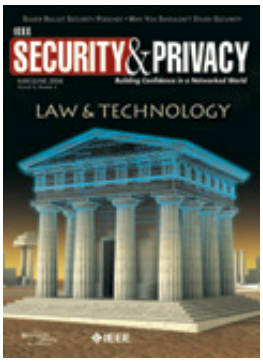
(*IEEE SECURITY & PRIVACY* EXCLUSIVE CONTENT)

CRYPTO CORNER

Lost in Translation: Theory and Practice in Cryptography

Kenneth G. Paterson and Arnold K.L. Yau
Royal Holloway, University of London



MAY/JUNE 2006

The perils of using encryption without authentication or integrity protection are well known in the cryptographic research community. Yet it's exactly the mandatory support for unauthenticated encryption that forms the basis of a serious security flaw in an IPsec implementation we recently discovered. In response, the UK's equivalent to CERT, the National Infrastructure Coordination Centre (NISCC) published a vulnerability advisory about the flaw.[1] Vendors also issued updated recommendations to customers, and we saw a flurry of discussion on Slashdot[2] and the sci.crypt[3] newsgroup. In the aftermath, we asked ourselves, how did this happen?

IPsec

IPsec is a suite of protocols that the Internet Engineering Task Force (IETF) developed to provide security at the Internet layer in the TCP/IP protocol stack. This allows any application that communicates over IP to exploit an IPsec deployment without re-engineering itself. Its transparency to end users makes IPsec a popular choice in implementing virtual private network (VPN) solutions whereby, over the public Internet, a remote user can securely access an organization's internal network, or two geographically disparate networks can connect virtually and securely (site to site). Implementations of IPsec exist in Microsoft Windows 2000 and XP, Mac OS X, all major flavors of Unix, and in the Linux kernel from release 2.6 onward. Open-source projects such as Openswan and strongSwan have sprung up, and IPsec is widely supported in commercial networking hardware.

IPsec operates in either transport or tunnel mode. In transport mode, end hosts need to be IPsec-aware. In tunnel mode (on which we focus here), a pair of security gateways provide IPsec security services on the end hosts' behalf, creating a secure packet tunnel across the Internet. This is achieved by cryptographically protecting the original packet (the inner packet), which is then placed inside a new outer packet before it's transmitted across the Internet. Encapsulating Security Payload (ESP) is an IPsec protocol that provides confidentiality and authentication services, whereas the Authentication Header (AH) protocol provides authentication independently. Usually in ESP, a block-cipher algorithm such as the Advanced Encryption Standard (AES) is used in cipher-block chaining (CBC) mode for bulk encryption. A network administrator can enter encryption keys manually or use the Internet Key Exchange (IKE) protocol to configure them automatically. The security policy database (SPD) and security association database (SADB) contain the information needed to decide whether and how each packet passing through is to be protected by IPsec.

In 1996, Steve Bellovin[4] highlighted the potential dangers of using ESP without authentication. Yet eight years later, when we took a close look at the state of the then-current IPsec standards, we discovered a flaw in IPsec that was exactly due to the use of encryption-only ESP, the configuration that Bellovin had warned against.[5] Based on the IPsec implementation in a version of the Linux kernel, we developed attacks on the ESP protocol in tunnel mode with CBC-mode encryption (that is, without any authentication being provided by ESP itself or by AH). This is arguably a common choice for VPN deployments. A network administrator might reasonably expect that encryption-only ESP should provide at least confidentiality protection for individual packets, but our attacks proved this wasn't so. Moreover, some configurations using ESP with AH are also vulnerable, as are those in which authentication is only supplied by a higher-layer protocol.

Our attacks

Our full paper gives a complete description of our attacks,[5] but here's a brief overview. We assumed the attacker could eavesdrop on all communications between two security gateways and inject arbitrary packets in the tunnel. The attacker's aim was to recover the plaintext packets corresponding to individual encrypted packets that he or she intercepted.

All our attacks are based on a well-known weakness in CBC-mode encryption known as *bit-flipping*. The CBC mode's chaining structure lets someone introduce one or more controlled bit flips (1 to 0, or vice versa) in the plaintext at the decryption end by making specific corresponding bit flips in the ciphertext.

In our first attack, the attacker had some knowledge about the destination address field in the inner packets within the intercepted outer packets. Because the inner packet was encrypted in CBC mode, he or she could modify the ciphertext by carefully flipping bits, such that when the receiving security gateway decrypted the inner packet, the inner destination address corresponded to a host in an attacker-controlled network range. The gateway then simply routed the inner packet to the attacker in plaintext.

In our second and third attacks, the attacker modified the source address and other fields in the inner packet, thus introducing specific types of errors into the inner packet. After processing the inner packet, the security gateway or end host sent a "helpful" error message about the encountered error to the host at the indicated source address. This error message contains information about the error type and, crucially, substantial portions of the original inner packet, all in plaintext. The attacker picked up these error messages and used them to reconstruct complete plaintext packets.

All these attacks were highly efficient and independent of encryption key length. Most attack variants could potentially perform near-real-time cryptanalysis after a short period of initial guesses. Moreover, we proved that these weren't just attacks on paper—we implemented them and demonstrated that they work in a realistic laboratory network setup.

Two main contributing factors allowed our attacks to work. First, the use of confidentiality-only ESP without authentication allowed undetected bit-flipping modifications to the inner packet. Proper authentication would have easily thwarted any bit-flipping by discarding modified packets that failed to authenticate. Second, and perhaps less obviously, the IPsec architecture standard specifies checks against the IPsec policy in the SPD and SADB that an IPsec implementation must perform for each received and decrypted inner packet. These checks would have failed due to the modified address fields in the inner packet, and the implementation should have dropped the modified packet at that point. In Linux, however, these checks are absent altogether.

In an attempt to understand the root causes of this security flaw, we want to scrutinize the process by which the sound and prudent "security message" the cryptography research community advocates is, like a Japanese director's instructions to an American actor, gradually lost in translation as it moves from researchers to standards developers, from standards developers to software implementers, and from software implementers to end users.

Mandatory support for unauthenticated encryption

The need for authenticated encryption is well understood in the cryptographic research community,[6,7] and the lack of authentication has been the basis of many published attacks against real-world protocols such as Secure Socket Shell (SSH),[8] Kerberos,[9] and Point-to-Point Tunneling Protocol (PPTP).[10] In fact, Bellovin discovered attacks on an earlier version of IPsec that are well known by IPsec standards developers.[4]

Bellovin's attacks meant that authentication *was* incorporated in the currently deployed version of ESP (ESPv2), as specified in the RFC 2406 standard.[11] However, the use of authentication is still optional, and implementations are in fact *required* to support encryption-only ESP. An IETF standard represents a consensus among bodies with various vested interests,[12] so a standards committee must trade off even established cryptographic best practices against other criteria such as flexibility, performance, and backward-compatibility. To compensate for this potentially dangerous trade-off, warnings against the use of ESP without authentication have been placed in the text of RFC 2406.

Omitted warnings

Even if RFC 2406 included warnings about encryption-only configurations with the best of intentions, it isn't at all clear what the implementer should do about them: after all, providing support for encryption-only ESP is mandatory in the standard. But the standard provides no clues about whether to prevent such configuration at the policy level, for example, or whether to pass warnings on to the end user.

Even highly skilled software engineers aren't cryptography experts. Their job is to create (to some defined extent) a standards-compliant implementation, which hopefully interoperates well with other systems, all to a hard deadline. An implementer would, as required, provide support for encryption-only ESP. Even if a warning registers in the implementer's mind, its starkness and potency might have worn off considerably after reading through a complicated set of RFCs. This is especially true in the absence of clear instructions about how to respond to the warning. Even if

implementers understand the warning, they might reasonably expect that it's their job to implement encryption-only ESP, but it's the end user's job to configure his or her system securely.

Choosing encryption-only ESP

Let's say that Dan, a network administrator, is tasked with setting up a VPN using IPsec. He knows he needs to set up an encrypted tunnel, and he understands that the Data Encryption Standard (DES) is insecure. Still, Dan is unsure. A Google search points him to http://lartc.org/howto/lartc.ipsec.tunnel.html, and he duly follows the line-by-line instructions, using the provided encryption-only setup, during which he encounters no warnings of any kind. After half an hour of setup time, testing, and even running a network sniffer to ensure that the tunneled packets are indeed encrypted, Dan takes a well-earned coffee break, satisfied with a job well done. The packets are triple-DES encrypted and definitely getting through. What can possibly go wrong?

Missing policy checks on Linux

Well, if Dan's IPsec gateways run on Linux, he could be vulnerable to the devastating attacks described earlier. Every packet that emerges from the tunnel is supposed to be subject to policy checks against the SPD and SADB, but these checks haven't been implemented in the official Linux kernel release 2.6.8.1 that we examined in our work. In fact, these checks aren't designed to be a strong cryptographic defense against malicious attacks, and therefore shouldn't be relied on to ensure security.

Ultimately, the IPsec implementer didn't have a fair chance of implementing these checks properly in the first place anyway—although ESP is specified in RFC 2406, these policy checks are only mentioned in the architecture document RFC 2401.13 Even then, RFC 2401 doesn't explicitly specify that the offending packets be dropped. Without understanding the implications, the IPsec implementer could choose to log the error and continue to process offending packets. Normal functional testing wouldn't reveal this bug because the absence of these policy checks doesn't hamper IPsec's operation.

Lessons to be learned

Our view is that developers of security standards should take more responsibility in ensuring that their standards are secure on paper by eliminating any potentially dangerous choices down the line. The equivalent of Murphy's law in information security states that whenever an insecure configuration can be chosen, it inevitably will be. To help translate standards into a secure implementation, any unnecessary complexity should be avoided. Documentation should be clear, concise, and easy to follow. If an important security decision is supposed to be left to the end user, the document should instruct the implementer to pass all relevant warnings to the user. None of this contravenes the IETF's maxim of allowing flexibility in endpoint implementations.

Software security is a specialized field, and security standard implementers should be trained accordingly. (Naturally, this is harder to ensure in Linux because of its open-source nature.) Having an acute security mindset when writing code helps prevent security bugs before one or more of them become the subject of a future NISCC advisory. System and network administrators should have a good working knowledge—or even better, a deep understanding—of tried and tested information security best practices: the difference between security and vulnerability often lies in subtle details, and, as our study of IPsec illustrates, it's easy to get them wrong.

Not unlike Charlotte and her inattentive photographer husband in the movie "Lost in Translation," there is a distinct lack of understanding between the different communities of cryptography researchers, protocol developers, implementers, and end users. By choice or necessity, these communities are in a committed relationship, yet because each of them operate within their own subculture, use their own lingo, and have their own roles and agendas, they neglect to appreciate that other communities work differently. Through ignorance, assumption, concession, and (mis)interpretation, important messages easily get lost in translation. We would like to see a greater level of collaboration between these communities to bridge the gulf of understanding, so that no one ever again feels like a stranger in a foreign land.

References

1. "NISCC Vulnerability Advisory IPSEC-004033," 9 May 2005.
2. "Flaw Found in VPN Crypto Security," 13 May 2005.
3. "Attacks on IPsec," 14 May 2005.
4. S.M. Bellovin, "Problem Areas for the IP Security Protocols," *Proc. 6th Usenix Unix Security Symp*., Usenix

Assoc., 1996, pp. 1–16.

5. K.G. Paterson and A.K.L. Yau, "Cryptography in Theory and Practice: The Case of Encryption in IPsec," to be published in *Advances in Cryptology: EUROCRYPT 2006*, S. Vaudenay, ed., Springer-Verlag, 2006.
6. M. Bellare and C. Namprempre, "Authenticated Encryption: Relations among Notions and Analysis of the Generic Composition Paradigm," *Proc. Asiacrypt 2000*, LNCS 1976, T. Okamoto, ed., Springer-Verlag, 2000, pp. 531–545.
7. J. Black and H. Urtubia, "Side-Channel Attacks on Symmetric Encryption Schemes: The Case for Authenticated Encryption," *Proc. 11th Usenix Security Symp.*, Usenix Assoc., 2002, pp. 327–338.
8. M. Bellare, T. Kohno, and C. Namprempre, "Breaking and Provably Repairing the SSH Authenticated Encryption Scheme: A Case Study of the Encode-then-Encrypt-and-MAC Paradigm," *ACM Trans. Information and System Security*, vol. 7, no. 2, 2004, pp. 206–241.
9. T. Yu, S. Hartman, and K. Raeburn, "The Perils of Unauthenticated Encryption: Kerberos Version 4," *Proc. Network and Distributed System Security Symp.*, Internet Soc., 2004. (pdf)
10. B. Schneier and Mudge, "Cryptanalysis of Microsoft's Point-to-Point Tunneling Protocol (PPTP)," *Proc. 5th ACM Conf. Communications and Computer Security*, ACM Press, 1998, pp. 132–141.
11. S. Kent and R. Atkinson, *IP Encapsulating Payload (ESP)*, IETF RFC 2406, Nov. 1998.
12. "The Tao of IETF: A Novice's Guide to the Internet Engineering Task Force," IETF, Aug 2001.
13. S. Kent and R. Atkinson, *Security Architecture for the Internet Protocol*, IETF RFC 2401, Nov. 1998.

Kenneth G. Paterson is a professor of information security at Royal Holloway, University of London. His technical interests include cryptography and network security. Paterson has a BSc in mathematics from the University of Glasgow and a PhD in mathematics from the University of London. He is a member of the IEEE Information Theory Society and the International Association for Cryptologic Research, and a fellow of the Institute for Mathematics and Its Applications.

Arnold K.L. Yau is a PhD student in the Information Security Group at Royal Holloway, University of London. His technical interests include network security, side channel attacks, and modes of operation for block ciphers. Yau has an MEng in computing from Imperial College London.

Editors: Peter Gutmann, David Naccache, and Charles C. Palmer

---

Cite this article:

Kenneth G. Paterson and Arnold K.L. Yau, "Lost in Translation: Theory and Practice in Cryptography," *IEEE Security & Privacy*, vol. 4, no. 3, May/June 2006, pp. 69-72.