Evil Offspring – Ransomware and Crypto Technology

Hilarie Orman • *Purple Streak*

Twenty years ago I encrypted a file that I was editing. Ironically, it concerned encryption. The file was on a Unix time-sharing system, and I needed to keep it confidential. Encrypting it meant that I need not rely on the access controls on Unix, which were easily circumvented through bugs in privileged programs. A few days later I returned to edit the file, but I had forgotten the password that unlocked the encryption key. Out of an abundance of caution and foolish faith in my own memory, I had not written it down. I had to start all over to create the file.

My brush with denial of service by encryption didn't suggest a new business venture, but that just shows my lack of imagination. Though I didn't realize it, there were already the rumblings of a lucrative revenue stream enabled by malicious encryption. Now, modern encryption methods have become the basis for monetizing malware. Several varieties of "crypto ransomware" have evolved that take advantage of modern encryption technology. The evil code encrypts all your files, deletes your backups, and asks for a Bitcoin payment in exchange for the decryption key. Hospitals, police departments, small businesses, and ordinary individuals have been faced with the choice of abandoning their data or paying the ransom.

Crypto ransomware is an interesting kind of new crime, one enabled by asymmetric cryptography, block-chaining systems, a large network of botnets, and the fact that no matter how much we wish otherwise, the software that drives our computing devices always has exploitable bugs.

Crypto ransomware is worrisome from a national security standpoint. In the classic treatise *The Art of War*,[1] there's a theme of achieving an advantage through position, preparation, or surprise. But with software technology, it's possible that any advantage can be replicated and turned against an enemy, be it a defender or attacker. Also, consider the section 6 item in that document as advice regarding a zero-day attack:

> *The spot where we intend to fight must not be made known; for then the enemy will have to prepare against a possible attack at several different points; and his forces being thus distributed in many directions, the numbers we shall have to face at any given point will be proportionately few.*

The reality of today's software is that the defenders have all too large an attack surface. Compare this to a recent government report on cybersecurity R&D plans.[2] The report states a goal of achieving advances "to reverse adversaries' asymmetrical advantages" within 3 to 7 years. Crypto ransomware's cleverness might show that such a goal will be very difficult to achieve.

If you have occasion to do forensic analysis or recovery on crypto ransomware, or if you're trying to design countermeasures, it will be useful to know the span of options available to the malware writers and how they might be tripped up or deflected.

History

The first crypto ransomware was probably the infamous AIDS Trojan[3] in 1990. It was distributed on a floppy disk handed out to attendees at an international conference about the AIDS disease, and the software encrypted file names (not the files themselves), and then displayed a demand for payment to a location in Panama. The perpetrator's motivation might have been rooted more in a desire for revenge on the conference organizers than in financial gain, but in any case, the attack was ineffective. The exact reason for this wasn't published, but a program for restoring the file names was quickly distributed.

1089-7801/16/$33.00 © 2016 IEEE

A good guess about how the restoration could have worked illustrates the first principles of successful ransomware: it must be easily reversible, and it must also resist collusion.

The AIDS Trojan probably used the same key for all its encryptions. If someone paid the ransom, then the perpetrator, should he wish to preserve his reputation as a "fair" businessman, could tell the victim what the key was, perhaps by return post. Because the file names were encrypted with a symmetric cipher, it would be easy for the virus software to decrypt the file names when given the key.

But anyone who got the key, either by paying for it or guessing it, could simply tell everyone else, and the scheme would fall apart quickly. I suspect that the software did a very poor job of hiding the key, and that was the basis for the restoration program. It was unnecessary for anyone to pay the ransom.

This early effort didn't kick off a wave of imitators, even though the Internet was making malware virus distribution easier each year. There were a handful of virus programs that used encryption to render a machine useless and demand a ransom, but these used symmetric encryption and were easily undone because they used one key for all encryptions and didn't hide that key very well.

To turn crypto ransomware into a truly dangerous attack, there were two more pieces of technology needed. Asymmetric cryptography was one of them, and although it had been invented two decades earlier and was readily available through Pretty Good Privacy (PGP) software and the GNU Multiple Precision (MP) library, it didn't gain much traction with the malware crowd. This was odd, because in 1996, Adam Young and Moti Yung published a paper describing exactly how to do this.[4] Their method involved generating a unique symmetric encryption key for each infected computer and then encrypting that with a master public key embedded in the virus software. The beauty of their method was that the infected machine didn't need to communicate with the perpetrator until the ransom was paid. At that time, the victim could post the public key encryption of the symmetric key, and the perpetrator could decrypt that and send the symmetric key back to the victim for decryption of the files.

Malware authors didn't pick up on this scheme for about 10 years. Maybe they didn't trust the anonymity or security of the keys, or maybe they were making too much money from other schemes. Or maybe they were wary of collecting payments. Although scams taking advantage of international banking were common, ransomware faced more difficult hurdles to remain hidden. In an ordinary scam, the victims were unlikely to realize their mistake for several days, but with ransomware, the victims would be calling law enforcement immediately, and the bank account would be tracked or shut down quickly. To reliably evade detection, the perpetrators needed anonymous payment. In 1996 there wasn't much in the way of digital cash, but help was just around the corner. Block-chaining and BitCoin to the rescue!

How It Works

Cryptography isn't an absolute necessity for ransomware, but it's the only way to get close to an unbreakable denial-of-service extortion attack.

Nonetheless, social engineering and a well-chosen price point can make even non-cryptographic ransomware ("locker ransomware" or "lockerware") an effective tool. Lockerware will divert the computer from its normal operation by getting control of a critical resource, perhaps by encrypting and replacing that resource, and then displaying a seemingly unremovable view of a demand for payment. The demand might appear to come from a law enforcement agency. Some lockerware uses a simple Javascript technique to take control of a browser, again with a ransom demand. If the ransom is paid, the user should receive instructions on how to regain control of his computer or browser.

A particularly insidious way of installing lockerware is to offer a fake antivirus scanning program via a website. The website will pop up a window claiming to have discovered a virus on the visitor's computer screen and will offer a free detection program. The installed software is really malware that will lock up the computer and display an extortion demand. There are many other clever ways of getting users to install software from untrusted sources, but the fake AV trick is the one I think is truest to the ancient story of the Trojan Horse.

If the lockerware ransom amount is low enough, users might pay up rather than spending time searching for information or services to disable the malware. Disabling it might be time-consuming or obscure (like restoring an overwritten master boot record), or even impossible for the general user (as we'll see with Internet of Things devices). Even if the convenience of paying the extortionist seems like an attractive option, victims should be extremely wary of paying it, because there's no guarantee whatsoever that the machine will actually be unlocked.

Several years after the Young and Yung paper, public key ransomware turned up in Russia.[5] There were some fears that the malware had unbreakable cryptography, but the early versions were still primitive things with symmetric ciphers and embedded keys. As with any disruptive technology, it took some years to refine it into a reliable, profitable, worldwide operation. Besides the necessary software engineering skills and an easily usable payment method, businesses need distribution networks, knowledge of optimal price points, revenue-sharing arrangements, and a reliable-yet-anonymous Internet presence. The shadowy figures behind ransomware kept building up their business components, and the industry seemed to reach some kind of fruition a few years ago. Today, most people know about ransomware and probably know someone who was affected by it.

By 2009, crypto ransomware had entered the public key cryptography arena in force, and its use is increasing rapidly. Unlike lockerware, there's no simple way to restore a critical resource and regain normal operation. The computer's files, accounting data, document drafts, contact lists, and so on — all have been transformed into encrypted data and only the encryption key will undo the damage.

From a technology perspective, successful ransomware must meet a handful of critical requirements.

1. Some resource that's valuable to the user must be made unavailable (denial of service).
2. The denial of the resource and the payment instructions must be announced to the user of the afflicted machine in an unavoidable, visible process.
3. The ability to restore the valuable resource must depend on a small amount of data that's available only to the extortionist and can't be inferred or calculated by any other process at reasonable cost.
4. The extortionist must be able to verify payment.
5. The extortionist must be able to accept payment and supply the information for restoring the resource without identifying himself.
6. The restoration process must run on the afflicted computer, it must be simple to use, and the restoration must be reasonably reliable.

Public key cryptography provides the means for achieving requirement, as noted in the Young and Yung paper. However, the only means of getting the strictest sense of "can't be inferred or calculated" would limit the ransomware to a painfully slow public key encryption method. Most ransomware trades off some security for performance, and this gives it the ability to encrypt more of the user's file data before being detected.

Symmetric Keys Only

Apparently the simple way of using symmetric encryption to enable unbreakable ransomware was never used, but it deserves some consideration in the taxonomy of techniques. There are no public keys in this method, and it illustrates the design options open to ransomware developers.

If each instance of the virus used a unique symmetric encryption key for its dirty work, and if it destroyed that key after using it, then file recovery would be nearly impossible. The only problem is that the extortionist must know what that key is in order to release the victim's files. Thus, the victim's machine has to hold some piece of data that that lets the extortionist know which key was used for that victim. Somehow, there must be communication between the extortionist and the victim's machine.

The malware can initiate that communication prior to beginning its work, or it can be done when it finishes encrypting. In the former case, the malware contacts the extortionist and receives a symmetric encryption key and a key identifier. In the latter case, the malware generates a random symmetric key and a random identifier and sends those to the extortionist. In both cases, after encrypting the files, the malware destroys the encryption key but retains the key identifier. If the victim pays the ransom and communicates the identifier to the extortionist, the extortionist will be able to send the corresponding encryption key.

If the victim's machine isn't connected to the Internet, then this attack might fail to get started, or it might fail to leave any way for the victim to recover his data. After the symmetric key is erased, we can only hope that the extortionist actually has the key and the key identifier!

Although this scheme is at the core of all crypto ransomware, as described here it has a serious flaw. Anyone who observes the communication between the malware and the extortioner will be able to see the symmetric key. It might show up in logs of network traffic, either locally or on a network monitor in the communication pathway. However, if the victim has no access to the messages, the method is quite sound.

Embedded Master Public Key

By using public key cryptography, ransomware can avoid the necessity of communicating directly with the extortionist. This is by far the simplest way of implementing ransomware. The method is similar to that in the previous section, but with a crucial difference: the malware has the extortionist's public key embedded in its software.

The malware begins by generating a random key for symmetric encryption. After encrypting the victim's files, the malware uses the embedded public key to encrypt the random ransom key. If the malware leaves no trace of the symmetric key, then the encrypted random key serves the job of the key identifier. After paying the ransom, the victim sends the encrypted key to the extortionist or publishes it in a pre-agreed place. The extortionist will use his private key to unlock the random symmetric key, and he can send it to the victim or publish it in a pre-agreed place.

This method has only two drawbacks. One is that the symmetric key might be visible if a suspicious victim dumps memory while the encryption is active. The other problem is that should the extortionist somehow leak the value of the private key, then all victims could use it to recover their data by decrypting their locally encrypted symmetric key. In fact, one extortioner ended his scheme by publishing the private key.[6] Perhaps these people sometimes experience remorse.

A Unique Public for Each Malware Instance

By adding one roundtrip message, ransomware can avoid the reliance on a single public key. Although most ransomware uses public keys that can't be "broken" in any reasonable computing scenario, still, one public key is only one layer of protection for the extortioner.

If the malware sends a request message to the extortioner's message service, such as a compromised website providing anonymity for the criminals, then the command and control center for the ransomware can send back a freshly computed public key. The malware on a victim's computer will encrypt the symmetric key using the public key. The public key itself serves as the identifier to use when paying the ransom. The extortioner's software will find the matching private key and send it to the victim.

An interesting variant on this method allows the malware to avoid using symmetric encryption. The symmetric methods have a point of vulnerability in that they have to keep the symmetric key in memory for the entire time that the user's files are being encrypted. If the process is interrupted, an examination of memory might reveal the key.

By using public key encryption, the malware will incur a huge time cost penalty. The user might detect that infection before many files are affected. However, the public key encryption methods will yield no useful information about decrypting the files. Only the matching private key, held by the extortioner, can undo the damage.

The Ransom Payment

Bitcoin or other anonymous payment systems protect the extortionists by moving the ransom money to them without identifying their bank accounts or location. Although the systems aren't perfectly anonymous, the money can move quickly enough through cooperating "laundering" sites to thwart law enforcement.

In a recent twist, the malware designers have found a way to use the cash transactions for a second purpose. The key that unlocks the victim's files, be it a symmetric key or a one-time private key, can be part of the transaction that pays the ransom. Bitcoin's block chain supports auxiliary transaction information, which is perfect for moving the key identifying information to the criminals and for letting them publish the symmetric or private key that unlocks the victim's files. The victim can attach the encrypted key blob or its identifier to the ransom payment, and the extortioner then puts the unlocked key into the transaction chain.

Other methods of delivering the decryption key are used. The ransomware can, for example, poll a command and control server. When payment is complete, that server will return the key to the victim's machine where, with any luck, the decryption will be completed quickly.

I haven't found any description of the methods used to verify payment and release the key. This must be a manual process, requiring the extortioner to communicate with a command and control server or to post the information in a public place. If law enforcement could infiltrate those processes, they might be able to release the data that unlocks the victim's machines.

Attacking New Platforms

Scott Adams' *Dilbert* cartoon on 12 May 2016 had the caption "My smartwatch was infected with ransomware" (http://dilbert.com/strip/2016-05-12). I laughed when I saw that, but experts warn that smartwatches are entirely hackable.[5] In fact, they're the harbingers of the world of smart and insecure wearables. The only saving grace is that these devices don't hold much data, and thus a factory reset should restore functionality.

While the attacks on digital accessories seem amusing now, the devices inexorably will acquire new features and importance in daily life. Our cellphones are becoming the linchpins of personal identity, reminders, and the way we contact other people. Unless we take care to provide offline storage for all this data, a ransomware attack could be devastating.

The major operating system providers take steps to insulate the various apps from one another's data, and this makes a complete takeover of a smartphone through a single compromised app unlikely. Nonetheless, all software has bugs, and a zero-day attack against a mobile OS kernel is sure to surface from time to time.

We can only hope that the designers of these gadgets realize their vulnerabilities and make sure that any essential data the gadgets hold is backed up with guards on the data's integrity and that it can be easily restored.

Offenses, Defenses

You're probably thinking that file backups are a simple way to defend yourself from ransomware. That's a good way to begin thinking about proactive measures, but the ransomware writers are way ahead of you. Unless you have a backup system that keeps copies of data offline and doesn't overwrite data for several weeks, you might still be vulnerable to ransomware. The malware designers methodically seek out backups, be they on the local machine's storage, on a shared file server, on a removable device, or in a cloud service.

When an afflicted machine has the ability to overwrite files on a shared server, all the files on the server are vulnerable to the crypto malware. Even one infected machine can destroy the files of a small business, for example.

Without a detailed understanding of how his files are backed up, a user might be at the mercy of ransomware. Some users have been dismayed to discover that their backups contained the encrypted files. This happens because when a file encryptor causes the file contents to change, backup system will notice the new version and will save it. To restore the unencrypted data, the user needs access to a backup that has been inaccessible to the malware and was written shortly before the malware began its work.

The unpleasant truth is that users need to understand their backup service in some detail before declaring themselves safe from crypto ransomware. They need to think about their backup service in terms of resilience from a concerted attack. When is a full backup done? Can it be deleted or overwritten without the user's explicit permission? How often are incremental backups done? Can they be deleted or overwritten?

Website administrators are usually at less of a risk, even though there's ransomware that targets them through http. The website content is usually stored on servers that aren't part of the website itself, and the content is uploaded to the servers. If the servers are hacked, the content can be easily restored from its normal repository.

As several people[7] have pointed out during the ongoing debates about encryption policy, almost all software has exploitable bugs, and ransomware is no exception. I would guess that given enough time, most skilled security firms could break any ransomware. The keys might be inadvertently exposed in the software, the public keys might have a lot of bits but be badly chosen, the encodings might leak data, the key generators could be faulty, or the command and control servers might be hackable.

Symantec researchers partially agree with that assessment when they state the following:

> *But even with improved encryption, some recent ransom schemes are still not always water tight. Poor operations and procedures dog the efforts of cybercriminals, leaving victims with room to maneuver. Even today, some still continue to make rookie mistakes such as leaving behind keys. This suggests that the current ransomware scene is highly fragmented with many new actors trying to establish themselves in a market already dominated by small groups of professional cybercriminals.*[5]

But most people don't have the luxury of doing without their data while the experts investigate. Paying the ransom might be the only practical solution. Further, there's reason to suspect that the skill level of ransomware developers is rising. Detailed examinations of two examples, zCrypt[8] and Maktub,[9] reveal sophisticated methods for evading detection while they encrypt files. Incidentally, zCrypt uses public key encryption on files and is therefore very slow. Strangely, it doesn't compensate by using Maktub's trick of compressing the files before encryption.

If ransomware continues its path toward a hardened, almost foolproof implementation, new methods of protection might be brought into play. The operational characteristics of encryption processes could be used against it. For example, the repetitive loop of the AES cipher could be detected by runtime execution monitors. The same is true of the large number of multiplications that RSA entails. Moreover, an encrypted file is radically different from a non-encrypted file. Most notably, the number of zeros and ones will be almost the same for an encrypted file, but ordinary files are unlikely to have such an even distribution. So theoretically you could devise an execution monitor that randomly sampled instruction traces in real time, and if encryption was happening in anything other than SSL or other "authorized" encryption program, the monitor would look at its open file descriptors to see if it was writing "gobbledygook" into an ordinary file.

The people behind ransomware seem to have a good grasp on a dangerous technology, and they've turned it into a profitable business. Although its delivery method is usually the antiquated trick of hiding malware in an email attachment, this remains effective and catches millions of people each year. Ransomware is becoming so notorious that one of the inventors of public key cryptography has said he feels like a parent whose child has become a terrorist.[10]

The cleverness of ransomware should be countered by a three-pronged approach. First, the delivery of malware through email attachments should be stomped out through better operating system protections on the major OSs. Second, backup services should specifically address ransomware through better retention times and protection from being written over or deleted by malware. And finally, the integrity of file system data should be the subject of more development. Malware shouldn't be able to write files.

Until the majority of computer systems (and that includes mobile devices) have these protections built-in, the ransomware industry seems likely to flourish. ▯

References

1. Sun Tzu, *The Art of War*; http://classics.mit.edu/Tzu/artwar.html.
2. Subcommittee on Networking and Information Technology Research and Development (NITRD), *Federal Cybersecurity Research and Development Strategic Plan: Ensuring Prosperity and National Security*, tech. report, US National Science and Technology Council (NSTC), 2015; www.whitehouse.gov/sites/whitehouse.gov/files/documents/2016_Federal_

Cybersecurity_Research_and_Development_Stratgeic_Plan.pdf.

3. K. Laffan, “A Brief History of Ransomware,” blog entry, *Varonis*, 10 Nov. 2015; https://blog.varonis.com/a-brief-history-of-ransomware.
4. A. Young and M. Yung, “Cryptovirology: Extortion-Based Security Threats and Countermeasures,” *Proc. IEEE Symp. Security and Privacy*, 1996, pp. 129–140.
5. K. Savage, P. Coogan, and H. Lau, *The Evolution of Ransomware*, version 1.0, white paper, Symantec, 6 Aug. 2015; www.symantec.com/content/en/us/enterprise/media/security_response/whitepapers/the-evolution-of-ransomware.pdf.
6. P. Ducklin, “TeslaCrypt Ransomware Gang Reveals Master Key to Decrypt Files,” *Naked Security*, *Sophos.com*, 19 May 2016; https://nakedsecurity.sophos.com/2016/05/19/teslacrypt-ransomware-gang-shuts-up-shop-reveals-master-key/.
7. S. Bellovin et al., “Lawful Hacking: Using Existing Vulnerabilities for Wiretapping on the Internet,” *Northwestern J. Technology and Intellectual Property*, vol. 12, no. 1, 2014; http://scholarlycommons.law.northwestern.edu/njtip/vol12/iss1/1.
8. MlwrHpstr, *ZCrypt Ransomware: Under the Hood*, blog, Malwarebytes Labs, 14 June 2016;https://blog.malwarebytes.com/threat-analysis/2016/06/zcrypt-ransomware.
9. Hasherezade, *Maktub Locker – Beautiful and Dangerous*, blog, Malwarebytes Labs, 24 Mar. 2016; https://blog.malwarebytes.com/threat-analysis/2016/03/maktub-locker-beautiful-and-dangerous.
10. J. Schwartz, “RSA Encryption Inventors Lament Its Use for Ransomware,” blog, *Redmond Magazine*; 22 Apr. 2015; https://redmondmag.com/Blogs/The-Schwartz-Report/2015/04/Encryption-Ransomware.aspx.

Hilarie Orman is a security consultant and president of Purple Streak. Her research interests include applied cryptography, secure operating systems, malware identification, security through semantic computing, and personal data mining. Orman has a BS in mathematics from the Massachusetts Institute of Technology. She’s a former chair of the IEEE Computer Society’s Technical Committee on Security and Privacy. Contact her at hilarie@purplestreak.com.

Rejuvenating Binary Executables • Visual Privacy Protection • Communications Jamming
IEEE SECURITY & PRIVACY
Building Dependability, Reliability, and Trust
SOFTWARE EVERYWHERE
January/February 2016 Vol. 14, No. 1
Policing Privacy • Dynamic Cloud Certification • Security for High-Risk Users
IEEE SECURITY & PRIVACY
IEEE Symposium on Security and Privacy
March/April 2016 Vol. 14, No. 2
Smart TVs • Code Obfuscation • The Future of Trust
IEEE SECURITY & PRIVACY
ECONOMICS OF CYBERSECURITY PART 2
May/June 2016 Vol. 14, No. 3
IEEE computer society
CELEBRATING 70 YEARS
IEEE Reliability Society
S & P