Secrecy, Security, and Obscurity

Bruce Schneier

https://www.schneier.com/crypto-gram/archives/2002/0515.html#1

A basic rule of cryptography is to use published, public, algorithms and protocols. This principle was first stated in 1883 by Auguste Kerckhoffs: in a well-designed cryptographic system, only the key needs to be secret; there should be no secrecy in the algorithm. Modern cryptographers have embraced this principle, calling anything else "security by obscurity." Any system that tries to keep its algorithms secret for security reasons is quickly dismissed by the community, and referred to as "snake oil" or even worse. This is true for cryptography, but the general relationship between secrecy and security is more complicated than Kerckhoffs' Principle indicates.

The reasoning behind Kerckhoffs' Principle is compelling. If the cryptographic algorithm must remain secret in order for the system to be secure, then the system is less secure. The system is less secure, because security is affected if the algorithm falls into enemy hands. It's harder to set up different communications nets, because it would be necessary to change algorithms as well as keys. The resultant system is more fragile, simply because there are more secrets that need to be kept. In a well-designed system, only the key needs to be secret; in fact, everything else should be assumed to be public. Or, to put it another way, if the algorithm or protocol or implementation needs to be kept secret, then it is really part of the key and should be treated as such.

Kerckhoffs' Principle doesn't speak to actual publication of the algorithms and protocols, just the requirement to make security independent of their secrecy. In Kerckhoffs' day, there wasn't a large cryptographic community that could analyze and critique cryptographic systems, so there wasn't much benefit in publication. Today, there is considerable benefit in publication, and there is even more benefit from using already published, already analyzed, designs of others. Keeping these designs secret is needless obscurity. Kerckhoffs' Principle says that there should be no security determent from publication; the modern cryptographic community demonstrates again and again that there is enormous benefit to publication.

The benefit is peer review. Cryptography is hard, and almost all cryptographic systems are insecure. It takes the cryptographic community, working over years, to properly vet a system. Almost all secure cryptographic systems were developed with public and published algorithms and protocols. I can't think of a single cryptographic system developed in secret that, when eventually disclosed to the public, didn't have flaws discovered by the cryptographic community. And this includes the Skipjack algorithm and the Clipper protocol, both NSA-developed.

A corollary of Kerckhoffs' Principle is that the fewer secrets a system has, the more secure it is. If the loss of any one secret causes the system to break, then the system with fewer secrets is necessarily more secure. The more secrets a system has, the more fragile it is. The fewer secrets, the more robust.

This rule generalizes to other types of systems, but it's not always easy to see how. The fewer the secrets there are, the more secure the system is. Unfortunately, it's not always obvious what secrets are required. Does it make sense for airlines to publish the rules by which they decide which people to

search when they board the aircraft? Does it make sense for the military to publish its methodology for deciding where to place land mines? Does it make sense for a company to publish its network topology, or its list of security devices, or the rule-sets for its firewalls? Where is secrecy required for security, and where it is mere obscurity?

There is a continuum of secrecy requirements, and different systems fall in different places along this continuum. Cryptography, because of its mathematical nature, allows the designer to compress all the secrets required for security into a single key (or in some cases, multiple keys). Other systems aren't so clean. Airline security, for example, has dozens of potential secrets: how to get out on the tarmac, how to get into the cockpit, the design of the cockpit door, the procedures for passenger and luggage screening, the exact settings of the bomb-sniffing equipment, the autopilot software, etc. The security of the airline system can be broken if any of these secrets are exposed.

This means that airline security is fragile. One group of people knows how the cockpit door reinforcement was designed. Another group has programmed screening criteria into the reservation system software. Other groups designed the various equipment used to screen passengers. And yet another group knows how to get onto the tarmac and take a wrench to the aircraft. The system can be attacked through any of these ways. But there's no obvious way to apply Kerckhoffs' Principle to airline security: there are just too many secrets and there's no way to compress them into a single "key." This doesn't mean that it's impossible to secure an airline, only that it is more difficult. And that fragility is an inherent property of airline security.

Other systems can be analyzed similarly. Certainly the exact placement of land mines is part of the "key," and must be kept secret. The algorithm used to place the mines is not secret to the same degree, but keeping it secret could add to security. In a computer network, the exact firewall and IDS settings are more secret than the placement of those devices on the network, which is in turn more secret than the brand of devices used. Network administrators will have to decide exactly what to keep secret and what to not worry about. But the more secrets, the more difficult and fragile the security will be.

Kerckhoffs' Principle is just one half of the decision process. Just because security does not require that something be kept secret, it doesn't mean that it is automatically smart to publicize it. There are two characteristics that make publication so powerful in cryptography. One, there is a large group of people who are capable and willing to evaluate cryptographic systems, and publishing is a way to harness the expertise of those people. And two, there are others who need to build cryptographic systems and are on the same side, so everyone can learn from the mistakes of others. If cryptography did not have these characteristics, there would be no benefit in publishing.

When making decisions about other security systems, it's important to look for these two characteristics. Imagine a "panic button" in an airplane cockpit. Assume that the system was designed so that its publication would not affect security. Should the government publish it? The answer depends on whether or not there is a public community of professionals who can critique the design of such panic buttons. If there isn't, then there's no point in publishing.

Missile guidance algorithms is another example. Would the government be better off publishing their algorithms for guiding missiles? I believe the answer is no, because the system lacks the second characteristic above. There isn't a large community of people who can benefit from the information, but there are potential enemies that could benefit from the information. Therefore, it is better for the government to keep the information classified and only disclose it to those it believes should know.

Because the secrecy requirements for security are rarely black and white, publishing now becomes a security trade-off. Does the security benefit of secrecy outweigh the benefits of publication? It might not be easy to make the decision, but the decision is straightforward. Historically, the NSA did not publish its cryptographic details -- not because their secrecy improved security, but because they did not want to give their Cold-War-world enemies the benefit of their expertise.

Kerckhoffs' Principle generalizes to the following design guideline: minimize the number of secrets in your security system. To the extent that you can accomplish that, you increase the robustness of your security. To the extent you can't, you increase its fragility. Obscuring system details is a separate decision from making your system secure regardless of publication; it depends on the availability of a community that can evaluate those details and the relative communities of "good guys" and "bad guys" that can make use of those details to secure other systems.