Cryptovirology FAQ
Version 1.15

Last updated: November 20th, 2004 1:37 am EST

Purpose of this FAQ

The purpose of this frequently-asked-questions (FAQ) document is to answer some common questions about what cryptovirology is and what it isn't. It explains why this topic may be of interest to individuals, software development companies, smartcard manufacturers, and organizations that are concerned with information security and privacy. It ties together certain subjects that might appear to be separate, namely, cryptovirology and kleptography. In fact, there is little difference between the two. It also covers certain topics that may be of interest to research scientists.

This FAQ does not cover virus and worm basics, nor does it give an introduction to cryptography. Suggested sources for these topics are given below. A certain level of familiarity with all of these subjects is assumed. This FAQ does not give advice on how to deal with a cryptovirology attack. The main goal of this document is to explain what cryptovirology is and at the same time prevent the spread of misinformation.

Comments, suggestions, and constructive criticism are welcomed. Direct e-mail to Adam Lucas Young, the FAQ maintainer, at adamy-at-acm-dot-org. This is an evolving document and will hopefully grow and improve with time.

RSA Labs Cryptography FAQ version 4.1 Virus-L/comp.virus FAQ version 2.00

The World Wide Web Security FAQ version 3.1.2 The Worm FAQ

Notice and Disclaimers

It is preferred that a hyperlink be established to this FAQ rather than copying it entirely. This is so that the latest version will be readily accessible. However, hardcopies for educational purposes are entirely acceptable.

Contributors

Moti Yung	*Columbia University*
Edward Wilding	*Data Genetics International*
Matt Bishop	*UC Davis*
Markus Jakobsson	*RSA Laboratories*
Yiannis Tsiounis	*Etolian Capital*
Gary McGraw	*Cigital Labs*
Philippe Golle	*Palo Alto Research Center*
Stuart Schechter	*Harvard University*
David Molnar	*UC Berkeley*
Stuart Staniford	*Silicon Defense*

Contents

1. What is cryptovirology?

Cryptovirology is the study of the applications of cryptography to malicious software [Yo95,YY96a]. It is an investigation into how modern cryptographic paradigms and tools can be used to strengthen, improve, and develop new malicious software (malware) attacks. Cryptovirology attacks have been devised to: give malware enhanced privacy and be more robust against reverse-engineering, give the attacker enhanced anonymity when communicating with deployed malware [YY97a], improve the ability to steal data, improve the ability to carry out extortion, enable new types of denial-of-service, enable fault-tolerance in distributed cryptoviral attacks, and so on. Also, recent work shows how a worm can install a back door on each infected system that opens only when the worm is presented with a system-specific ticket that is generated by the worm's author. This is called an access-for-sale worm [SS03b].

Cryptography has traditionally been used for defensive purposes. Ciphers *defend* against a passive eavesdropper. Public key infrastructures *defend* against an active adversary that mounts a man-in-the-middle attack. Digital signature algorithms *defend* against a forger. E-cash systems *defend* against a counterfeiter and a double-spender. Pseudorandom bit generators *defend* against a next-bit predictor, and so on. Cryptovirology extends beyond finding protocol failures and design vulnerabilities. It is a forward-engineering discipline that can be used for *attacking* rather than *defending*.

2. It seems like cryptovirology just helps virus writers...Why study it?

A cryptovirologist attacks a computer system in the same sense that a cryptanalyst attacks a cryptosystem. Should we stop trying to cryptanalyze cryptosystems and hope that they will be secure? Of course not. By the same token we should not stop trying to anticipate what attackers might do once they break into our computers. The justification for doing research in cryptovirology derives from the proverbial phrase, "It takes a thief to catch a thief" (English, mid-17th century). The notion is that thieves are the experts when it comes to thieving, and they would know best regarding how to catch other thieves. Cryptovirology is a proactive anticipation of the opponent's next move and suggests that certain safeguards should be developed and put into place.

3. What is a cryptovirus?

In computer security, a cryptovirus is defined as a computer virus that contains and uses a *public key*. Usually the public key belongs to the author of the virus, though there are other possibilities as well. For instance, a virus or worm may generate and use its own key pair at run-time [Yo03]. Cryptotrojans and cryptoworms are the same, except they are Trojan horses and worms, respectively. Note that under this definition, a virus that uses a symmetric key and not a public key is not a cryptovirus (this is particularly relevant in the case of polymorphic viruses).

4. What is cryptoviral extortion?

Cryptoviral extortion is a three-round protocol that is carried out by an attacker against a victim. The attack is carried out via a cryptovirus that uses a hybrid cryptosystem to encrypt host data while deleting or overwriting the original data in the process [YY96a]. The protocol is as follows:

(protocol setup phase) An asymmetric key pair is generated by the virus author on a smartcard and the public key is placed within the virus. The private key is designated as "non-exportable" so that even the virus author cannot obtain it's bit representation. Thus, the private key is generated, stored, and used on the smartcard. Ideally, the smartcard will implement two-factor security: something the virus author knows (a PIN number) and something the virus writer has (the smartcard that contains the private key). Also, the card will ideally be immune to differential power analysis, timing attacks, etc. to prevent the virus author from ever learning the bits of the private key. A standards-based approach can be used, e.g., the use of an approved FIPS 140-2 level 2 or higher device (e.g., when it is level 4 the private key will be destroyed if the casing is breached). In the U.S. the virus author cannot be forced to bear witness against himself or herself (Fifth Amendment) and so the PIN can remain confidential. The purpose of this setup phase is to limit the effectiveness of seizing and analyzing the smartcard under subpoena or warrant (competent evidence).

1) (virus author -> victim) The virus author deploys the cryptovirus. At a later time the virus activates on what could be tens or even hundreds of thousands of machines. The remainder of this description will cover the protocol for just one such machine. When the virus activates, it uses a true random bit generator (TRBG) to generate a symmetric key and initialization vector (IV) uniformly at random. It is *essential* that the TRBG produce truly random bits to prevent the symmetric key and IV from being guessed or otherwise determined by the victim at a later date. The virus then encrypts host data with this random symmetric key and IV (e.g., using cipher-block chaining (CBC) mode). The virus concatenates the IV with the symmetric key and then encrypts the resulting string using the public key of the virus author (e.g., using RSA-OAEP). The encrypted plaintext is then held ransom. The virus notifies the victim that the attack has occured (e.g., via a dialog box on the victim's screen) and states that the asymmetric ciphertext will be needed to restore the data. The virus author states his or her demands in return for the data. The virus author and victim can send asymmetrically encrypted messages to each other via a public bulletin board to try to preserve the attacker's anonymity. Alternatively, digital pseudonyms and mix-networks can be used.

2) (victim -> virus author) If the victim complies by paying the ransom and transmitting the asymmetric ciphertext to the virus author then the virus author decrypts the ciphertext using the private key that only the virus author has access to (the one on his or her smartcard). This reveals the symmetric key and IV that was used in the attack.

3) (virus author -> victim) The virus author sends the symmetric key and IV to the victim. These are then used to decrypt the data that was held ransom.

(security) The attack is ineffective if the data can be recovered from backups. Antiviral experts cannot retrieve the private decryption key by analyzing the virus since only the public key will be found. The importance of using hybrid encryption can be seen from the following argument. Suppose that a smartcard was not used and asymmetric encryption were performed using electronic code-book (ECB) mode. It follows that if the private key were revealed to one victim then that victim could give the private key to others. In this case the virus author cannot hope to victimize very many people (perhaps even just one person). The alternative would be for the virus author to demand the entire ECB ciphertext of the data that was held ransom so that it could be deciphered without revealing the private decryption key. However, this is unacceptable for two reasons. First, the file could be huge and therefore make transmission cumbersome. Second, many victims may refuse to cooperate since it would reveal to the virus author the data that was held ransom (privacy violation). This makes the use of hybrid encryption (not just public key encryption) essential.

One question that is typically asked in regards to cryptoviral extortion is the following. How could an extortionist ever expect to receive payment? Truly anonymous *e-cash* (which may someday be minted off-shore) could provide a safe medium for ransom. *Mix networks* are also a critical infrastructure for allowing the extortionist to maintain his or her anonymity. Also, the extortionist could seek information that resides on the host machine instead of money. In this case it may be possible for the malware to asymmetrically encrypt the following: cryptographic hash of the desired data concatenated with the randomly generated symmetric key. This would make it so that the symmetric key could not be recovered without revealing the correct hash [YY96a]. See [YY04] for attacks of this nature.

This leads to an important consideration for organizations that rely heavily on information technology. It is important to be able to estimate the value of an exploit to an outside thief. A model for doing exactly this has been proposed, and it can be used by an organization to gauge its attractiveness to outside thieves [SS03a]. The paper introduces the notion of economic threat modeling, and notes that a cryptoviral extortion attack may allow a thief to profit without taking anything.

5. What is a "black-box" cryptosystem?

Cryptovirology relies heavily on the notion of a "black-box" cryptosystem when it comes to developing provably secure malware attacks against cryptosystems. A black-box cryptosystem is both a theoretical

abstraction as well as a common everyday reality.

In short, a black-box cryptosystem is a cryptosystem that is implemented in such a way that the underlying implementation (source code or circuitry) cannot be scrutinized. A black-box cryptosystem has a public I/O specification and its general functionality is disclosed (though the true functionality could differ).

By definition then, a black-box cryptosystem can only be used without verifying the correctness of its implementation. A smartcard is a black-box cryptosystem unless the user disassembles it, verifies the circuitry and the data that resides in memory, and then reassembles it. Similarly, a cryptosystem that is implemented in software is a black-box cryptosystem unless its code is disassembled and verified. Note that this definition states that the implementation in question must be verified, not the design specification for the whole product line. A manufacturer can sell thousands of cryptosystems and put a backdoor in just one of them.

Millions of people use black-box cryptosystems every day. When an item is purchased on-line, often the secure sockets layer (SSL) is used. When a user does not verify the implementation of SSL, then SSL is a black-box cryptosystem. Most uses of cryptography are like this.

6. What is a subliminal channel?

Many kleptographic attacks are based on the notion of a subliminal channel. The term *subliminal channel* refers to an information transmission channel that can be used to send information out of (or potentially into) a cryptosystem. A subliminal channel is a type of covert channel. However, *covert channels* are broader in scope since they are not specific to cryptosystems (covert channels are discussed in [An01]).

A concrete example will go a long way to explain what a covert channel is. Suppose that Alice and Bob are connected to a computer that is running a multiuser operating system. In a secure operating system that can be used for sensitive (e.g., military) applications it should not be possible for a process that Alice is running to transmit information covertly to a process that Bob is running. But, suppose that a printer is connected to this machine. Each process can make an operating system call to print data. This call will return a result code indicating success or failure. The result code will also indicate if the printer is busy printing out a document. Alice's process can utilize a special communication protocol to speak with a process that Bob is running. For example, printing out two short documents with a brief pause in between could correspond to a binary "1" and printing out one document could be a binary "0." Bob's process calls the operating system routine in a busy waiting fashion to receive bits from Alice's process. This is not a subliminal channel, but it is a covert channel.

Subliminal channels are characterized by: their inability to be detected when in use, their inability to be read even when it is assumed that they are in use, and their inherent channel capacity, or bandwidth. However, the code that transmits information over a subliminal channel is readily identifiable by cryptographers when they inspect the code. So, that attacker must ensure that subliminal channels are only utilized in black-box cryptosystems.

Gus Simmons investigated subliminal channels while assessing the security of a nuclear arms control verification protocol [Si94,Si98]. He then extended the notion to other scenarios. The classic use of a subliminal channel is in the *prisoners' problem* [Si84].

In the prisoners' problem, two prisoners are allowed to communicate to each other but are not allowed to send encrypted messages to each other. They are only permited to exchange public keys and digitally sign their messages. The problem is to devise a way, using the digital signature algorithm in question, for the two prisoners to communicate secretly with each other through digital signatures in such a way that the warden cannot detect or read the subliminal messages.

The applications of subliminal channels grew to encompass insider attacks against smartcards as well [Si85]. A very general type of subliminal channel has been shown to exist that is based on the quadratic

residuosity problem. The channel involves placing a small set of primes, which must remain secret, within a smartcard. It has been shown that this channel can be used by a malicious designer to covertly obtain the DSA private signing key of the user of the smartcard [Si93].

There have been efforts by the research community to try to eliminate subliminal channels in certain algorithms. This is addressed in the FAQ question, "What is being done to mitigate the threat of kleptography?".

7. What is kleptography?

Kleptography is the study of stealing information securely and subliminally [YY97b,WL02,CS03]. In a kleptographic attack, there is an explicit distinction between confidentiality of the messages (e.g., the private keys of the users) and awareness that the attack is taking place. A secure kleptographic attack is undetectable as long as the cryptosystem is a black box. Also, if the black-box is opened, it may be evident that a kleptographic attack is underway, but confidentiality is preserved. Kleptography is a natural extension of the theory of subliminal channels.

Kleptographic attacks often utilize subliminal channels to transmit things like: private signing keys, private decryption keys, symmetric keys, etc. outside of a cryptosystem (e.g., smartcard). The requirement that kleptographic attacks have that exceeds the requirements of a subliminal channel is *robustness against reverse-engineering*. A kleptographic attack is only secure if the confidentiality of the subliminal messages holds even after the black-box is opened and inspected. This must hold for all previously transmitted messages as well as future subliminal messages that may be sent. Asymmetric cryptography is used to achieve this type of confidentiality. It is this added robustness in confidentiality that makes kleptographic attacks more attractive to carry out in practice.

An example will go a long way to explain what a kleptographic attack is. A kleptographic attack against a softare based RSA cryptosystem like PGP has been demonstrated [YY96b,YY04]. In this attack, RSA keys are not generated normally. However, the RSA public modulus n is still the product of two primes p and q. The modulus n is generated such that its upper order bits effectively constitute the asymmetric encryption of a value that allows n to be efficiently factored. Computing such composites n is possible using a well known subliminal channel in the products of two primes. The asymmetric encryption is computed using the public key of the attacker that is embedded in the RSA key generation algorithm. As a result, a database of public keys (i.e., the CA) is a database of public keys and ciphertexts of the corresponding private keys from the perspective of the attacker.

The novelty in this kleptographic attack is the following. It can be deployed in software in a single binary program (that may be code-signed) such that everyone obtains the same copy. The key pairs that the program outputs do not reveal that a kleptographic attack is occuring (they appear to be normal). If a reverse-engineer examines the key generation code then he or she will learn that a kleptographic attack is underway. However the reverse-engineer will still be out of luck in actually learning the private key of anyone who uses the binary.

8. What cryptographic algorithms succumb to kleptographic attacks?

Kleptographic attacks have been devised for factoring-based key generation algorithms [YY96b]. These attacks leak the private key securely and subliminally through the public key itself. This can potentially affect any implementation of RSA, Rabin, Goldwasser-Micali, Paillier, and so on. Kleptographic attacks have been devised for key exchanges [YY97b]. They have been devised for digital signature algorithms as well. These include ElGamal, Schnorr, DSA, Pointcheval-Stern, and others. There is a general way to attack any cryptosystem that displays a modular exponentiation in its output when the exponent is chosen randomly [YY97c,Yo02]. Kleptographic attacks also work against elliptic curve cryptosystems. A more comprehensive listing of the work in this area is given in [YY04].

9. How are cryptovirology and kleptography related?

A kleptographic attack is carried out by building a malicious implementation of a black-box cryptosystem. The malicious implementation is designed to have the same I/O specifications as the correctly implemented cryptosystem, yet securely and subliminally leaks private information to the attacker. Kleptography grew out of the notion of a cryptovirus and the realization that subliminal channels could be used by cryptotrojans to covertly transmit host data to the attacker, data that has been asymmetrically encrypted using the public key of the attacker. The code that carries out a kleptographic attack is therefore malicious software that contains and uses the public key of the attacker. In other words, the code that carries out a kleptographic attack *is* a cryptotrojan. So, from the perspective of cryptovirology, kleptography can be thought of as the study of cryptotrojans that only infect hosts that are cryptosystems.

10. Why should I be concerned about cryptovirology?

If you have confidential information on your computer, your computer is connected to the Internet, and you have not analyzed the code for the programs that you run, then cryptovirology could affect you. If this is true of the company you work for then cryptovirology could affect you. If you purchase items "securely" on-line then you could be affected as well. Advanced malware attacks are a general security problem that can adversely affect the lives of many people.

The potential impact of cryptovirology is especially acute when it comes to smartcard use. Down the road the private key that an executive (or military officer) generates and uses may be compromised and the executive may have to clear up a problem caused by something that he or she was completely unaware of. Similarly, contracts signed using an executive's private key may have to be litigated when the executive disavows them, and asserts that his or her private key was compromised (e.g., by a kleptographic attack). The only way to prove that a kleptographic attack did not occur is to reverse-engineer the device *in question*. This is likely to be both time consuming and expensive. Kleptography has the potential to cause serious damage in these situations, and the potential payoff for embedding kleptographic backdoors is likely to increase with time. To make matters worse, there has been a recent trend towards developing commercial off-the-shelf (COTS) products off-shore. This may affect the likelihood that backdoors will be present in software and hardware that is used domestically.

11. What is a "FIPS" standard?

FIPS stands for "Federal Information Processing Standard." FIPS standards are published in FIPS PUBS. These standards and guidelines are issued by NIST for use by the U.S. government. NIST develops FIPS when there are compelling federal government requirements for security and interoperability and there are no acceptable industry standards or solutions to these requirements.

Of particular relevance to cryptovirology is the FIPS 140-2 standard entitled, "Security Requirements for Cryptographic Modules" [FIPS140] and its annexes. The annexes employ the FIPS 186-2 standard entitled, "Digital Signature Standard (DSS)" [FIPS186] for such things as key generation. Collectively, these standards dictate such things as: physical security requirements, approved RNGs, approved key generation algorithms, zeroization of keys, statistical tests for randomness, electromagnetic interference and radiation issues, and so on.

These standards are relevant to cryptovirology since companies rely heavily on them for information security and privacy. Companies often regard a smartcard as being secure if and only if it is FIPS 140-2 certified. So, these standards should probably be enhanced to mitigate the threat of kleptographic attacks. Some approaches for doing so are addressed in the FAQ question, "What is being done to mitigate the threat of kleptography?" Also, there are other attacks that need to be considered, such as *timing attacks* [Ko96] and *differential power analysis* [KJJ99].

12. Don't FIPS, PKCS, and other standards prevent kleptographic attacks?

FIPS 140, its annexes, and FIPS 186 do not address the threat of kleptographic attacks. The PKCS standards do not address these threats either. In other words, they do not incorporate nor do they attempt

to incorporate existing algorithms that can be used to help minimize the threat of subliminal channels and kleptographic attacks.

However, even if they did, the problem would be far from solved. Suppose that a provably secure subliminal-free protocol between Alice and Bob is "used" as the defense. The fact that the specification is secure does not necessarily mean that the implementation that is in Alice and Bob's hands matches the specification. It is possible to insert a backdoor into some, but not all, of the devices that are produced.

To attempt to implement a verifiable cryptographic black-box, one may try to separate the device that produces randomness from the deterministic algorithm that uses it. This way users can try to verify the correctness of the deterministic portion of cryptographic algorithm. However, the deterministic key generation device may have a secret random number generator (that's hidden in circuitry for instance) that it uses in a given invocation with inverse-polynomial probability. Therefore, the resulting deterministic key generator may in fact be a probabilistic poly-time algorithm that misbehaves somewhat infrequently. It then becomes a game for the manufacturer to see how many invocations of the cryptographic algorithm he or she can compromise without being detected. It is a game because detecting the attack amounts to choosing a large enough polynomial to represent the number of output values that are analyzed. The manufacturer can always choose a larger polynomial for its inverse-polynomial attack probability. This suggests the need to continually check the black-box for misbehaviors.

13. Can't I trust my FIPS 140-2 certified smartcard?

You can trust your FIPS 140-2 certified smartcard if and only if you trust that the manufacturer did not place a backdoor in it. The manufacturer could have placed a backdoor in it and have access to every private key that you generate in your card. However, there are techniques that can be used to alter this trust relationship. For instance, by encrypting a plaintext with multiple public keys that have the corresponding private keys generated, stored, and used in multiple, independently-manufactured smartcards, no single manufacturer will be able to access the data. This of course assumes that the manufacturers are not in collusion. This approach is called *cascading encryptions*, and constitutes a form of secret sharing. This approach and others are detailed in [YY04].

14. Can't I trust open-source software?

Open source software is any software in which the source code is available for users to look at, modify, and compile (actually, this definition overlooks various licensing issues which are not dealt with here). There are open source operating systems, compilers, cryptographic libraries, etc. An advantage to open source is that the source can be analyzed for the presence of backdoors and other malicious code. The code can then be compiled and the resulting program used. However, if there is a backdoor in the *compiler* then a backdoor can wind up in the program that is output. Ken Thompson described an insidious Trojan that inserts a backdoor into a login program each time that the login program is compiled. The Trojan also reinserts itself into the compiler should the compiler ever be recompiled [Th84]. So, unless you have analyzed the compiler that you use to compile open source software for malware, it is possible that malware will nevertheless end up in the newly compiled program.

Similar backdoor functionality can be built into the central processing unit (CPU). For example, each time that a certain sequence of instructions occurs or when certain values exist in machine registers, the CPU can act in a completely rogue and unexpected fashion. So, the short answer to this question is that yes, you can trust open source software if you trust that the code reviewer is competent. However, this alone does not imply that you should trust a compiled version of the open source software.

15. What is being done to mitigate the threat of kleptography?

There is a long line of research geared towards eliminating subliminal channels. By eliminating subliminal channels, the ability to carry out kleptographic attacks is greatly hampered. Gus Simmons introduced the idea of using a protocol involving randomization to destroy subliminal channels [Si84]. To destroy a particular subliminal channel that was identified, Simmons has the warden (in the prisoners'

problem) generate a random number x and modify the message that is sent from one prisoner to the other prisoner using x. This shows the dual nature of randomization; it can pave the way for subliminal communications, but at the same time it can be used in carefully crafted protocols to virtually eliminate the existence of subliminal channels.

For other early results that use randomization to eliminate subliminal channels, see [Si85,DGB88]. See also [De88a,De88b,DGB88,HMP94,De96]. There are also concrete methods to protect against kleptographic attacks in particular. The nature of these solutions varies greatly. For example, some are protocols that involve a third party verifier that verifies that no kleptographic attacks are taking place [JG02]. Some are stand-alone heuristic algorithms [SB93,Yo04], while other solutions involve distributing trust by using multiple, independently designed smartcards. Many of these approaches are covered in [YY04].

This accounts for the efforts on behalf of the research community to mitigate the threats of subliminal channels and kleptography. However, what is happening in practice? Unfortunately, not much attention has been paid to this issue. Part of the purpose of this FAQ is to raise the awareness of these threats, leading to more research, stronger standards, and careful auditing and use of cryptosystems.

16. Is there a cure-all for cryptoviruses?

There is no cure-all for cryptovirus attacks. Cryptoviruses attack computer systems using the same tools that are used to protect computer systems. So, a weakness in the design of a secure cryptovirus implies a weakness in a block cipher, stream cipher, asymmetric cryptosystem, etc. A product that claims to protect you specifically from the threat of cryptoviruses is snake-oil. The best defense is to: verify the authenticity of all programs that you run, protect your machine from infiltration, use existing antiviral tools, be diligent about archiving data, and so forth.

17. Is there a cure-all for kleptography?

Algorithms and protocols already exist that could greatly reduce the threats of certain subliminal channels and kleptographic attacks. It isn't safe to say that these have achieved "cure-all" status, but they are certainly solid measures that could be taken. They have yet to be adopted in industry standards, let alone de facto industry standards. By their very nature kleptographic attacks are *designed* not to be found. Combine with this with the habit that organizations have of covering up attacks to avoid embarrassment, and you have a problem that could be lurking behind the scenes for a very long time.

18. Can a polymorphic virus be a cryptovirus?

Certainly. If a polymorphic virus [Sl94,Bi03] contains and uses a public key then it is a cryptovirus. A polymorphic virus usually contains and uses a *symmetric key* for the purposes of obfuscating and de-obfuscating its own code. So, if this is the only cryptographic key it uses then it is not a cryptovirus.

In laboratory experiments, Fred Cohen produced viruses that had no common sequences of over three bytes between each subsequent generation by using encryption [Co87,Co88]. Such viruses are called *polymorphic viruses*, otherwise known as evolutionary viruses. Numerous polymorphic viruses have appeared in the wild. For example, the Tremor virus is a polymorphic virus that has almost 6,000,000,000 forms [Sl94].

Polymorphic viruses often decrypt and then send control to the main portion of their code, called the virus body, at run-time. They may generate new keys periodically and produce new ciphertexts of their bodies to make virus detection more difficult. The body also contains code that alters (morphs) the decryption code at the beginning of the virus. This makes it such that the virus changes its entire appearance. Although it would make sense to call this a "cryptovirus," this is not the way the computer term was originally defined [YY96a].

19. Isn't a cryptovirus a plant disease too?

Yes. The term cryptovirus was first coined in reference to plant viruses. For instance, there is the "Beet cryptic I" cryptovirus that has been found in Tasmania and that has spread to Europe, U.S.A., and Japan. However, the term was so fitting for viruses that use asymmetric cryptography that it was coined for that purpose as well.

20. Has a cryptovirus ever been written?

Yes. Cryptoviral extortion was described in [Yo95] and it was "implemented" as part of that Masters Thesis. Also, the thesis contains the actual RSA key pair that the experimental cryptovirus used. A summary of the first cryptovirus appears in [YY96a]. The virus was programmed using ANSI C and Motorola assembly and spread on a Macintosh SE/30. It used AT&T's truerand to generate random bits. It used Wheeler and Needham's Tiny Encryption Algorithm (TEA) and RSA. RSA was implemented using the GNU multiprecision library. Careful measures were taken to ensure that the virus did not spread into the wild.

21. Has a cryptoviral extortion attack ever occurred?

It does not appear that a cryptoviral extortion attack that employs asymmetric encryption has ever occurred. It is of course possible that it happened but the victim or victims have not told anyone.

An article reported that a cryptoviral extortion attack has occurred in Europe [Mc96]. The description of the attack is quite vague. It makes no mention of public keys, nor asymmetric encryption, nor hybrid encryption. Also, the statements in the article pertaining to Edward Wilding (the article's "source") are fabricated. For instance, the article states that Wilding had "assisted half a dozen companies targeted by cryptovirus writers." This is patently false and it is rather unfortunate that this fictitious report has been treated as fact [DB97,Ko99].

22. Do kleptographic attacks only work against probabilistic cryptosystems?

No. Kleptographic attacks have been devised for symmetric encryption algorithms that are deterministic [RP97,PG97,YY98,YY03] (however, some of these have been cryptanalyzed [WBDY98,DKZ99,Bi00]). This question is addressed in this FAQ since it encompasses relatively recent results and is therefore likely to be of interest to cryptographers.

It might seem counter-intuitive that information can be leaked within a keyed bijection. However, recall that a block cipher is considered to be secure only if it can withstand a chosen-plaintext attack (or an adaptive chosen plaintext attack). So, it may be assumed that the adversary has plaintext/ciphertext pairs at his or her disposal. In the case of pseudorandom one-time pad encryption (akin to the Vernam cipher), this implies that the pad is known under the bitwise XOR operation. Hence, there can be information transmission outside the black-box cryptosystem. This intuitive explanation closely resembles the principles behind that attacks presented in [YY98,YY03].

23. How can I learn more about cryptovirology?

There are a couple of ways to learn more about cryptovirology. You can access the information at www.cryptovirology.com. You can obtain and read some of the original conference papers. The NEC Research Institute has many of these papers on-line. They can be accessed via SiteSeer. There is also a book covering the subject [YY04].

24. What is www.kleptography.com all about?

Kleptography.com is a website run by Don Ellis that is full of beautiful photographs that he took with a Canon G1 and G2. His pictures range from flowers, to buildings, to automobiles, to people, and breathtaking scenery. Go to www.kleptography.com to have a see for yourself. He independently created the term "kleptography" in 2001 in reference to an old American Indian belief. The belief is that you can steal someone's soul by taking a photograph of him or her. Don has graceously explained the two

different definitions in the "intro" section of his website.

Bibliography

[An01] R. J. Anderson, *Security Engineering: A Guide to Building Dependable Distributed Systems*, Chapter 7, subsection 7.5.3 - Covert Channels, John Wiley & Sons, 2001.

[Bi00] E. Biham, "Cryptanalysis of Patarin's 2-Round Public Key System S Boxes (2R)." In proceedings of Eurocrypt '00, pages 408-416, 1999.

[Bi03] M. Bishop, *Computer Security: Art and Science*, Addison-Wesley-Longman, 2003.

[Co87] F. Cohen, "Computer Viruses - Theory and Experiments," IFIP-TC11 Computers and Security, pages 22-35, v. 6, 1987.

[Co88] F. Cohen, *Computer Viruses*, PhD Thesis, University of Southern California, ASP Press, 1988.

[CS03] C. Crepeau, A. Slakmon, "Simple Backdoors for RSA Key Generation," In proceedings of CT-RSA, Marc Joye (Ed.), LNCS 2612, pages 403-416, Springer, 2003.

[DB97] D. Denning, W. Baugh, "Encryption and Evolving Technologies as Tools of Organized Crime and Terrorism," National Strategy Information Center's US Working Group on Organized Crime, May 15, 1997.

[De88a] Y. Desmedt, "Abuses in Cryptography and How to Fight Them," In proceedings of Crypto '88, S. Goldwasser (Ed.), pages 375-389, LNCS 403, Springer-Verlag, 1988.

[De88b] Y. Desmedt, "Subliminal-free authentication and signature," In proceedings of Eurocrypt '88, LNCS 330, Springer, 1988.

[De96] Y. Desmedt, "Simmons' Protocol is Not Free of Subliminal Channels," Proceedings of the Computer Security Foundations Workshop, pages 170-175, IEEE Computer Society Press, 1996.

[DGB88] Y. Desmedt, C. Goutier, S. Bengio, "Special Uses and Abuses of the Fiat-Shamir Passport Protocol," In proceedings of Crypto '87, C. Pomerance (Ed.), LNCS 293, pages 21--39, Springer-Verlag, 1988.

[DKZ99] Y. Ding-Feng, L. Kwok-Yan, D. Zong-Duo. "Cryptanalysis of the "2R" schemes." In proceedings of Crypto '99, pages 315-325, 1999.

[FIPS140] National Institute for Standards and Technology (NIST), "Security Requirements for Cryptographic Modules," FIPS PUB 140-2, May 25, 2001.

[FIPS186] National Institute for Standards and Technology (NIST), "Digital Signature Standard (DSS)," FIPS PUB 186-2, Jan. 27, 2000.

[HMP94] P. Horster, M. Michels, H. Peterson, "Subliminal channels in discrete-logarithm based signature schemes and how to avoid them," TR-94-13-D, University of Technology Chemnitz-Zwickau, Sept., 1994.

[JG02] A. Juels, J. Guajardo, "RSA Key Generation with Verifiable Randomness," In proceedings of Public Key Cryptography, D. Naccache, P. Paillier (Eds.), pages 357-374, Springer-Verlag, 2002.

[Ko96] P. Kocher, "Timing Attacks on Implementations of Diffie-Hellman, RSA, DSS, and Other Systems," In proceedings of Crypto '96, Neal Koblitz (Ed.), LNCS 1109,pages 104-113, Springer-Verlag, 1996.

[KJJ99] P. Kocher, J. Jaffe, B. Jun, "Differential Power Analysis," In proceedings of Crypto '99, M. Wiener (Ed.), LNCS 1666, pages 388-397, Springer-Verlag, 1999.

[Ko99] Bert-Jaap Koops, "The Crypto Controversy. A Key Conflict in the Information Society," PhD Thesis, Chapter 4: Cryptocriminals, a public concern. page 68, Kluwer Law International, 1999.

[Mc96] M. McCormack, "Europe hit by cryptoviral extortion," Computer Fraud & Security, issue 6, page 3, June, 1996.

[PG97] J. Patarin, L. Goubin, "Asymmetric Cryptography with S-Boxes." In Proceedings of ICICS, Springer, LNCS 1334, pages 369-380, 1997.

[RP97] V. Rijmen, B. Preneel, "A Family of Trapdoor Ciphers." In proceedings of Fast Software Encryption, E. Biham (Ed.), pages 139-148, 1997.

[Si84] G. J. Simmons, "The Prisoners' Problem and the Subliminal Channel," In Proceedings of Crypto '83, D. Chaum (Ed.), pages 51-67, Plenum Press, 1984.

[Si85] G. J. Simmons, "The Subliminal Channel and Digital Signatures," In Proceedings of Eurocrypt '84, T. Beth, N. Cot, I. Ingemarsson (Eds.), LNCS 209, pages 364-378, Springer-Verlag, 1985.

[Si93] G. J. Simmons, "Subliminal Communication is Easy Using the DSA," In proceedings of Eurocrypt '93, T. Helleseth (Ed.), LNCS 765, pages 218-232, Springer-Verlag, 1993.

[Si94] G. J. Simmons, "Subliminal Channels: Past and Present," IEEE European Transactions on Telecommunication, v. 5, n. 4, pages 459-473, 1994.

[Si98] G. J. Simmons, "The History of Subliminal Channels," IEEE Journal on Selected Areas in Communication, pages 452-462, v. 16, n. 4, 1998.

[Sl94] R. Slade, *Robert Slade's Guide to Computer Viruses*, Springer-Verlag, 1994.

[SS03a] S. Schechter, M. Smith, "How Much Security is Enough to Stop a Thief?" In proceedings of Financial Crypto, Springer-Verlag, 2003.

[SS03b] S. Schechter, M. Smith, "Access for Sale - A New Class of Worm," In proceedings of WORM '03, ACM, 2003.

[SB93] M. E. Smid, D. K. Branstad, "Response to Comments on the NIST Proposed Digital Signature Standard," In proceedings of Crypto '92, E. F. Brickell (Ed.), LNCS 740, pages 76-87, Springer-Verlag, 1992.

[Th84] K. Thompson, "Reflections on Trusting Trust," Communications of the ACM, vol. 27, no. 8, pages 761-763, 1984.

[WBDY98] H. Wu, F. Bao, R. Deng, Q. Ye, "Cryptanalysis of Rijmen-Preneel Trapdoor Ciphers." In proceedings of Asiacrypt '98, pages 126-132, 1998.

[WL02] R. Weis, S. Lucks, "All your key bit are belong to us - the true story of black box cryptography," In proceedings of SANE, May 27-31, 2002.

[Yo95] A. Young, "Cryptovirology and the Dark Side of Black-Box Cryptography," Master's Thesis, Moti Yung (Advisor), Comp. Sci. S6902, Columbia University Dept. of Comp. Sci., Summer, 1995.

[Yo02] A. Young, "Kleptography: Using Cryptography Against Cryptography," PhD Thesis, Moti Yung (Advisor), Columbia University, 2002.

[Yo03] A. Young, "Non-Zero Sum Games and Survivable Malware," 4th Annual IEEE Information Assurance Workshop, United States Military Academy, West Point, New York, 2003.

[Yo04] A. Young, "Mitigating insider threats to RSA key generation," CryptoBytes, RSA Laboratories, vol. 7, no. 1, Spring, 2004.

[YY96a] A. Young, M. Yung, "Cryptovirology: Extortion-Based Security Threats and Countermeasures," IEEE Symposium on Security & Privacy, pages 129-141, May 6-8, 1996.

[YY96b] A. Young, M. Yung, "The Dark Side of Black-Box Cryptography, or: Should we trust Capstone?" In proceedings of Crypto '96, Neal Koblitz (Ed.), LNCS 1109, Springer-Verlag", pages 89-103, 1996.

[YY97a] A. Young, M. Yung, "Deniable Password Snatching: On the Possibility of Evasive Electronic Espionage," IEEE Symposium on Security & Privacy, pages 224-235, May 4-7, 1997.

[YY97b] A. Young, M. Yung, "Kleptography: Using Cryptography Against Cryptography," In proceedings of Eurocrypt '97, W. Fumy (Ed.), LNCS 1233, pages 62-74, Springer-Verlag, 1997.

[YY97c] A. Young, M. Yung, "The Prevalence of Kleptographic Attacks on Discrete-Log Based Cryptosystems," In proceedings of Crypto '97, B. S. Kaliski (Ed.), LNCS 1294, pages 264-276, Springer-Verlag, 1997.

[YY98] A. Young, M. Yung, "Monkey: Black-Box Symmetric Ciphers Designed for monopolizing keys," In proceedings of Fast Software Encryption, pages 122-133, Springer-Verlag, 1998.

[YY03] A. Young, M. Yung, "Backdoor Attacks on Black-Box Ciphers Exploiting Low-Entropy Plaintexts," Eighth Australasian Conference on Information Security and Privacy (ACISP), LNCS 2727, pages 297-311, Springer-Verlag, 2003.

[YY04] A. Young, M. Yung, *Malicious Cryptography: Exposing Cryptovirology*, John Wiley & Sons, 2004.