Zettaset

Encrypting Data-at-Rest and in-Motion for Pivotal Greenplum

by Ian Redzic

Hackers want data, and data warehouses are prime targets—all the customer data is just sitting there in one place. If it is unencrypted, it is scary to think what a breach affords. In addition, regulatory standards, like HIPAA, HITECH, and PCI, are absolute requirements in the world of compliance.

Given these facts, our Pivotal Greenplum customers often look for additional encryption on data-at-rest (DAR) and data-in-motion (DIM). The massively parallel processing (MPP) architecture of Pivotal Greenplum provides an architecture that is unlike traditional OLAP on RDBMS for data warehousing, and encryption capabilities must address the scale-out architecture. In this case study, we will cover key crypto architecture considerations, Pivotal Greenplum's architecture, Zettaset's XCrypt™ Full Disk Encryption capabilities, and the way XCrypt Full Disk addresses DAR and DIM within Pivotal Greenplum's architecture.

Top Level Crypto Architecture Considerations

In big data land, performance and scale are important. When we add crypto to a big data architecture, we always take a hit on performance. So, performance degradation becomes a key consideration and is dependent on several factors—the amount of data, decisions about what requires encryption versus left in the clear, algorithms used, key server management, key renewal periods, granularity of access control, ease of implementation, ease of administration, and more. Just like every other crypto solution out there, Zettaset XCrypt Full Disk Encryption implementations need to be looked at through this lens.

Pivotal Greenplum®

Overview

Business Challenge

- Data warehouses and other data stores have become prime targets for threat actors
- Cloud and VM environments frequently share disks or volumes, increasing data privacy concerns
- In high-volume data environments, performance, scale and ease-of-use must be primary considerations

Zettaset Solution

- All-inclusive, easy-to-deploy, software-defined approach combines encryption software with integrated virtual key manager and virtual HSM (hardware security module)
- Fully automated key management and key storage deliver greater security
- Combined protection for data at-rest and data in-motion on and between nodes

Business Value

- All-software approach is optimized for today's distributed/virtual computing environments, eliminates the need for proprietary hardware, and significantly reduces TCO
- Automated installation and deployment processes simplify IT operations and reduce reliance on security specialists
- Full-disk encryption software encrypts all data on a disk providing a higher level of protection for sensitive business and customer information

There are also important standards compliance considerations for key management interoperability protocol (KMIP) and PKCS #11, and Zettaset XCrypt Full Disk Encryption supports both with certified interoperability for several key management solutions and hardware security modules.

Maintaining Encrypted Systems

The largest advantage that I found in working with Zettaset XCrypt Full Disk Encryption's DAR and DIM modules was how easy they were to deploy. Usually, adding encryption is a painful endeavor. However, Zettaset's installation was fairly simplistic. But, due to the typically huge undertaking and sensitive nature of of encryption system installations, people don't want to touch them once they are up and running. This is a security issue in itself, as a system that isn't maintained and updated is prone to becoming a liability. You cannot set it up once and assume it will always be good. Making maintenance a simple, disciplined process is a key part of the overall solution.

Standard Pivotal Greenplum Architecture and Security

As a standard practice for a secure Pivotal Greenplum environment, it is recommended that only the master nodes (and potentially the ETL nodes) are allowed to have connectivity to systems and networks outside of the cluster (see Figure 1). Segment host nodes should only able to access the interconnect network and are cut off from any external connectivity. Users querying the data have no need to actually access a segment host directly. They will connect via a psql session with the appropriate username and password to the Pivotal Greenplum master. The master server receives the query from the user, authenticates them and determines what data then have been authorized to access. It then checks their data access rights and masks all the mechanization which is done behind the scene (in the cluster) in order to return an answer. Behind the curtain, Pivotal Greenplum, being a massively parallel system, is reading data from multiple servers in parallel and setting up connections between the segment host nodes in order to move data across the network so it may be shared between processes as necessary.

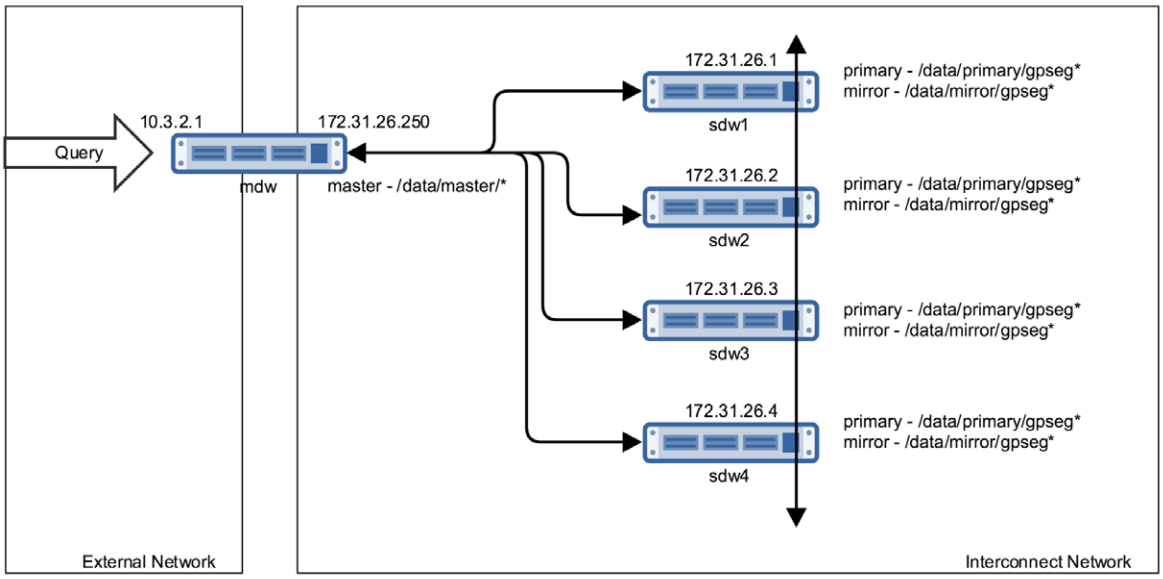


Figure 1. Basic Segmented Network without Zettaset XCrypt Full Disk Encryption

This setup creates a strong logical and often physical separation between the backend that stores all of the data and the frontend access point used to query that data making it easier to keep unwanted users out of the system. Additionally it allows administrators to create the backend segment node cluster with a minimal amount of access rules and user accounts, making the servers much easier to keep secure.

For organizations looking to run systems on shared virtualized or cloud environments, or require additional security to prevent unauthorized persons from obtaining your data, this basic configuration will not suffice. This is where Pivotal has partnered with Zettaset to provide additional data protection options, such as encrypting data-at-rest and data-in-motion.

Encrypting Data-at-rest

Companies want to protect data-at-rest. There are two main tactics used when we look at encrypting the bits sitting on the disk. The first we covered in a previous post on Protegrity and involves making function calls to encrypt or decrypt specific pieces of s and need to be replaced. If the technician replaces the malfunctioning equipment and walks out the door with the bad drive or server and it is unencrypted, they are walking out the door with some data they could potentially access. Additionally, cloud and VM environments frequently share disks or volumes—these are reused by newly launched instances. And, you may need to guarantee that other users will not be able to access data that was resident on that piece of equipment.

This is where Zettaset's XCrypt Full Disk Encryption technology comes into play. As the server boots up and mounts an encrypted partition, it needs to exchange information with a key management server. Once the proper handshakes have taken place, the Zettaset technology allows a decrypted version of the server volumes to be mounted and treated like a normal partition. Zettaset provides the pieces to automate all of this and integrate with your existing key management and HSM (hardware security module) solutions. The Zettaset XCrypt Full Disk Encryption solution also includes a virtual key manager and virtual HSM which can alternatively be deployed if needed. Figure 2 below depicts the mount points that you would typically encrypt in a Pivotal Greenplum environment in order to protect the data. In this scenario you would be using the Zettaset key management server to store and manage credentials. As the servers in the cluster booted they would do a key exchange with the Zettaset server following the LUKS specification. If this exchange works, the server would then be able to mount the /data partition us dm-crypt so that the master could read the files it needs out of /data/master. The segment nodes would each individually go through their own exchange and validations so that they could access the /data partition which contains the files necessary to run the primary and mirror and present their data.

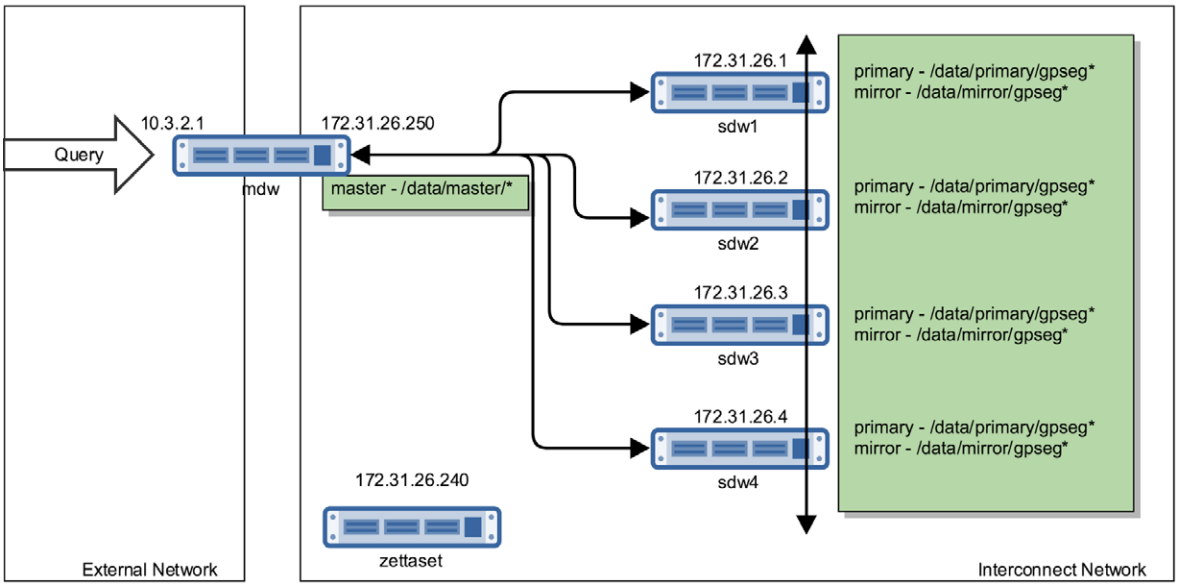


Figure 2. Pivotal Greenplum with Zettaset DAR

In this case, you have setup Zettaset XCrypt Full Disk Encryption for DAR to encrypt the /data mountpoint. Let's say you have an issue with the motherboard in sdw3. You migrate to a backup node in the rack, and a technician comes in to replace the server, drives and all. The technician takes the server back to their shop to do work on it. As they power on the system, it will be unable to negotiate the key exchange. At that point, the /data mount would not be able to be attached to in an unencrypted manner and they would not be able to access the data on it.

Setup is fairly simple. First, you download and unzip Zettaset's XCrypt Full Disk Encryption DAR package along with the documentation. You will need to install some prerequisite crypto and python rpms and then edit a configuration file with the certificate you wish to use, key management server info (they bundle one if you don't have your own), and the partitions targeted for encryption.

In my case, I used AWS and mounted an EBS volume for encryption. I ran my Zettaset server installation on it's own server. The host and partition information section in my configuration file looked like the following. Each line represents each node in the environment.

```
mdw encrypted_blockdev=/dev/xvdb encrypted_mountpoint=/var/lib/zts/slave/crypt1 encrypted_mountnames=crypt1 encrypted_preserve=n
sdw1 encrypted_blockdev=/dev/xvdb encrypted_mountpoint=/var/lib/zts/slave/crypt1 encrypted_mountnames=crypt1 encrypted_preserve=n
sdw2 encrypted_blockdev=/dev/xvdb encrypted_mountpoint=/var/lib/zts/slave/crypt1 encrypted_mountnames=crypt1 encrypted_preserve=n
sdw3 encrypted_blockdev=/dev/xvdb encrypted_mountpoint=/var/lib/zts/slave/crypt1 encrypted_mountnames=crypt1 encrypted_preserve=n
sdw4 encrypted_blockdev=/dev/xvdb encrypted_mountpoint=/var/lib/zts/slave/crypt1 encrypted_mountnames=crypt1 encrypted_preserve=n
```

With no previous information on the device I wanted to save, I went ahead and set encrypted_preserve=n. This tells the system not to preserve any data and format the volume. There is also an option to save the data and write it back to the now encrypted device or to wipe the device by writing zeros to it before using it. Next, I ran the installer. The installer uses Ansible to setup and configure the software components that are necessary to implement and manage the volume encryption on the nodes were specified. Post install, I ssh-ed over to a node, and, by using lsblk as show below, I can see the encrypted device has been mounted.

```
[root@sdw1 ~]# lsblk -o NAME,FSTYPE,SIZE,MOUNTPOINT
NAME            FSTYPE       SIZE MOUNTPOINT
xvda                          8G
└─xvda1         ext4          8G /
xvdb            crypto_LUKS  100G
└─crypt1 (dm-0) ext4         100G /var/lib/zts/slave/crypt1
```

To simplify things for my testing, you will note that I encrypted /var/lib/zts/slave/crypt1 and then I created a symlink to /data. Normally I would have just made the encrypted device actually be /data. In this case I was doing testing that involved switching back and forth between an EBS volume where I had an encrypted set of data files and an EBS volume with an unencrypted set and it was easier to keep the database configuration the same and change the symlink. Below you can see output of the lsblk command showing the encrypted volume mounted to the system.

With the server and each agent up and running, the key exchange takes place automatically when the server boots. I don't have to do any manual steps each time the system comes up. If the key management server is down or the target server is unable to communicate with it, the device will not mount. It is also worth noting that there is some flexibility in the granularity of crypto controls. For example, you can encrypt data on segments 1 and 2, while segments 3 and 4 can be used to store/transmit unencrypted data.

Encrypting Data-in-motion

Many companies also want to protect data as it is passed between nodes. Normally, this traffic sits on it's own interconnect, and it is segmented away from any other network access. This is typically enough protection for most use cases. Since we see more cloud and virtualized deployments of Pivotal Greenplum, there are more requests to encrypt the traffic that passes between the nodes. Zettaset's XCrypt Full Disk Encryption for DIM (Data-In-Motion) installs and manages the pieces that allows you to encrypt data as it passed between nodes. The encryption is applied to communication from the master to segment hosts, segment hosts to the master, and between the segment hosts themselves.

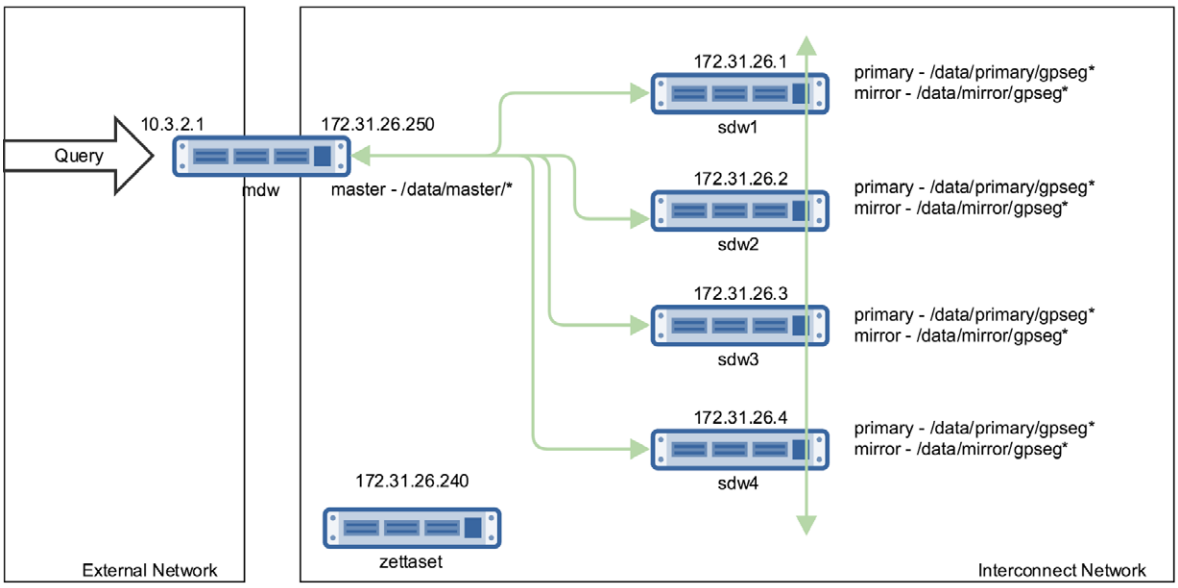


Figure 3. Pivotal Greenplum with Zettaset DIM. Green lines indicate encrypted connections between master and segments.

Again implementation involves downloading the appropriate package and documentation from Zettaset. The setup is quite similar to that of DAR. There is a second Ansible-driven deployment that is defined by configuration file parameters. Here, there is a key difference. Instead of calling out mount points, you specify network coverage for encrypted and unencrypted traffic. We will use an unencrypted channel for a comparison test, but this is not how a real-world network would likely be set up.

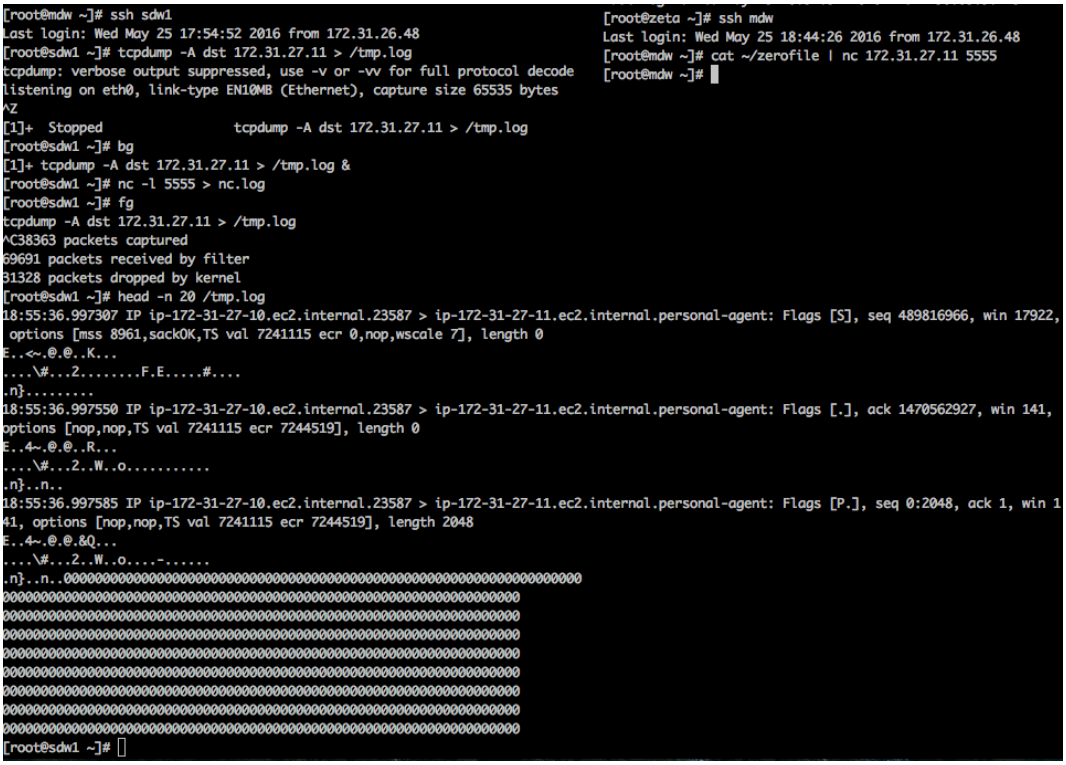
```
# EDIT configure public and private networks
net_public=172.31.27.0/24
net_private=172.31.28.0/24
```

Since my cluster was running in Amazon, I added two additional IPs to each node to test the ability to run encrypted and unencrypted traffic between the nodes. I executed commands similar to the following on each node, ensuring that the newly added addresses communicated with addresses on the other nodes that were similarly grouped as part of the encrypted/private or unencrypted/public IP address allocations.

```
ip addr add 172.31.27.10 dev eth0
ip addr add 172.31.28.10 dev eth0
ip route add 172.31.27.0/24 via 172.31.16.1 src 172.31.27.10
ip route add 172.31.28.0/24 via 172.31.16.1 src 172.31.28.10
```

At this point, traffic in the 172.31.28.x range should be encrypted and traffic in the 172.31.27.x range should be in the clear. In order to verify this, I created a 500M file filled with zeros. Tcpdump was setup to watch the interface, and a couple of netcat commands transferred the file between nodes. This being a simple test to simulate the movement of data between two segment nodes within the cluster. When I moved file across the public interfaces on the 172.31.27.x network, we can see the zeros show up in the tcpdump, as you would expect for unencrypted traffic.

Moving over to the encrypted interface on the 172.31.28.x network, I did the same thing, and, as expected, we see a much different story. The plainly visible zeros are now nowhere to be seen.



Now, anyone sitting on the network with a packet sniffer would have to spend a large amount of brute force effort to to decode my 500MB of zeros without a key, if they could even manage to break the encryption. It is also worth noting that there is some flexibility in the granularity of cryptographic controls using Zettaset XCrypt Full Disk technology for data-at-rest (DAR) and data-in-motion (DIM).

Learning More

To learn more about Pivotal Greenplum and Zettaset integration, please visit the Pivotal-Zettaset partner page. For more about Zettaset, visit their XCrypt Full Disk product page.

Zettaset
465 Fairchild Drive, Suite 234, Mountain View, CA 94043 // USA: +1.650.314.7920 // Fax: +1.650.314.7950
sales@zettaset.com // www.zettaset.com