Why Encryption Didn't Save TJX

March 31 2007

TJX: It's the target of the largest known customer record theft of all time, and it's a case in point that encryption is not a silver bullet.

This is the heart of the encryption problem , quoted from the 10-K filing The TJX Companies made to the Securities and Exchange Commission:

"Despite our masking and encryption practices on our Framingham system in 2006, the technology utilized in the Computer Intrusion during 2006 could have enabled the Intruder to steal payment card data from our Framingham system during the payment card issuer's approval process, in which data (including the track 2 data) is transmitted to payment card issuer's without encryption. Further, we believe that the Intruder had access to the decryption tool for the encryption software utilized by TJX."

Encryption has no value when data isn't encrypted, obviously, but credit cards can't be processed when their numbers are encrypted. Hence, a smart crook will seek a way to get the data during that window of time when it's in that state of being "in the clear" - that is, unencrypted.

TJX's intruder also had a backup plan if data in the clear wasn't attainable: namely, the decryption key.

There are several reasons why encryption didn't save TJX and won't save many companies, regardless of how much legislators have mandated or want to mandate its use. (One example of which is the June 2006 White House mandate requiring federal agencies to encrypt the hard drives of all their laptops and mobile devices.)

In an interview with eWEEK, McAfee Chief Security Officer Dr. Martin Carmichael said that after he had read TJX's take on the intrusion, he can say that it's plain that encryption was involved, but not what kind of encryption: shared key, in which the sender and receiver of encrypted data both have the same key, or asymmetric, which uses a public/private key pair.

Shared-key encryption is inherently risky, since humans think up convenient but absurdly insecure places to store their keys. "We have seen … some companies that chose to use shared-key - encryption - that stores the key with the data," Carmichael said. "Which is outside of most policy. Sometimes ease of development can be - counter to - good security process." In fact, Carmichael has seen keys in data files that are named "key to data."

Another encryption trap is the use of weak encryption. Original DES (Data Encryption Standard) encryption is now considered to be insecure for many applications, chiefly due to its 56-bit key size being too small. DES keys have been broken in less than 24 hours. Some analytical results point to theoretical weaknesses in the cipher, as well, although those have not been proven in practice. In May 2002, DES was superseded by AES (Advanced Encryption Standard) following a public competition, but DES remained in widespread use as late as 2004; Carmichael said it was "very common in a lot of applications."

Did TJX use DES? TJX has determined that its data was first accessed by an unauthorized intruder in July 2005, and DES was widely used in 2004, so it's imaginable that the company did.

Asymmetric cryptography gives part of a key to the data sender and part of the key to the data receiver. The receiver of data - for example, a bank that's receiving your bank account number or user name and PIN - can publish what's called the public part of the key to the whole world. The only thing that encrypts data, however, is the private part of the key. You as a bank customer can contact your bank using one part of the key, and the bank can match that up with its part of the key, thus having an encrypted session with two different keys.

This type of public/private key cryptography is used because key distribution is a major problem, Carmichael said. Shared keys have to be stored somewhere. They can be unsecure, no matter where they're kept.

Those who use public/private key cryptography have the private key stored in a "very special place," Carmichael said - a certificate server that's hardened and secured.

Did the TJX intruder stumble on a key stored with the encrypted data, a la shared-key cryptography, or did the intruder have access to a certificate server? The question is moot, given that the intruder figured out a way to take the data before it was encrypted, but the details of nabbing an encryption key will be instructive if we discover them as TJX's investigation continues.

And so that leaves us with asymmetric, aka public/private, key cryptography. Is it safe to consider that form of encryption a silver bullet? Definitely more so than shared-key encryption, but it's a bullet that can backfire.

Ted Julian, vice president of strategy for Application Security, said in an interview with eWEEK that the practical issue for customers contemplating encryption is that the technology always has performance overhead. This, in fact, is a common deal breaker, he said.

The reason for the performance hit is that so many applications use sensitive data as an index field. One example Julian offered was the formerly common practice of using a college student's Social Security number as a college ID number. To look up any information about a given student meant queries had to be run against that one index field, whether it was looking up grades or tuition payment records.

But given that that field contains a sensitive piece of data - a Social Security number - that index field is also the field an organization will eventually want to encrypt. Once that happens, Julian said, the system will be brought to its knees.

"I don't care if it's native encryption in an Oracle 10gR2 database or not," he said. "It will be untenable."

To change that scenario, you'd have to change all the applications so that they use a different index field. That's a lot of work. And there's no guarantee that that work won't break applications.

Another equally important issue with public/private key encryption has to do with architecture. The considerations range from how strong the keys are, to where they're stored, to who gets access. "Not that any of those require a roomful of rocket scientists to figure out, but it takes expertise, and you have to make sure you get it right," Julian said. "You have to test it in the lab, have to make sure it's working effectively, have to get involvement from multiple parts of the organization to make sure it's in line with security policy, - and so on - ."

Then again, there's the question of application impact. Applications that once handled data that only ran in the clear now have to handle ciphered data. That kind of load change can "quite possibly" break applications, Julian said. An application that wasn't expecting to get a large quantity of data back could easily suffer a buffer overflow.

"They'll slow down, but you don't know until you build a trial version and do tests in the lab," he said. "You use a simulated production environment, see how it's working and slowly roll out an application. It's a six- to 12-month process for sure."

And that time, Julian said, is only for one application. One that might break under the strain, to boot.

For those organizations trying to figure out whether to use encryption or how to avoid becoming another TJX - or both - there's hope. For a fraction of the time it would take to set up encryption, an organization could do far more for its security by doing a database vulnerability assessment and setting up active database monitoring, Julian said.

The assessment would include looking for default IDs and passwords that might still be present, for example, Julian said - a situation that's far from rare. It would also include looking for known vulnerabilities, patching them and hardening the databases against attack. Just there, Julian said, an organization can "make material improvements in a single day."

Monitoring database activity will alert an organization not only to people who are trying to attack a database but also to trusted individuals performing unwarranted actions. Even a DBA, for example, should never do a select-* on a credit card number column.

Those two steps - database assessing and monitoring - could "enormously improve the security posture of a database," Julian said, "and you haven't even started with crypto. You're still talking about it."

None of that apparently helped TJX. "Nothing's foolproof, to be clear, but in this case it sure appears that monitoring would work," Julian said. "You'd think 47.5 million credit cards would show up on your screen. If you were watching."

Citation: Why Encryption Didn't Save TJX (2007, March 31) retrieved 19 April 2024 from https://phys.org/news/2007-03-encryption-didnt-tjx.html