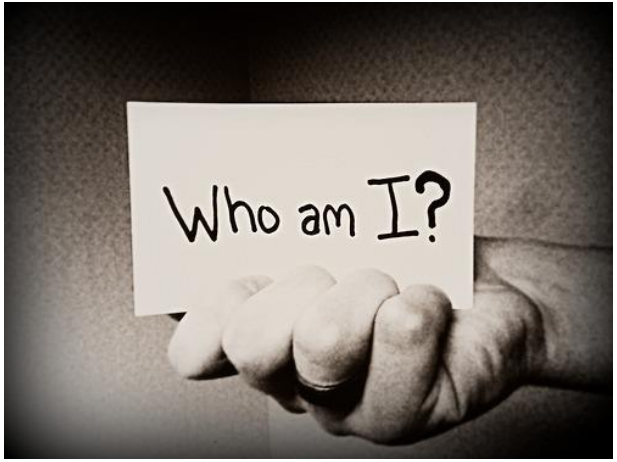
Securing your Big Data Environment

Ajit Gaddam

@ajitgaddam



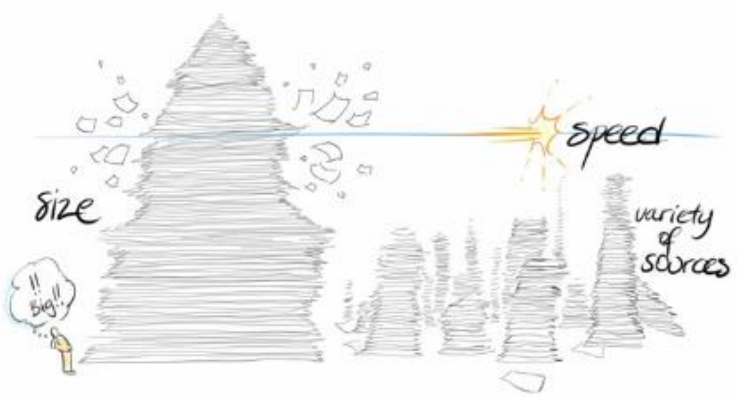
@ajitgaddam

- @VISA Chief Security Architect
- Before – senior tech roles at diff tech & FI companies
- Co-founder of 2 startups
- Co-Author of *Hadoop in Action-2* book by Manning Publications
- SABSA, CISSP, GSEC, GPEN, TOGAF

Agenda

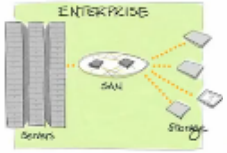
- ❑What is Big Data and why should I secure it?
- ❑Security Risks & Threat Models
- ❑Big Data Security Framework
- ❑Successes, Failures, and Best Practices

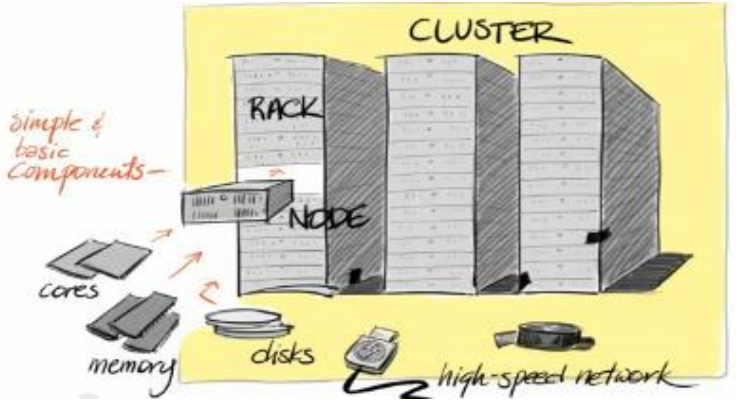
What is Big Data



"Big Data refers to datasets whose size and/or structure is beyond the ability of traditional software tools or database systems to store, process, and analyze within reasonable timeframes"

HADOOP is a computing environment built on top of a distributed clustered file system (HDFS) that was designed specifically for large scale data ope





Images source : EMC Big data 2012
http://emc.im/MprmL3

Three Reasons for Securing Hadoop (.. atleast)

1. Contains Sensitive Data

Teams go from a POC to deploying a production cluster, and with it petabytes of data.

Contains sensitive cardholder and other customer or corporate data that must be protected

2. Subject to Regulatory Compliance

With #1 comes compliance to PCI DSS, FISMA, HIPAA, EU laws, US federal/state laws to protect PII, cardholder, and other in-scope data

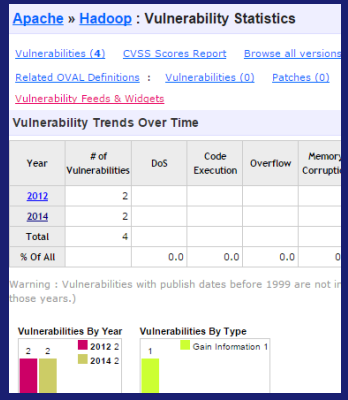
3. Can enable your business

Before, usage was broad and possibly restricted to non-sensitive data.

With security in place, you can allow for sensitive workloads on restricted datasets

Image source: World's Biggest Data Breaches (Source: Information Is Beautiful, DataBreaches.net, IdTheftCentre, press reports)

Current State of Big Data Security



CVE Metrics

Only 4 reported & fixed Hadoop Vulnerabilities in past 3 years

Security Rigor

Traditional RDBMS platforms underwent decades of security evaluations & assessments

OSS Fragmentation

No standardization or portability of Big Data security solutions

Data Breaches over the past year

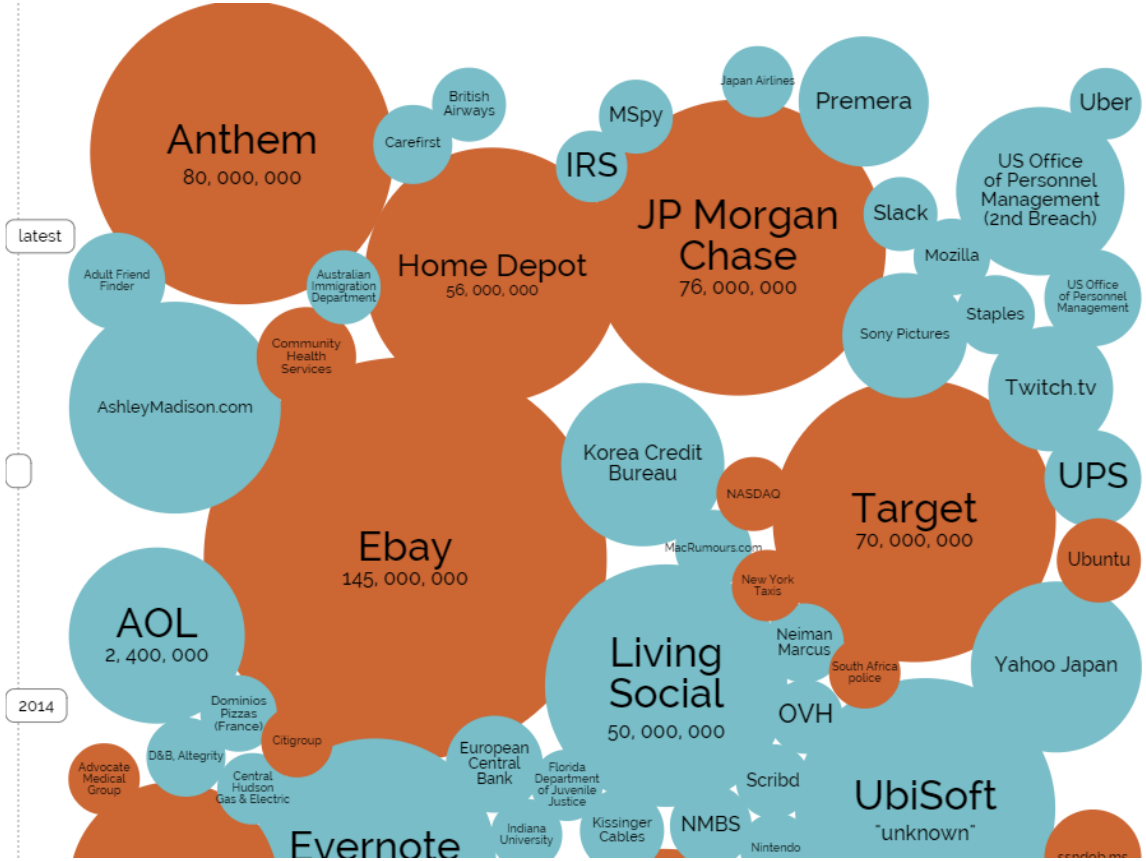
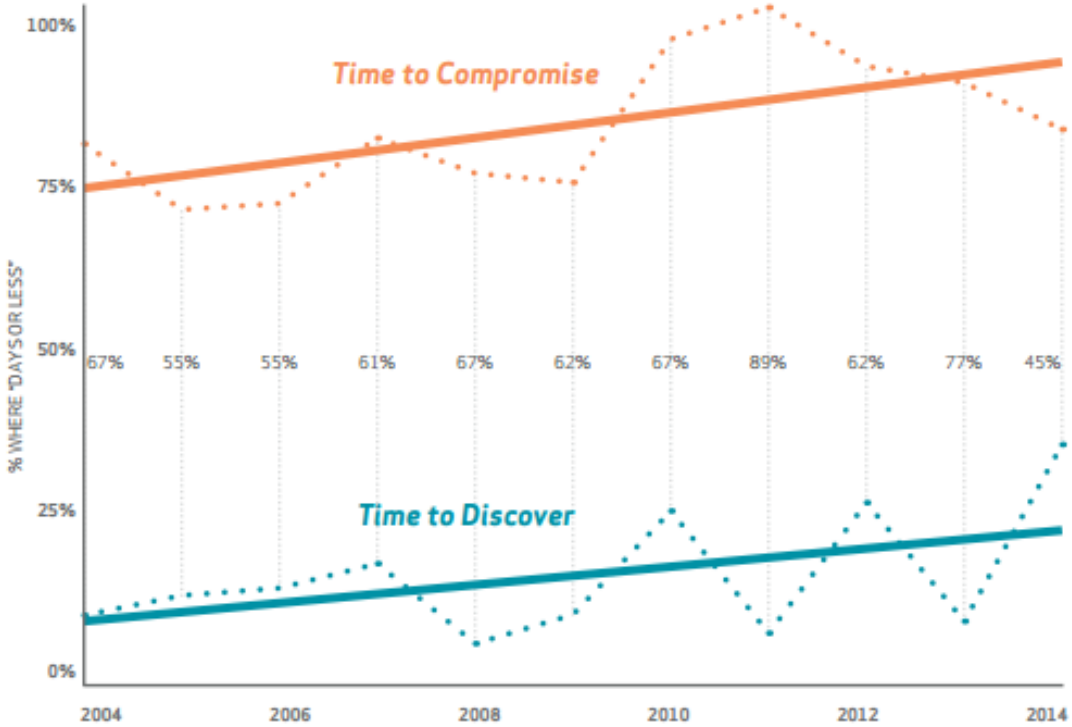


Image source: World's Biggest Data Breaches (Source: Information Is Beautiful, DataBreaches.net, IdTheftCentre, press reports)

Different kinds of PII, financial data, and IP breached.

Federal Govt., Financial Institutions, Tech companies etc.



Source: Verizon Data Breach 2015 report

60%

IN 60% OF CASES, ATTACKERS ARE ABLE TO COMPROMISE AN ORGANIZATION WITHIN MINUTES.


BOY,
THAT ESCALATED QUICKLY
quickmeme.com

Ultimate Goal of an Attacker

- The primary goal is to obtain sensitive data that sits in a Big Data cluster.
- This could include different kinds of regulated data (e.g. Payment data, Heath data) or other personally identifiable data (PII)
- Other attacks could include attacks attempting to destroy or modify data or prevent availability of this platform. However, these attacks are less common than data theft, and can be mitigated with other strategies and controls which we will talk about.

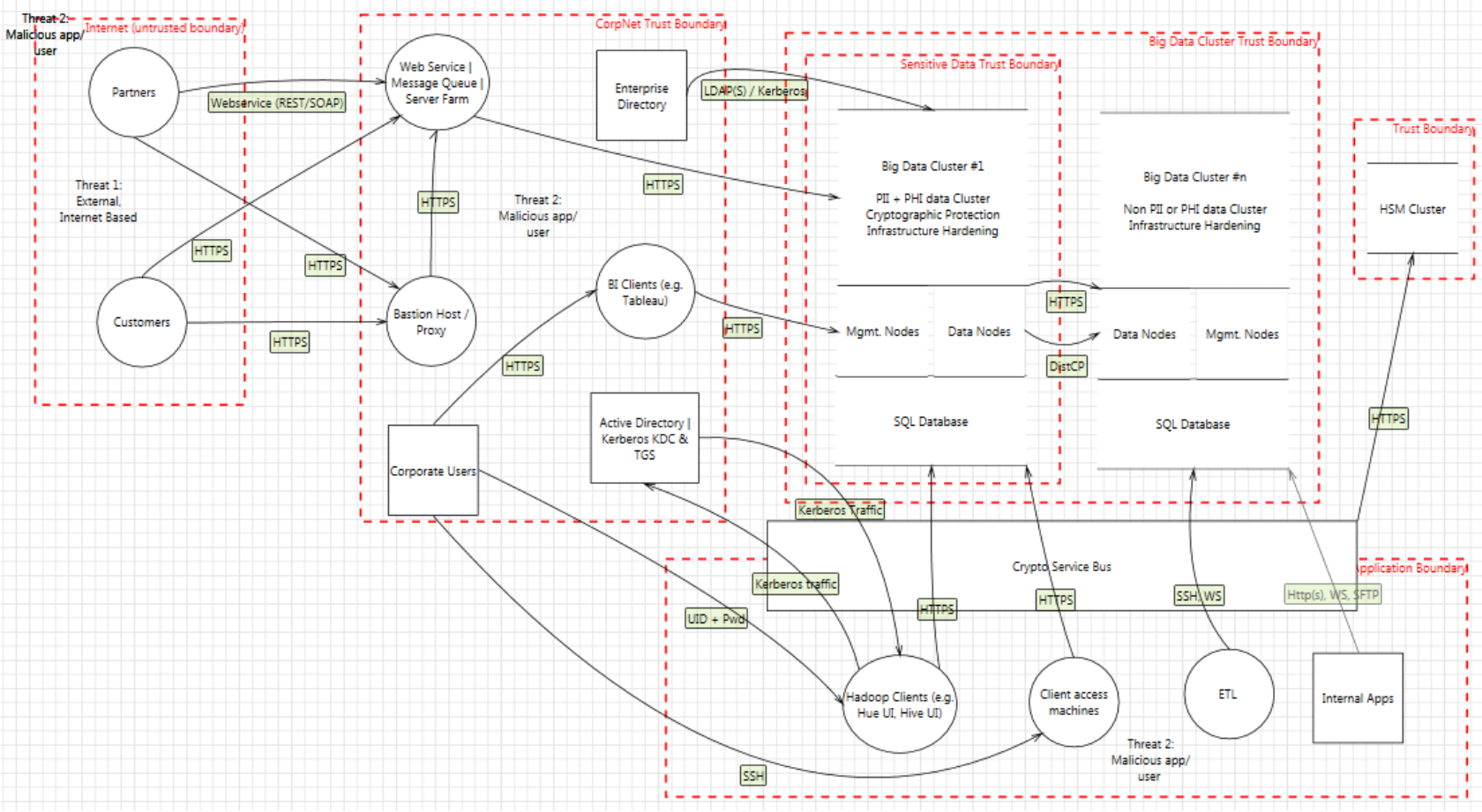
Threats on Big Data Platforms (part 1)

- **Unauthorized access**: Attacker attempts to gain privileges they do not have
  - Authentication through Kerberos (for services in Hadoop and clients connecting to Hadoop cluster) and through LDAP (enterprise clients authenticating to services). SAML is also used in certain cases (e.g. Hue) to provide SSO integration.
  - Authorization through file and directory permissions in HDFS. Components like Sentry can provide authorization in core Hadoop. SAML provides authZ for CM and Hue
  - Auditing authN and authZ decisions to capture evidence when accounts get compromised or when insider attacks happen.
- **Network based attacks:** Packets can be dropped, tampered with, or read in flight. The only mitigation is to encrypt all data in flight.
  - Transport Layer Security (TLS) provides confidentiality of data and provides authentication via certificates and data integrity verification
  - SASL Encryption is a feature in core Hadoop. Should not be used over TLS since it has not undergone the extensive testing and validation unlike TLS

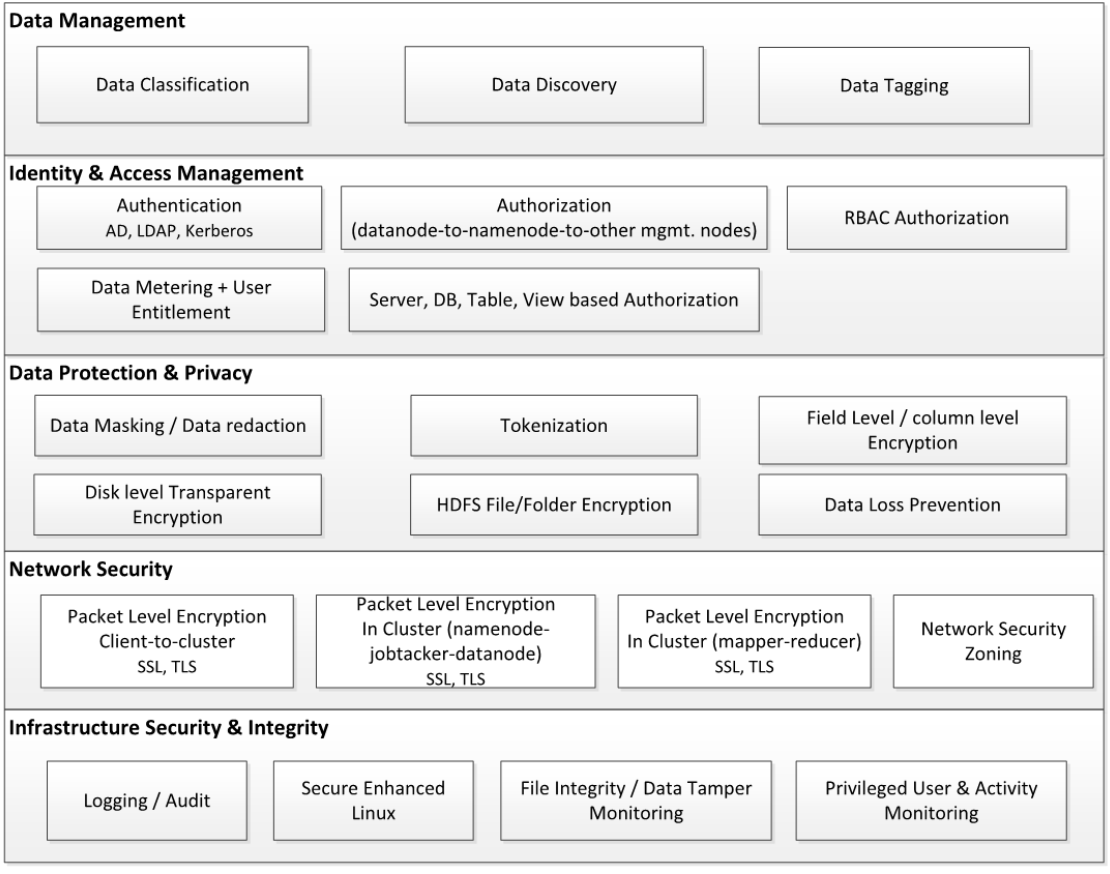
Threats on Big Data Platforms (part 2)

- **Host-Level Data at Rest Attacks**: Since data is an attackers ultimate target, encrypting data on disk is very important. Also, artifacts like Kerberos keytab files reside on the machine file system and could be accessed & impersonated by privileged users or services.
  - Application Level Cryptographic protection: Through encryption or tokenization or data masking techniques when in-use or when stored at rest.
  - HDFS Level Encryption: Data is encrypted natively inside HDFS and users in the OS can only see encrypted data when files with HDFS data are accessed.
  - File-system / volume Level Encryption: Least common denominator but does provide threat mitigation coverage if done right
- **Infrastructure Security**: Large exposure surface for the data lake. It has middleware components like Java that needs patching. Rogue nodes can be added to the cluster.
  - Automation: Use [Sec]DevOps and leverage Chef/Puppet to validate nodes
  - SELinux: Label data and enforce MAC

Threat Modeling of your Big Data Environment



Big Data Security Framework



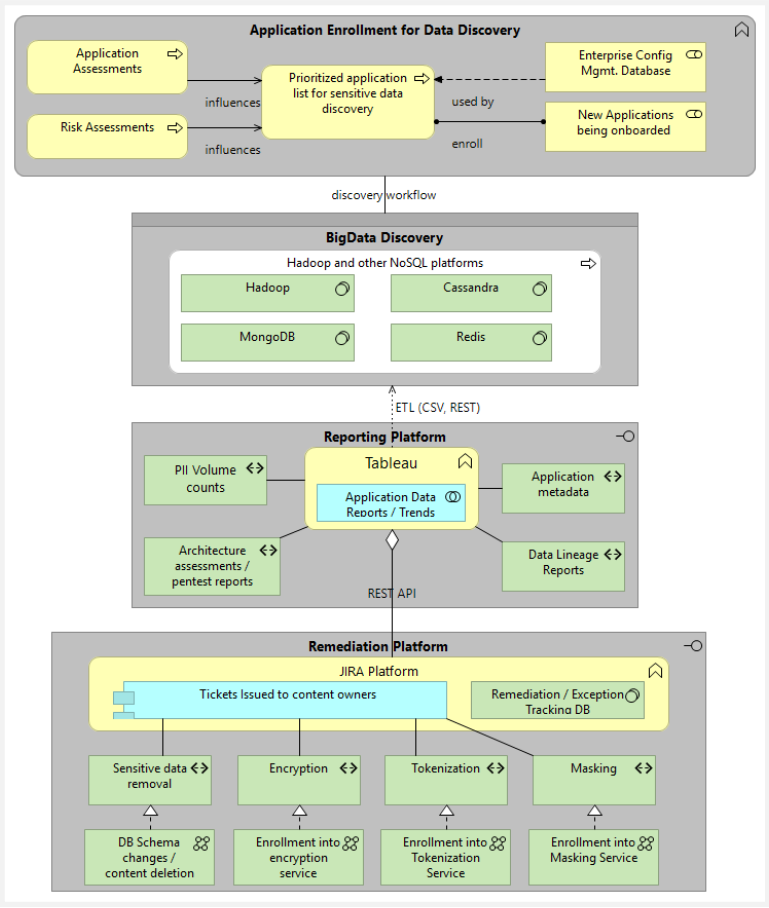
5 Pillars of Big Data Security

- Data Management
- Identity & Access Management
- Data Protection & Privacy
- Network Security
- Infrastructure Security & Integrity

Pillar #1: Data Classification & Prioritization

Data Element	Strength of a security control applied to the data {B}	Value to an Attacker {C}	Total Likelihood (B+C)	Capital Costs (e.g. compliance, revenue impact) (quantitative) {E}	Brand Impact Cost (qualitative) {F}	Impact to the customer {G}	Total Impact Score (E+F+G)	Final Score (Likelihood * Impact)
Social Security Number	8	8	16	3	8	10	21	336
Credit Card Number	6	10	16	10	10	9	29	464
Corporate Strategic Information	4	10	14	10	8	1	19	266

Pillar #1: Sensitive Data Discovery



Data Discovery is ground zero for data protection

- Determine prioritized application list
- Perform conditional discovery searches
- Report and visualize relevant metrics
- Feed this data to data protection services

Pillar #1: Data Tagging

- Understand your end-to-end data flows, especially the ingress & egress methods from your Big Data Cluster. Bake back into your Threat Model

Ingress Methods into Big Data Cluster (sample)

These are going through some meta layer – copy files or create+write

1. CLI
2. Java API (+Oozie)
3. Pig
4. Datameer
5. Flume with HDFS sink
6. Impala ODBC/JDBC
7. Sqoop Import
8. SSH in
9. Batch ETL loads – through 3rd party connectors

Egress Components out of your Cluster (sample)

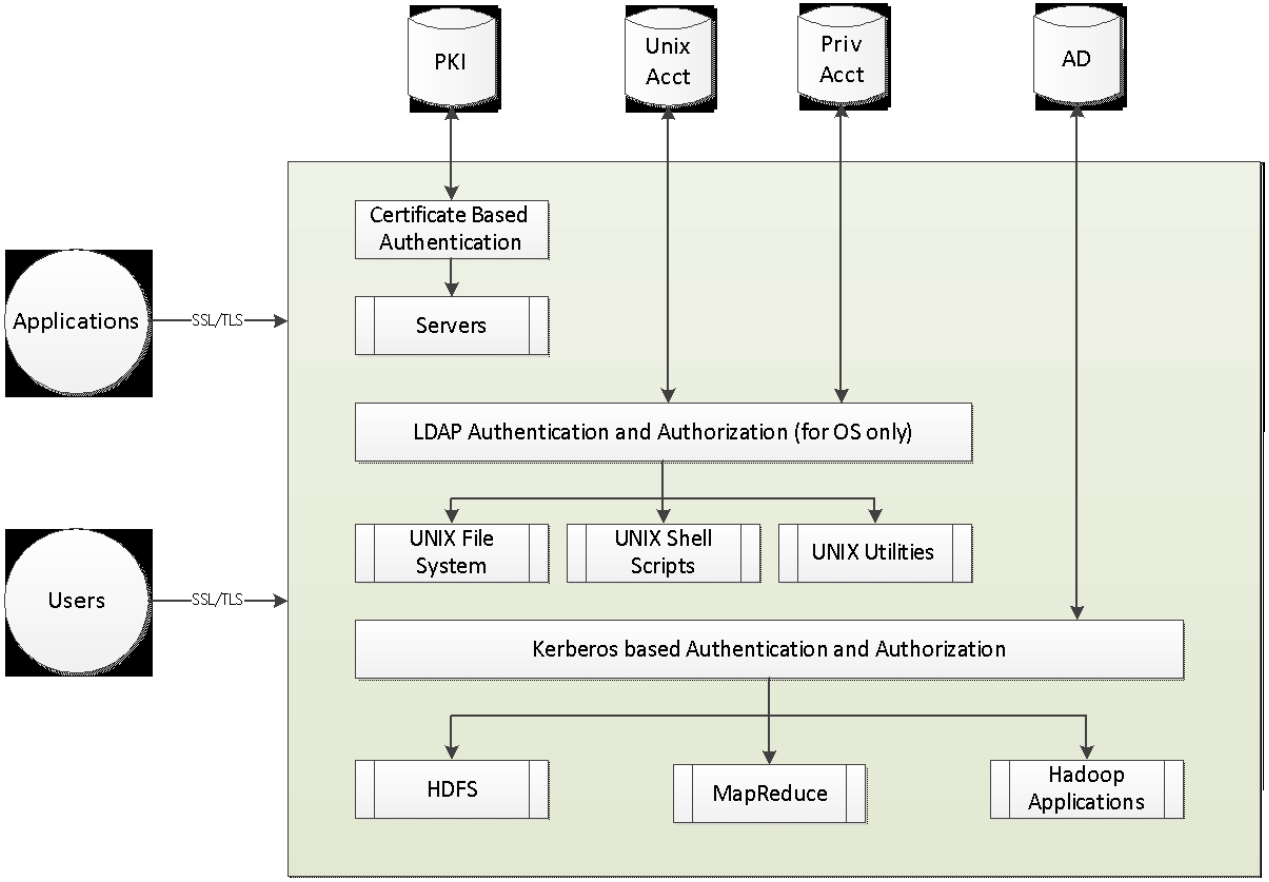
1. Hive Query (CLI, HUE, ODBC/JDBC, Oozie)
2. Impala Query (CLI, HUE, ODBC/JDBC, Oozie)
3. PIG job (reads either files or Hive tables via Hcatalog) – CLI, Hue, Oozie
4. Custom MapReduce jobs (Java MR, Streaming, Hive, Pig)
5. Sqoop export
6. Copy files (CLI, Java API, REST API, Oozie, Hue, Datameer)
7. Microstrategy – connecting to Hive & Impala
8. Tableau
9. HUE Web UI

- Follow the data, and tag them using discovery capability – pass it to identity & audit controls to manage data access (implement RBAC, ABAC ..)

Pillar #1: Data Management Summary

1. Search for, identity and classify on presence of your sensitive data.
2. Collect metrics (e.g. volume, unique counts, conditional searches (CC + zip)
3. Account for data structure in your Hadoop cluster. Move away from sequence and delimited files into a columnar storage format like Apache Parquet
4. Plan for once data has been protected (e.g. encrypted, tokenized etc.)
5. Data compression algorithms (e.g. max compression but slower I/O - gzip)
6. Understand your end-to-end data flows, especially the ingress & egress methods

Pillar #2: Identity & Access Management



Pillar #2: IAM Summary

1. **User Entitlement:** Provide users access to data; centrally manage policies; tie policy to data and not access method
2. **RBAC Authorization:** Manage data access by role (and not user)
3. **Authorization:** Leverage Attribute based access control and protect data based on tags that move with the data through lineage; permissions decisions can leverage the user, environment (e.g. location), and data attributes.
4. Use 2FA for admin access and segregation of duties between the different roles
5. Use tools:
   - Do not allow your end users to connect to data nodes, but to name nodes only (Apache Knox can help here)
   - Apache Sentry is a good platform to leverage to manage RBAC; it can leverage AD to determine user's group assignments automatically and keeping its permissions upto date.

Pillar #2: IAM Summary

```
[root@proxy ~]# ktutil
ktutil:  rkt /etc/krb5.keytab
ktutil:  l
slot KVNO Principal
---- ---- ---------------------------------------------------------------------
   1    2       HTTP/proxy.example.lan@EXAMPLE.LAN
   2    2       HTTP/proxy.example.lan@EXAMPLE.LAN
   3    2       HTTP/proxy.example.lan@EXAMPLE.LAN
   4    2                   HTTP/proxy@EXAMPLE.LAN
   5    2                   HTTP/proxy@EXAMPLE.LAN
   6    2                   HTTP/proxy@EXAMPLE.LAN
ktutil:  _
```

Protect keytab files in a Kerberos cluster.

- The keytab file (short for key table) contains the principle (service key) used by a service to authenticate to the KDC
- Threat: It is possible to impersonate root or an application ID by copying its keytab

Keytab files are analogous to a user/application password.

- Store the keytab files on local disk and make them only readable by the root
- Have the highest level of monitoring on these files. Never send it over an insecure network
- Ensure that root account is limited (actions of those people, when authenticated as Root, cannot be tied back to those individuals)
- {protection} SELinux policies & disk level cryptography

Pillar #3: Cryptographic protection of Data-at-Rest

- Format Preserving or Transparent protection
- Strong access control
- Sep. of duties & keys

Application Level	HDFS – Level	Disk - Level
• Application Integration *Precise user control & authN, authZ. tokenization if cannot accommodate encrypted (binary) data* • Secure data during ingestion • Bulk protect existing data • Use an External Key Mgr. with crypto keys in HSM	• Transparent Encryption *No application code changes. Content encrypted on write and decrypted on read. Best for new data* • Protects against file-system & OS level attacks • Use an External Key Mgr. with HSM key hierarchy • AES-NI Hardware acceleration	• Transparent Encryption *Layer between application & file system. Best option for existing data in cluster* • Process based access control • Can secure metadata, logs & config files • Use an External Key Mgr.

Pillar #3: Data Protection & Privacy Summary

- Protection against different threat models (data center, cloud, external, internal)
- Need to protect all data (structured, unstructured, metadata, files)
- Many types of users; many types of data; different sensitivity levels; diff tools
- Crypto downstream impact
- My favorite crypto method – Tokenization
- FPE is still evolving (no standard yet; NIST has FFX, BPS as finalists)
- Field level encryption can provide security granularity and audit tracking, but comes at expense of manual intervention to determine sensitive fields and where and how to enable authorized decryption

Pillar #3: Data Protection & Privacy Summary

- Use a central key mgmt. server to manage the crypto keys. Also separates keys from data. Remember, it is not just crypto keys. There are SSL certs, SSH keys, passwords, Kerberos keytab files also.
- Go for standards based technologies where possible (e.g. using KMIP vs. proprietary key mgmt. technologies to avoid vendor lock-in)
- Leverage encryption at hardware level where possible – e.g. AES NI with Intel Optimization
- For encryption, determine bottlenecks based on where crypto is done (local, remote over WS); network backbone
- Support individual crypto keys for different types of data (e.g. PCI, PII)

Pillar #4: Network Security

- While network based attacks like certain service denial attacks can be mitigated through load balancing technology, core Hadoop does not provide native safeguards.
- Service denial attacks could include DoS, DDoS, flooding a cluster with jobs, or running jobs consuming a high amount of resources.
- Untrusted mappers can be altered to snoop on requests, alter MapReduce scripts, or alter results.
- Perform packet level encryption and protect the client to cluster data with TLS
- Protect communication traffic within cluster (namenode-jobtracker-datanode)
- Protect traffic in cluster (mapper-reducer) jobs

Pillar #4: Network Security Summary

- Implement secure communication between components (e.g. consoles to servers)
- Use TLS protocol to authenticate and ensure privacy of communications between nodes, name servers, and applications
- Allow your admins to configure and enable encrypted shuffle and TLS/https for HDFS, MapReduce, YARN, HBase UIs etc.
- Hadoop 2.0 is first a resource mgmt. framework. Although app instances will be launched as close to data as possible, there is a possibility that the scheduler cannot find resources next to the data and may need to read data over the network
- With large data sets, it is nearly impossible to identify malicious mappers that may create significant damage, especially for scientific and financial computations.
- Do Endpoint validation - Ingestion of data from trusted and mutually authenticated sources through secure mechanisms to mitigate the threats from malicious entry and compromise of transmission

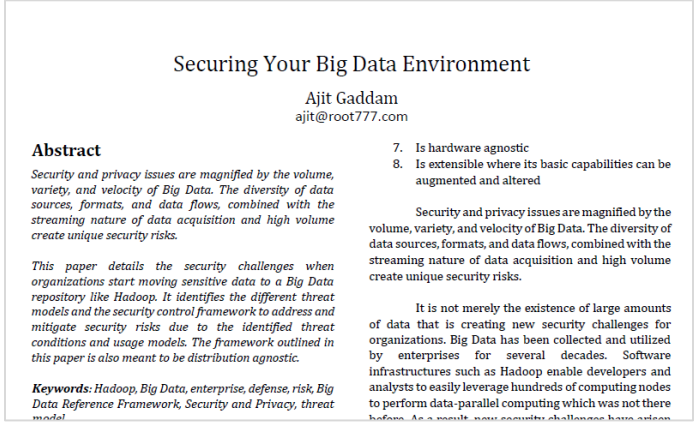
Pillar #5: Infrastructure Security - SELinux

- Built by NSA and eventually adopted by upstream Linux Kernel
- Systems built on Linux typically do DAC. SELinux is an example of MAC
- Even if a user changes any settings on their home directory, the policy prevents another user or process from accessing it
- Prevent command injection attacks – e.g. lib files with x but no w or vice-versa
- Write to /tmp but cannot execute. Malicious actors can have fun reading our policy documents.
- Labeling files and granting permissions is what SELinux is all about
- Run SELinux is permissive mode and use warnings generated to define the SELinux policy which after tuning can be deployed in a ‘targeted enforcement’ mode

Future Work

Book: Hadoop in Action by Manning Publications

- The discount code is lam2cf and that is worth 50% off at manning.com
- ISBN: 9781617291227
- The book page is here: http://manning.com/lam2/.



Securing Your Big Data Environment

Ajit Gaddam
ajit@root777.com

Abstract

Security and privacy issues are magnified by the volume, variety, and velocity of Big Data. The diversity of data sources, formats, and data flows, combined with the streaming nature of data acquisition and high volume create unique security risks.

This paper details the security challenges when organizations start moving sensitive data to a Big Data repository like Hadoop. It identifies the different threat models and the security control framework to address and mitigate security risks due to the identified threat conditions and usage models. The framework outlined in this paper is also meant to be distribution agnostic.

***Keywords**: Hadoop, Big Data, enterprise, defense, risk, Big Data Reference Framework, Security and Privacy, threat model*

7. Is hardware agnostic
8. Is extensible where its basic capabilities can be augmented and altered

Security and privacy issues are magnified by the volume, variety, and velocity of Big Data. The diversity of data sources, formats, and data flows, combined with the streaming nature of data acquisition and high volume create unique security risks.

It is not merely the existence of large amounts of data that is creating new security challenges for organizations. Big Data has been collected and utilized by enterprises for several decades. Software infrastructures such as Hadoop enable developers and analysts to easily leverage hundreds of computing nodes to perform data-parallel computing which was not there

White paper around this talk will be released shortly next week with all the details

Recap / Summary

1. Understand your data landscape; don't be afraid to pick different vendors for different levels of data sensitivity based on your security control assessment
2. Everything starts with Discovery & classification; build hygiene and protection plan
3. IAM – use Kerberos and protect keytab files
4. Encryption – start with transparent encryption first; mature to app level and tokenization for data at rest; ensure key mgmt. leverages standards; don't have vendor lock-in; see if AES-NI can be leveraged for crypto performance
5. Leverage an external key manager; integrate with HSMs (Java Key Store can be short-term); use standards like KMIP for key exchange
6. Enable TLS – start with client to cluster first; mature to Hadoop nodes (namenode-jobtracker-datanode) and then to the MR jobs. SELinux is cool ! Try it.
7. Data Exfiltration is the final piece of the puzzle; think kill-chain; log & monitor; follow-the-data and build models for your users/apps

Please share your feedback

Electronic evaluations have been sent to your phone

Image source: https://ottomat3ch.files.wordpress.com/2013/11/feedback1.gif

Ask Me Anything

Thank You