithm and Its Use with IPsec, *RFC 3602*, Sept. 2003.
10. R. Housley. Using Advanced Encryption Standard (AES) Counter Mode With IPsec Encapsulating Security Payload (ESP), *RFC 3686*, January 2004.
11. S. Kent. IP Encapsulating Security Payload (ESP), *Internet-Draft*, March 2005.
12. X. Lai, J. Massey. A Proposal for a New Block Encryption Standard, *EUROCRYPT'90*, pp 389-404.
13. T. Mallory, A. Kullberg. Incremental Updating of the Internet Checksum, *RFC 1141*, January 1990.
14. A. Menezes, P. van Oorschot, S. Vanstone. Handbook of Applied Cryptography, *CRC Press*, 1996.
15. C. McCubbin, A. Selcuk, D. Sidhu. Initialization vector attacks on the IPSec protocol suite, *WETICE'00*, IEEE Computer Society, 2000, pp. 171-175.
16. *NIST FIPS PUB 46*, Encryption Standard (DES), 1977.
17. *NIST FIPS PUB 46-3*, Triple-DES (3DES), 1999.
18. *NIST FIPS PUB 81*, DES Modes of Operation, 1980.
19. *NIST FIPS PUB 197*, Advanced Encryption Standard (AES), 2002.

20. J. Postel. User Datagram Protocol, *RFC 768*, Aug. 1980.
21. J. Postel. Transmission Control Protocol, *RFC 793*, Sept. 1981.
22. K. Paterson, A.Yau. Cryptography in Theory and Practice: The case of Encryption in IPSec, *Cryptology ePrint Archive*, 2005/416, to appear *EUROCRYPT'06*.
23. R. Pereira, R. Adams. The ESP CBC-Mode Cipher Algorithms, *RFC 2451*, Nov. 1998.
24. R. Rivest, The RC5 Encryption Algorithm. *FSE'94*, pp. 86–96.
25. H. Schulzrinne, S. Casner, R. Frederick, V. Jacobson. RTP: A Transport Protocol for Real-Time Applications, *RFC 3550*, July 2003.
26. B. Schneier. Description of a New Variable-Length Key, 64-bit Block Cipher (Blowfish), *FSE'93*, pp. 191–204.
27. A. Tanenbaum, Computer Networks, *Prentice Hall*, 2002.
28. T. Yu, S. Hartman, K. Raeburn. The Perils of Unauthenticated Encryption: Kerberos version 4, *NDSS'04*, 2004.