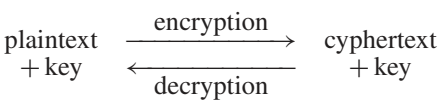
1
Introduction

1.1. Encryption and decryption

The most familiar use of cryptography is in concealing the contents of a message or a document. A *cryptosystem* is a system to achieve this. It consists of two parts: an encryption method and a decryption method. The unaltered, readable form of the message or document is called *plaintext* or *cleartext*. The altered, presumably unreadable form is called *cyphertext*. (Another name for *cryptosystem* is *cypher*, also spelled *cipher*.)



Encryption is the process of obtaining cyphertext from plaintext. The encryption method requires two inputs: the plaintext and the secret key. Similarly, the decryption requires two inputs: the cyphertext and the key – and outputs the plaintext.

In a traditional cryptosystem, the same key is used to encrypt and to decrypt. Such a system is called a *symmetric-key cryptosystem*. If Alice is to send an encrypted message to Bob, they both have to know the key. (This is to be distinguished from a *public-key cryptosystem*, in which one key is used for encrypting and another for decrypting. We discuss public-key cryptosystems later.) If Alice and Bob wish to keep the contents of the message a secret, the key had better be a secret, for any eavesdropper who knows the key and intercepts the cyphertext can determine the plaintext.

We are assuming here that every prospective eavesdropper knows the encryption and decryption methods; this assumption is discussed in Section 1.4. We continue to make this assumption throughout this text; it is methodologically fundamental to the modern study of cryptography. To

amplify, *always assume that every prospective adversary knows all the details of every cryptographic system you employ.*

Cryptanalysis is the process of trying to crack a cryptosystem; an eavesdropper would employ methods of cryptanalysis to try to figure out the contents of Alice's message to Bob. In Section 1.6, we briefly describe some of the different kinds of attacks an eavesdropper might mount, but details of cryptanalysis are beyond the scope of this text.

1.2. Channels, secure and insecure

We use a variety of communication media: telephone networks, the radio waves, TV cable networks, local computer network, the Internet, print media. Banks use one network to connect their automatic teller machines to central computers and another network to execute electronic funds transfer. Paging services use a combination of cable and radio. Some communication between satellite and earth makes use of microwaves. I often use my computer's memory to communicate between my present self and my future self. When I use a telephone, even before my voice reaches the handset, the sound passes through the air. In fact, in the act of dialing I communicate with the phone system.

We would like to apply the concepts of digital security to any and all of these communication media. To this end, we use a single, generic term, *channel*, to abstract away the differences between them. A *channel* is a medium for communication between two parties. (I like to think of a string connecting two tin cans.)

Of course, most communication media enable communication between more than two parties. However, for most purposes the notion of a channel that connects two parties is sufficient; if more parties are involved, we simply invoke the presence of more channels.

Whether one considers a particular communication medium (e.g., the phone network) to be secure or insecure depends on one's point of view. For example, we ordinarily think of the phone network as being reasonably secure, but many hundreds of government wiretap and bug orders are approved each year, and each order leads to surveillance of a couple of thousand conversations on average.

For the purposes of studying cryptography, we shall simply declare whether we consider a channel to be secure or insecure. The channel is insecure if it is possible for a third party (an eavesdropper) to intercept (listen in on) a message passing through the channel. In some cases, it may even be possible for the eavesdropper to alter the message as it goes from sender to receiver.

A *secure* channel is a channel that is immune to eavesdropping or tampering. Cryptography is much more interesting, of course, when applied to an insecure channel. Fortunately (or unfortunately, depending on your point of view), insecure channels abound in real life. In the remainder of this section, we outline some of the characteristics of three communication media that make them insecure. These are intended as examples; readers can no doubt find sources of insecurity in other communication media, including those mentioned at the beginning of this section.

1.2.1. The Internet

Today, the most obvious applications of cryptography involve the Internet, for three reasons. First, the Internet is in part composed of computers, and computers are great at cryptography. Second, the Internet is the perfect medium for facilitating spontaneous communication between large numbers of previously unacquainted parties. Third and most important, the very structure of the Internet renders it needful of security mechanisms. Each computer in the Internet is *directly* connected to very few other computers. When you send a message from the computer in your bedroom in Rhode Island to your parents' computer in California, your message travels through many intermediate computers. Each of these intermediate computers is expected to do its best to forward your message to another computer closer to the message's destination, and there are mechanisms to check whether the message eventually finds its way there. However, nothing in the system prevents an intermediate computer from storing a copy of the message it forwards, or altering the message before forwarding it. A rogue computer could even fail to forward your message but return a message to your computer indicating that your original message did get through.

(The situation is not as dire as this description would make it seem; the route taken by your message is frequently unpredictable, and often your message is split into pieces and the pieces sent along different routes. Most of the intermediate computers encountered are not likely to be rogues.)

A worse situation arises during *remote log-in*. If you have an account on a computer in California, you can use your computer in Rhode Island to log in to the computer in California, by using a program called "telnet." The Californian computer will, of course, send a message requesting your password. When you provide it, the password travels through all the intermediate computers between Rhode Island and California. Any one of these intermediate computers could store the address of the Californian computer, your user name, and your password, and later gain access to your account. There is evidence

that tens of thousands of passwords have been obtained in this way by rogue computers.

Now that the Internet is widely used for commercial purposes, the dangers have proliferated. Suppose you use your Web browser to obtain up-to-date stock information before making a business decision; it is possible (though not all that probable) that your business rival has set up a computer to substitute his own, falsified data for the stock information you sought. Suppose you decide to download your favorite computer-game company's latest demonstration program. It is possible that some rogue computer has intercepted your browser's request and responded with an altered, virus-infected version of the program you requested. Finally, suppose you are viewing the Web page of an online bookseller. They provide a way to encrypt your credit card number, and you send it to them. The page you were viewing might not be the bookseller after all; it might be a front put up by a rogue computer.

1.2.2. Local area networks

One need not communicate with locales as far apart as California and Rhode Island to encounter security risks. Your *local area network* (which connects your computer to servers that store programs and provide mail service) may not be secure; rogue computers connected to that network may be able to intercept your communication (using a program called a "packet-sniffer") and even inject altered data.

1.2.3. Cellular phones

Of course, cellular phone communication takes place over the airwaves and thus can be intercepted, as Newt Gingrich, Speaker of the House, found out in January 1997. It is not only one's conversations that are at risk; when a call is initiated, the cellular phone transmits an account number to be charged. The cellular telecommunications industry suffered an estimated 450 million dollars in fraud losses in 1995,[1] much of that due to cellphone "cloning": bandits would intercept the account numbers and install the numbers on other cell phones, thereby "cloning" the original phones. Calls on the new cell phones would be charged to the old ones.

The cellular telecommunications industry is taking steps to prevent fraud, incorporating security features. Old communications standards die hard,

[1] Source: Bell Atlantic, http://www.ba.com/nr/95/may/freddie.html

however. Moreover, the security features introduced are not necessarily full-proof, as discussed in Section 1.3.

One need not even be using one's cellular phone to be vulnerable to a security risk. As a cellular phone travels, it registers its change of location with the system; this transmission can be intercepted and used to help determine the location of the phone (and of its owner). The FBI has requested standards to mandate that every cellular phone could provide police with the location of phone users.[2]

1.3. Security through obscurity

Consider the following:

- During World War II, the U.S. military hired Navajo Indians to handle secure communications in the Pacific theater. No cryptography was used to ensure the privacy of the communication; it was considered secure enough that the communication took place in the Navajo language. As far as we know, the security of this system was never broken.
- In Edgar Allan Poe's "The Purloined Letter," the eponymous epistle was "hidden" in an obvious place – and remained hidden from the police (though not from the story's protagonist).
- An imaginary colleague of mine at another university wants to make the rough draft of the midterm available to his TAs to review – so he leaves the midterm in a public folder but names it "systemstats," thinking nobody would bother to look at this folder.

These are examples of "security through obscurity." The idea is that one can achieve security by keeping a particular mechanism secret. The concept (though not the name) has a long history – a history with many failures. When one party (often a government) relies on a secret method for achieving security, another determined party can often exploit espionage, luck, and/or long hours of work to determine the method being used. Examples from the twentieth century include the Polish-British success in cracking the German enigma cryptosystem and the American success in cracking the Japanese encryption machine 07-shiki O-bun In-ji-ki.[3]

Developers of security methods are still seeking security through obscurity. A recent example, dating from March 1997, concerns a cryptosystem used in

[2] The Communications Assistance for Law Enforcement Act. Source: http://www.epic.org/privacy/wiretap/

[3] Kahn, *The Codebreakers.*

digital cellular telephony to encrypt numbers punched in by a user of a cellular phone, for example, for dialing. The details of the cryptosystem were supposed to be known only to industry engineers, but were leaked and published on the Web. Some researchers subsequently showed that the system is much less secure than desirable.[4]

Relying on obscurity to achieve security is a mistake. This is especially the case in an age when digital security is ubiquitous, when it is used primarily for commerce, when most secure communication takes place between parties that have had no prior contact. The *usefulness* of security mechanisms in commerce depends on their being widely distributed. Under these circumstances, it is wildly optimistic to assume that details of a security method will not fall into the hands of someone capable of finding and exploiting any security hole.

1.4. The alternative: The Kerckhoffs Doctrine

Given that obscurity is not a reliable source of security, what alternatives do we have? After all, *something* must be secret.

The answer, or at least part of it, was first articulated in 1881 by an impressive polymath who was at that time a professor of German in a Paris university.[5] He was born Jean-Guillaume-Hubert-Victor-François-Alexandre-Auguste Kerckhoffs von Nieuwenhof but later shortened his name to Auguste Kerckhoffs. He spent much of his working life teaching English and German but on occasion also taught Italian, Latin, Greek, history, and mathematics. The subjects of books he authored include grammar, the origins of German drama, the relation of art to religion, and, of course, cryptography. In *La Cryptographie militaire*, he recognized that cryptosystems then being proposed were far from secure:

> . . . I am stupefied to see our scholars and our professors teach and recommend for wartime use systems of which the most inexperienced cryptanalyst would certainly find the key in less than all hour's time.

Kerckhoffs recognized that the spurious claims of security, often based on such misleading calculations as the number of centuries required to systematically try all keys, helped foster ignorance and gullibility about cryptography:

> . . . it may. . . be believed that the immoderate assertions of certain authors, no less than the complete absence of any serious work on the art of solving secret writing,

[4] http://www.counterpane.com/cmea.html

[5] This material is from Kahn, *The Codebreakers.*

> have largerly contributed to give currency to the most erroneous ideas about the value of our systems of cryptography.

Kerckhoffs thus realized that the security of a cryptosystem cannot generally be established by its developer via pure reasoning but must be subjected to the rigors of attack by independent crypt analysts.[6] He inferred the principle that we shall call the Kerckhoffs Doctrine: *The security of a cryptosystem should depend only on the secrecy of the key used, not on the secrecy of the system itself.*

The Kerckhoffs Doctrine requires you to assume that every prospective eavesdropper and hacker has access to all the details of the cryptosystem you are using. This is a pessimistic assumption but, especially in cases where there is a great deal at stake, one that is frequently warranted. As Schreier writes, "...one would assume that the CIA does not make a habit of telling Mossad [the Israeli secret service] about is cryptographic algorithms, but Mossad probably finds out anyway." The Kerckhoffs Doctrine is particularly applicable for security systems intended to be used by many people (such as the security systems in Web browsers), for all these people must be granted at least implicit access to the details of the systems – and your adversary is likely to be among these people.

The Kerckhoffs Doctrine is appealing but would be empty if no cryptosystem existed that could satisfy the Doctrine's stringent requirement. What hope do we have of coming up with a system so secure that a determined adversary with full knowledge of the system could not crack it? As Edgar Allan Poe wrote in his celebrated story *The Gold-Bug*, "it may well be doubted whether human ingenuity can construct an enigma of the kind which human ingenuity may not, by proper application, resolve." As I hope to show in this text, there are good reasons to believe that secure cryptosystems do in fact exist.

1.5. A taxonomy of cryptography

Although *cryptography* in a narrow sense means the study of secret writing (*encryption*), it has come to be a generic term for a variety of technologies arising in digital security. In studying cryptography, it is useful to make distinctions among different conceptual levels. For now, the following five categories suffice:

1. Vague security goals
2. Formal security goals

[6] We shall see an exception to this rule when we study perfect secrecy.

3. Protocols
4. Cryptographic building blocks
5. Realizations of these building blocks

Vague security goals are the most abstract elements in cryptography, and are what motivate all the rest. "I want my communications with so-and-so to be private." "I want an unforgeable document." "I want to ensure that any modifications to the computer program are detectable." Though the point of cryptography is to serve such goals, it is hard to verify whether such goals have been met because the terms are so vague. To ensure that communication will be private, for example, we must guarantee that no eavesdropper learns the contents of the communication. What information and what resources are available to a prospective eavesdropper? What actions are available to her? Whether or not it is possible for the eavesdropper to learn the content of communication depends crucially on the answers to these questions.

For more precision of language and more specificity, we must turn to *formal security goals*. A formal security goal states that an adversary pursuing a particular kind of attack and possessing certain resources (e.g., time, memory, computational power) cannot succeed. We discuss different kinds of attacks in Section 1.6. We can delve deeper into formal security goals only after we have discussed probability theory and computational complexity.

A cryptographic *protocol* is a set of rules for communicating between multiple parties to achieve some cryptographic goal. For example, we discuss later how two parties can communicate to agree securely on a secret key; the protocol to achieve this is called *exponential key agreement*. One can think of encrypt-send-receive-decrypt as a very simple protocol: one party encrypts the plaintext and sends it to the other party, who decrypts it. Another simple protocol, used for *authentication*, is logging in to a computer: a person sends a message requesting access to the computer; the computer responds by asking for the password; the person sends back the password; the computer sends back a message granting access. We will study more sophisticated authentication protocols as well.

Protocols are built on *cryptographic building blocks*. The most familiar kind of cryptographic building block is a cryptosystem, but in this text we discuss several other kinds as well, including one-way functions, message digest functions, and digital signature systems. One commonly associates with each cryptographic building block an informal or formal security goal.

It is often useful to distinguish between conceptual building blocks and particular *realizations of these building blocks*. Thus, for example, there are many different cryptosystems but they all fill essentially the same role in

achieving security goals. Part of the power of modern cryptography derives from researchers having identified and characterized useful cryptographic building blocks at an abstract level.

Abstraction enables the developer of a cryptographic protocol to achieve much greater generality. For example, if the protocol makes use of a cryptosystem, the developer need not spell out precisely which cryptosystem to use ("encrypt a message using the cryptosystem DES…"), only what security goals the cryptosystem must satisfy. This generality in turn makes the protocol more robust: if someday the use of DES is abolished (as some argue it should be today), the protocol can simply use another cryptosystem, such as IDEA.

The generality afforded by abstraction is not merely a tool of researchers. Protocols such as SSL, which is built into current Web browsers and is invoked for most secure Web interactions, is designed to make use of any of a large variety of realizations of the fundamental cryptographic building blocks.

For the most conceptually novel building blocks, such as public-key encryption and digital signatures, we will describe the precise realization used in practice. However, as mentioned in the preface, for some other building blocks – symmetric-key encryption, one-way functions, and message digest functions – we do not describe in detail the realizations typically used in practice. Those details are not all that enlightening, and the design principles that gave rise to them are beyond the scope of this text. In their place we describe other realizations for the same cryptographic building blocks. The realizations we describe, if used correctly, can be just as secure as the practical realizations. We will be careful to inform readers when our descriptions diverge from practice.

1.6. Attacks on cryptosystems

Let us make this slightly more concrete by focusing on the most familiar cryptographic tool, encryption. Suppose Alice needs to send a private message to Bob, and therefore encrypts the message using a secret key known only to her and Bob. The goal here of our eavesdropper, Eve, is to violate privacy of communication. In the worst case, this could mean that Eve is able to read precisely what has been communicated. Typically this entails her knowing the secret key used for encryption. (Recall Casanova's story from the preface.) However, it would still be a violation of Alice and Bob's privacy if by observing the cyphertext Eve were to gain some information about the contents of the message without learning it precisely and wholly. Even such a partial break in

a cryptosystems can have serious implications (e.g., the VENONA project, which we shall study later).

The phrase "gain some information" is pretty vague. We do not have the technical apparatus, at least at this stage in the text, to formalize that notion. Let us proceed, nevertheless, to consider what sort of information Eve might use in cracking the security of the cryptosystem. Traditionally, a distinction is made between four types of attacks. In increasing order of power, they are:

- Cyphertext-only attack
- Known plaintext attack
- Chosen plaintext attack
- Chosen cyphertext attack

Suppose that after intercepting one or more cyphertexts sent from Alice to Bob, Eve could analyze these cyphertexts and gain some information about the corresponding plaintexts. We say in this case that Eve has used a *cyphertext-only* attack.

Suppose Eve intercepts a cyphertext for which she happens to already know the plaintext. She might be able to analyze the relation between plaintext and cyphertext to determine some information about the key. Such information could help her to decrypt other cyphertexts encrypted with the same key. We say in this case that Eve has used a *known plaintext attack*. In a more general version of this attack, she might be able to obtain and make use of *many* plaintext–cyphertext pairs.

Now we consider a more active attack by Eve. Suppose she is able to choose plaintexts for Alice to encrypt. Alice innocently goes along, encrypting plaintexts of Eve's choosing. It is conceivably useful to Eve to have plaintext–cyphertext pairs for which she has chosen the plaintext. What we have described is called a *chosen plaintext* attack. As an historical example, the U.S. ambassador to Japan during World War II reported that "one of the high officials of the Japanese Government wanted to send a secret message to our Government which they did not want the Japanese military to see and in passing this message on they asked me to please put it in our most secret code. I said of course I would do so."[7]

In a slight twist on her attack, Eve might be sufficiently friendly with Bob to get him to decrypt putative cyphertexts of her choosing. That is, Eve makes up alleged cyphertexts without having any idea of the corresponding plaintext, and convinces Bob to provide her the decryptions of these alleged cyphertexts. This is called a *chosen cyphertext* attack, and it can in fact be useful, for example,

[7] Kahn, p. 495.