The Pervasive Encryption Imperative

IBM Competitive Project Office
Mark Moore
Senior Software Engineer

June 2017

Contents

Introduction

Data breach may be the most pressing problem facing modern business. Some businesses recognize the existential threat data loss poses and attempt to mitigate it. Others are required by industry regulation to provide protection for data they handle. As technology becomes ever more invasive and people give up personal information – sometimes willingly, sometimes unwittingly – in exchange for convenience, governments are beginning to pass more comprehensive regulation to protect the public.

One far reaching example is the European Union General Data Protection Regulation (GDPR), which will go into effect in May of 2018. The regulation covers all collectors and processors of EU resident data. If such an entity experiences data breach, those entities must report the breach to the supervisory authority in its member state. If the authority determines there is an adverse effect to a breach, affected parties must be informed. Sanctions can be imposed.

There is an exception to the reporting rule. If the data is pseudonymized so that it cannot be associated with any individual, the breach does not have to be reported. Encryption is an excellent way to pseudonymize data. Therefore, encryption is an excellent way to address reportable data breaches under this new regulation.

Pervasive encryption to address a pervasive threat

Data in a solution transitions through many states. It may be at rest on a disk. It may be in flight to a client application. It may be between nodes in a high-availability cluster, or waiting to be of use in a swap space.

To be truly pervasive, encryption must be applied to the data in all of its states. One may wonder, “Who could be breaching my cluster network?” A flippant answer would be, “I don't know and neither do you.” The serious answer is that threats come both from the outside and from within. Along with all of the functional requirements of a solution that directly address daily business come the non-functional requirements that address the operation of that solution. Administrators have access to hardware and software that, from a security perspective, it would be preferable for them not to have. When an IT specialist is monitoring cluster traffic, is that maintenance or a breach? The activity is the same. The difference is intent. If the activity is a breach but the data is encrypted, it is not a reportable breach.

Just considering data at rest, self-encrypting drives or Linux's LUKS, for example, would not be sufficient to avoid reportable breaches. One must consider that the disks are unlocked when in use, leaving data exposed both to a black-hat hacker who might gain a toe-hold on the server, or a rogue administrator with root access.

If full-disk encryption is not a panacea, one has to consider encryption scoped to the sensitive data itself. With that, a burden is created for application developers to keep track of and encrypt sensitive data not only in databases, but also log files and other data sets. Every code change is an opportunity to leak data. More granular encryption could lead to more difficult audits.

One final concern is the impact to service level agreements. Encryption is computationally expensive. The initial application of encryption may require service interruption.

Starting from scratch

Starting from scratch with commodity servers may seem to be a way to address reportable data breaches cheaply. After all, x86 processors include support for the math associated with cryptographic functions in the form of AES-NI. Linux is a fine operating system. As referenced above, Linux has full-disk encryption in the form of LUKS. Individual directories can be encrypted using the Linux tool eCryptfs. Swap can be encrypted. Solutions can use SSL to communicate with clients. Many enterprise databases support encryption. At first glance, that may seem to check the boxes on the larger requirements. However, there are a number of challenges to that approach.

Booting a server with LUKS requires unlocking the disks. Without key management, that is probably a manual process, but that means any administrator with the passphrase can unlock the disk for both legitimate and nefarious reasons. A keyfile might be used, but that is not secure, as it makes the key available to any administrator with access to the file.

Even with full-disk encryption, via LUKS or in in the disk itself, individual data stores must still be encrypted against rogue administrators with root access. eCryptfs would seem to fit the bill, but unlocking the store would have downsides similar to LUKS. Additionally, a solution based on this approach would require rigorous development to ensure all data is stored in the correct location. Guarding against leaks to unsecured space, like writing sensitive data to a log, would require careful review at every step. Also, encrypting a directory with eCryptfs means data must be copied away, the directory secured, and then data copied back so that it can be encrypted. This will likely lead to service interruption.

Because attempting compliance in this way involves a number of difference Linux features, administrative procedures, and disciplined solution development, audit could be difficult, lengthy, and expensive.

Bolting-On Security

Understanding that addressing reportable data breaches with a commodity Linux server is difficult, at best, one might consider bolting on a security solution with a Hardware Security Module (HSM).

An HSM would solve the problem of key management. The governing standard for security modules is a U.S. Government standard, FIPS 140-2. It defines four levels of security. They range from level 1, which has no requirement for physical security mechanisms, to level 4, which provides a complete envelope of protection, detecting and actively thwarting attacks. Most HSMs implement the standard at level 2 or 3, where physical security and countermeasures against obvious physical attack are specified.

Bolt-on security software that provides “transparent encryption” can enable administrative staff to create secured, encrypted areas and leverage an HSM to unlock those areas just for particular applications or users. This eliminates the risk of a rogue administrator with root access gaining unauthorized access to that data. Another advantage over the do-it-yourself approach is there is a single point of reference for audit. With that in mind, there may service interruption as above with eCryptfs.

However, a number of problems remain with this approach. An application's access to a secured area is no assurance that the secured area will be used exclusively. As noted above, rigorous development

habits, code review, and audits would be required to ensure against data leakage. Sensitive data may be leaked through simple logging. Core dumps created when an application crashes may, too. All of these things may or may not be written to secure areas. In a worst case, a developer may become the rogue threat and leak data intentionally.

In addition to those concerns, these types of bolt-on security solutions concern themselves with data at rest. Locking down data that may flow between tiers of a solution, ensuring that communications flowing in and out of a server are secured to the required standard, and more are unaddressed.

Pervasive encryption with IBM Z

IBM Z provides a complete pervasive encryption capability for data both at rest and in flight. This capability begins with its hardware and follows through to the moment the data leaves the server.

- **Superior hardware support for cryptography:** Each processor in IBM Z has a coprocessor called a CPACF (Central Processor Assist for Cryptographic Functions). It performs a set of full symmetric cryptographic and hashing functions. x86 AES-NI only supports the math for those functions, which are performed in software. Some additional acceleration comes from Intel's Platform Controller Hub (PCH), but there is only one, not part of the processor, and busy with many other functions like networking and device I/O.

  IBM also offers an HSM called Crypto Express. This card is certified to FIPS 140-2 level 4 – its highest level. It provides cryptographic operations above and beyond CPACF as well as key storage. The unique relationship between Crypto Express and CPACF on IBM Z enables a feature called "Protected Key". A key secured in the Crypto Express HSM can be unwrapped within the card and then re-wrapped so only a particular IBM Z partition can use it. That key is then used by a CPACF to perform cryptographic functions for that partition. The underlying clear value of the key never exists in any address space, but only exists inside the secure hardware or the CPACF.

  So, where Crypto Express and CPACF have a special relationship with each other through the firmware of IBM Z, bolt-on solutions for commodity hardware does not. The HSMs are less secure, with lower FIPS 140-2 or 3 ratings. The keys must pass through operating system address space to be used. Cryptographic functions are performed, to a large degree, in software.

- **Dataset encryption by Policy:** Where the commodity server-based solutions require rigor to avoid data leakage, IBM Z can encrypt datasets as a matter of policy, requiring no further attention from administrators or developers, and without service interruption.

  This stands in contrast to securing particular areas on commodity servers, requiring developers to be diligent to use them and auditors to verify that continues to be the case.

- **Database encryption:** zTPF (Transaction Processing Facility) database can enjoy transparent encryption, applied without service interruption. Even data cached in memory is encrypted.

- **Full disk encryption:** The IBM Z can serve keys from the HSM to unlock disks in DS8000 storage systems. Here again, functionality and security is enabled by the special relationship of the hardware with IBM Z.

Where commodity bolt-on security is concerned with data-at-rest, IBM Z pervasive encryption includes data-in-flight.

- **Parallel Sysplex:** Traffic between nodes and the coordinating Coupling Facility is encrypted.
- **Client:** A readiness tool, zERT, is provided to enable administrators to determine if network traffic meets specified policy.

Because, an auditor can independently verify the IBM Z data and infrastructure is protected and encrypted according to established enterprise security policy, any breach can be determined to be non-reportable in near real time.

Avoiding reportable data breaches

Attempting to address reportable data breaches with commodity hardware and software is likely to be time consuming and of limited efficacy. Even bolting on transparent encryption software only addresses part of the problem. Issues may still crop up with data at rest and data in flight is not considered.

With closely integrated hardware and software, IBM Z provides a comprehensive pervasive encryption capability that addresses both data at rest and data in flight. It does so with minimal impact to existing development processes and minimal service interruption. Superior cryptographic hardware provides superior security with minimum impact on SLAs. IBM Z pervasive encryption is the logical approach to addressing today's data security requirements.

© Copyright IBM Corporation 2017

IBM Corporation
Route 100
Somers, NY 10589
USA

Produced in the United States
June 2017
All Rights Reserved.

IBM, the IBM logo, IBM Z, z14, and Crypto Express are trademarks or registered trademarks of International Business Machines Corporation in the United States, other countries, or both.

Other company, product or service names might be trademarks or service marks of others.

The information contained in this documentation is provided for informational purposes only. While efforts were made to verify the completeness and accuracy of the information contained in this documentation, it is provided "as is" without warranty of any kind, express or implied. In addition, this information is based on IBM's current product plans and strategy, which are subject to change by IBM without notice. IBM shall not be responsible for any damages arising out of the use of, or otherwise related to, this documentation or any other documentation. Nothing contained in this documentation is intended to, nor shall have the effect of, creating any warranties or representations from IBM (or its suppliers or licensors), or altering the terms and conditions of the applicable license agreement governing the use of IBM software.

No other publication, distribution or use is permitted without the prior written consent of IBM. Customers who want a "deep drill down" on CPO Competitive Case Studies should be directed to the IBM Competitive Project Office under NDA. An NDA is required to explain CPO's methodologies, processes and competitive comparison. You can contact IBM Competitive Project Office by sending an email to ibmcpo@us.ibm.com

References in these materials to IBM products, programs, or services do not imply that they will be available in all countries in which IBM operates. Product release dates and/or capabilities referenced in these materials may change at any time at IBM's sole discretion based on market opportunities or other factors, and are not intended to be a commitment to future product or feature availability in any way.

IBM Competitive Project Office (CPO) Case Study results are based on IBM internal studies of IBM brand name solutions designed to replicate a typical IBM customer workload usage in the marketplace. The results were obtained under laboratory conditions, and not in an actual customer environment. IBM's internal workload studies are not benchmark applications, or industry standards, nor are they based on any benchmark standard.