Privacy Innovations

At the time of the writing of this article, the SolarWinds supply chain hack[1] is the big news within information security. Due to the sophistication of the attack and the follow-up, information security researchers believe a nation-state actor, an advanced persistent threat (APT), developed and executed the hack.

The reality is that every organization is always at risk for an attacker to compromise third-party software each uses, rendering them vulnerable. It might not be software but a process. I can remember multiple hardware vendors with compromises to their base workstation image. Malware made its way into the image and computers shipped, already infected. These cases are intrinsic risk scenarios and organizations must plan for them. Individual organizations may not be tangling with an APT, but a simple phishing attack with a sequence of unfortunate events could result in a similar compromise.

K. Brian Kelley, CISA, CSPO, MCSE, Security+
Is an author and columnist focusing primarily on Microsoft SQL Server and Windows security. He currently serves as a data architect and an independent infrastructure/security architect concentrating on Active Directory, SQL Server and Windows Server. He has served in a myriad of other positions including senior database administrator, data warehouse architect, web developer, incident response team lead and project manager. Kelley has spoken at 24 Hours of PASS, IT/Dev Connections, SQLConnections, the TechnoSecurity and Forensics Investigation Conference, the IT GRC Forum, SyntaxCon, and various SQL Saturdays, Code Camps and user groups.

Most of the time, the adversary wants information, of which privacy data are a subset. As a result, there has been significant innovation in data protection over the last decade. However, products will not do everything for us. When we are looking at our own systems, we have to plan to protect privacy data. As the studies have shown, the sooner in the process we start, the less costly it will be long term. That means we have to do our own "baking."

"Baking In" Privacy Protection

A complete compromise will not stop the adversary from getting everything, given enough time. What we are looking to do, then, is make it as difficult as possible to extract information while ensuring that our own systems are not negatively impacted. Usually this means that if you access the data with the correct application and the correct procedures, you get to the data easily. If you try to extract the data via another mechanism, there is more time and effort involved.

The more effort and the more time, the more likely it is that we are going to spot the adversary. That is the goal. However, if we are "bolting on" products to address gaps, it is never as effective as innovation early on. We may be using the exact same products, but by addressing the privacy risk early in the cycle, we are able to more fully leverage these products and, therefore, get the most out of them. We should be able to reduce the overall risk.

When we start planning early on, we call that "baking in" the solution. When it comes to developing solutions for privacy data protection, there are several key areas that immediately come to mind:

- Encrypting data at rest
- Encrypting data in flight
- Protecting against a privileged user attack
- Auditing every relevant action

Encrypting Data at Rest

When we talk about encrypting data at rest, we usually mean the files themselves. Plenty of

solutions can encrypt the files transparent to the systems that use them. Copy the files off without the appropriate keys, however, and you cannot read them. That is, if you can even copy them off. Some systems block such operations unless you have the key to unlock.

When I think of data at rest, though, I think we need to go beyond that simple definition. For instance, when a Relational Database Management System (RDBMS) opens its data files and has access to the data, unless you have access to the proper keys, can you view the data? If the answer is yes, then while technically this does not meet the definition of data at rest, it is certainly not data in flight. I would rather not coin a new phrase, but perhaps "data on the runway" might be appropriate, given the analogy to flight?

In any case, we do have to consider data in this state, too. For instance, some platforms leave the data encrypted on disk, but once read and/or modified, the data are kept in memory in an unencrypted state. Technically, when you consider swap files, flash drives and other mechanisms of management involving storage, the data might end up on said storage, even if the intent is to keep the data in memory. In this case, the data would be unencrypted.

For instance, a database platform might encrypt the overall file. However, if the process can access the file and decrypt it, the data within the file are effectively unencrypted to the platform. It is this unencrypted version of the data that goes into the memory buffers to speed up database performance. This differs from the case that within the file itself, say at the column or cell level, the data are encrypted. In this particular case, the database platform puts the data in memory as they exist within the database file, which is encrypted at the column or cell level. Should we have a page operation to write to the swap file, etc., the encrypted version would be what is written to disk. I have used the example of the database platform, but the application can also be guilty of caching unencrypted data in memory.

Knowing this risk, if we can design solutions that minimize the time the data are accessible in an unencrypted fashion, all the better. For instance, in the Microsoft-centric realm, using Always Encrypted with secure enclaves and Structured Query Language (SQL) Server 2019[2] will protect the data in memory on the database platform. On the application side, .NET Framework, .NET Core, and .NET 5 have Data Protection to encrypt a file or stream (such as one residing in memory).[3]

Encrypting Data in Flight

On enterprise systems, when data are consumed by an application, the majority of the time the data are transmitted over the network. There are cases where the data source and the consuming application are on the same system and the transmission is in memory, but these represent the exception rather than the rule. Something to consider: If the data go across the network unencrypted but the network engineers do not have access to the data through a system or database platform, we have an issue. We expect the network engineers to be able to look at the traffic across the network. Given that most platforms use well-defined protocols and many packet analysis tools understand those protocols and can break them down automatically, that means it is entirely possible for a network engineer to see data in this manner to which he or she should not have access.

Therefore, it is important to think about how the data will traverse the environment. Sometimes, creative solutions are required, especially if the application or back-end platform itself does not support any sort of encryption. There are other solutions available. For instance, at the network layer, between the two endpoints, using Internet Protocol Security (IPSec)[4] might be the best answer. As far as the system is concerned, nothing has changed. However, when looking at the network traffic, the packets show that, at a minimum, the payloads are encrypted using IPSec.

Protecting From Admin Access

Administrators, as expected, typically have complete rights over the system. Protecting against them seeing the data is extremely challenging. Most of the time, bolted on solutions cannot work. Therefore, protecting privacy data has to be "baked in." Some of the solutions already mentioned provide some capabilities to protect against administrator access. For instance, the combination of Always Encrypted with secure enclaves as well as protected memory at the application layer implemented properly prevents an admin from scanning the memory for unencrypted data.

Enjoying this article?

- Read *A Global Look at Privacy: ISACA Research Report. www.isaca.org/global-privacy-2020*
- Learn more about, discuss and collaborate on privacy in ISACA's Online Forums. *https://engage.isaca.org/onlineforums*

However, unencrypted data can end up in many unexpected places. Years ago, I was working on a human resources (HR) system with the database credentials in plaintext in a text file named after the database vendor. While this was not directly allowing access to unencrypted privacy data, it indirectly did. The platform did not encrypt the data within the database. Therefore, if you could get the database credentials, you were able to access the data. We ended up coming up with a solution involving file permissions and auditing on the file itself as mitigating controls because we were not changing the application.

Oftentimes, teams will have to innovate to figure out ways to either block admin access or report when an admin accesses the data. That audit trail is important. Speaking of audit trails, they may be the only control we have available. Specifically, they serve as a detective control to protect the data.

"Baking In" Appropriate Auditing

About a year after pop star Michael Jackson's death, a report indicated that someone had improperly accessed his medical records.[5] This is a clear violation of US Health Insurance Portability and Accountability Act (HIPAA) and was one of a series of privacy breaches for Ronald Reagan UCLA Medical Center. The system likely recorded the improper access in an audit log, which enabled the medical center to address the issue.

I have seen cases where auditing was added to a system after implementation. In those cases, the development teams had to become extremely innovative to reach a workable solution. I have also seen cases where auditing could not be added. Systems such as the one that was likely in use at Ronald Reagan UCLA Medical Center build the proper auditing in from the start. We never want to be in a situation where we have to have better auditing, but we are not sure if it is possible without a new system or a major or even complete rewrite of an existing one. This brings us back to baking in privacy protection.

There is a fine line between not enough and too much. We must determine where that line is as early in the process as possible, which is another reason to consider how to add auditing from the start. Through the various test cycles, different use cases should reveal whether or not the system is capturing enough data and show when the system is recording too much information, resulting in unwanted noise. While security information and event management (SIEM) systems can filter out most of that noise, we do so at the expense of compute resources, which also means time. Depending on the sensitivity of the privacy data, this may delay our alerting for cases that require attention.

The Road Forward

Attacks are going to continue to improve in sophistication. Data protection products are going to continue to roll out with new features. As new attacks develop, we need to innovate new mitigation strategies and steps. The best scenario is when we can take our new methods and put them into the cycle at the beginning. Otherwise, there are likely to be gaps we just cannot fix or resolve. We need to consider how to protect privacy data both at rest and in flight, limit administrators' access to the data, and have proper audit trails in place. While we might not be able to prevent all attacks, we will hopefully reduce their effectiveness and duration.

Endnotes

1 Cybersecurity and Infrastructure Security Agency, "Active Exploitation of SolarWinds Software," 14 December 2020, *https://us-cert.cisa.gov/ncas/current-activity/2020/12/13/active-exploitation-solarwinds-software*

2 Microsoft Documentation, "Always Encrypted With Secure Enclaves," 31 October 2019, *https://docs.microsoft.com/en-us/sql/relational-databases/security/encryption/always-encrypted-enclaves?view=sql-server-ver15*

3 Microsoft Documentation, "How to: Use Data Protection," 14 July 2020, *https://docs.microsoft.com/en-us/dotnet/standard/security/how-to-use-data-protection*

4 Thomas, J.; A. J. Elbirt; "How IPsec Works, Why We Need It, and Its Biggest Drawbacks," *CSO*, 6 January 2004, *https://www.csoonline.com/article/2117067/data-protection-ipsec.html*

5 Hennessy-Fiske, M.; "Michael Jackson's Medical Records at UCLA Were Improperly Accessed, Source Says," *The Los Angeles Times*, 10 June 2010, *https://latimesblogs.latimes.com/lanow/2010/06/michael-jacksons-medical-records-at-ucla-were-improperly-accessed-source-says.html*