Securing Hadoop: Security Recommendations for Hadoop Environments

Version 2.0
Updated: March 21, 2016

Securis, L.L.C. 515 E. Carefree Highway Suite #766 Phoenix, AZ 85085 **T** 602-412-3051
info@securosis.com www.securosis.com

This report licensed by Hortonworks.

HORTONWORKS®
POWERING THE FUTURE OF DATA™

Hortonworks is the leader in emerging Open Enterprise Hadoop and develops, distributes and supports the only 100% open source Apache Hadoop data platform. Our team comprises the largest contingent of builders and architects within the Hadoop ecosystem who represent and lead the broader enterprise requirements within these communities. The Hortonworks Data Platform provides an open platform that deeply integrates with existing IT investments and upon which enterprises can build and deploy Hadoop-based applications.

Hortonworks has deep relationships with the key strategic data center partners that enable our customers to unlock the broadest opportunities from Hadoop.

For more information, visit www.hortonworks.com

This report licensed by Vormetric.

Vormetric
Data Security™

Vormetric's comprehensive high-performance data security platform helps companies move confidently and quickly. Our seamless and scalable platform is the most effective way to protect data wherever it resides—any file, database and application in any server environment. Advanced transparent encryption, powerful access controls and centralized key management let organizations encrypt everything efficiently, with minimal disruption. Regardless of content, database or application —whether physical, virtual or in the cloud—Vormetric Data Security enables confidence, speed and trust by encrypting the data that builds business.

Please visit: www.vormetric.com and find us on Twitter @Vormetric.

Author's Note

The content in this report was *developed independently of any licensees*. It is based on material originally posted on the Securosis blog, but has been enhanced, reviewed, and professionally edited.

Special thanks to Chris Pepper for editing and content support.

Copyright

Table of Contents

Executive Summary

There is absolutely no question that Hadoop is a fundamentally disruptive technology. New advancements — in scalability, performance, and data processing capabilities — have been hitting us every few months over the last 4 years. This open source ecosystem is the very definition of innovation. Big data has transformed data analytics — providing scale, performance, and flexibility that were simply not possible a few years ago, at a cost that was equally unimaginable. But as Hadoop becomes the new normal IT teams, developers, and security practitioners are playing catch-up to understand Hadoop security.

This research paper lays out a series of recommended security controls for Hadoop, along with the rationale for each. Our analysis is based upon conversations with dozens of data scientists, developers, IT staff, project managers, and security folks from companies of all sizes; as well as our decades of security experience. These recommendations reflect threats and regulatory requirements IT must address, along with a survey of available technologies which practitioners are successfully deploying to meet these challenges.

Summary of Recommendations

In this paper we focus on how to build security into Hadoop to protect clusters, applications, and data under management. Our principal concerns include how nodes and client applications are vetted before joining the cluster, how data at rest is protected from unwanted inspection, assuring the privacy of network communications, key management, and how various functional modules are managed. The security of the web applications that use big data clusters is equally important, but those challenges also exist outside big data clusters, so they are outside our scope for this paper. Our base recommendations are as follows:

- Use Kerberos — typically bundled with Hadoop — to validate nodes and client applications before admission to the cluster, and to support other identity functions.
- Use file/OS layer encryption — to protect data at rest, ensure administrators and applications cannot directly access files, and prevent information leakage.
- Use key/certificate management — you cannot simply store keys and certificates on disk and expect them to be safe. Use a central key management server to protect encryption keys and manage different keys for different files.

- Use Apache Ranger to track module configuration and to set usage policies for fine grained control over data access.
- Validate nodes prior to deployment — through virtualization technologies, cloud provider facilities, and scripted deployments based on products such as Chef and Puppet.
- Use SSL or TLS network security — to authenticate and ensure privacy of communications between nodes, name servers, and applications.
- Log transactions, anomalies, and administrative activity — through logging tools that leverage the big data cluster itself — to validate usage and provide forensic system logs.

Introduction

Hadoop is Enterprise Software.

There, we said it. In the last few years Hadoop has matured from a simple distributed data management system for running MapReduce queries, into a full-blown application framework for processing massive amounts of data using just about any method you can conceive. We are long past wondering whether Hadoop is a viable technology, but still coming to terms with its broad impact on data processing in general. Treating Hadoop as just another open source distribution or analytics tool is a mistake. Our research shows Hadoop is affecting the course of *most* new application development within enterprises.

Having demonstrated its value, Hadoop is now being embraced both by enterprises and the mid-market, running just about every type of data processing and analytics you can imagine. Over 70% of large firms we spoke with are running Hadoop *somewhere* within their organizations. A percentage are running Mongo, Cassandra or Riak in parallel, but Hadoop is unquestionably "the face of big data".

But many IT personnel still regard open source with suspicion, and Hadoop as an interloper — a large part of the rogue IT problem. This may surprise some readers, but most big data projects grow organically as 'skunkworks' development efforts— often outside IT governance — so IT folks still struggle to accept Hadoop as "enterprise ready". Make no mistake — "proof of concept" projects just a couple years ago have evolved into business-critical enterprise applications. Hadoop's value is widely accepted, and these systems are here to stay, but now they must adhere to corporate security and data governance frameworks.

The Enterprise Adoption Curve

Getting Hadoop secure is a basic hurdle most IT and security teams now face. They are tasked with getting a handle on Hadoop security — but more importantly applying existing data governance and compliance controls to the cluster. Like all too many security projects, these initiatives to secure existing installations are coming in late. We spoke with many people responsible for Hadoop security, but who are *not* fully versant in how Hadoop works or how to manage it — much less how to secure it. The most common questions we get from customers are simple ones like "How *can* I secure Hadoop?" and "How can I map existing data governance policies to NoSQL platforms?"

It has been about four years since we published our research paper on Securing Big Data — one of the more popular papers we've ever written — but the number of questions has increased over time. And no wonder — adoption has been much faster than we expected. We see hundreds of new big data projects popping up. At the time we wrote the original paper, security for Hadoop clusters was something of a barren wasteland. Hadoop did not include basic controls for data protection; and most third-party tools could not scale along with NoSQL and so were little use to developers. And things did not look likely to get better any time soon as leaders of NoSQL firms directed resources to improving performance and scalability rather than security. Early versions of Hadoop we evaluated did not even require an administrative password!

Hadoop has (mostly) reached security parity with the relational platforms of old, and that's saying a lot given their 20-year head start.

The good news is that we were wrong about the pace of security innovation. Hadoop's functional maturity improvements over the last few years have been matched by much better security and governance capabilities. As large enterprises started to embrace this technology platform, they pushed the vendors of enterprise Hadoop distributions, and made security and compliance non-negotiable components of their minimum requirements. Between the open source community and the commercial vendors, they delivered the majority of the missing pieces. Now security controls are not only *available*, but for most requirements *more than one option*.

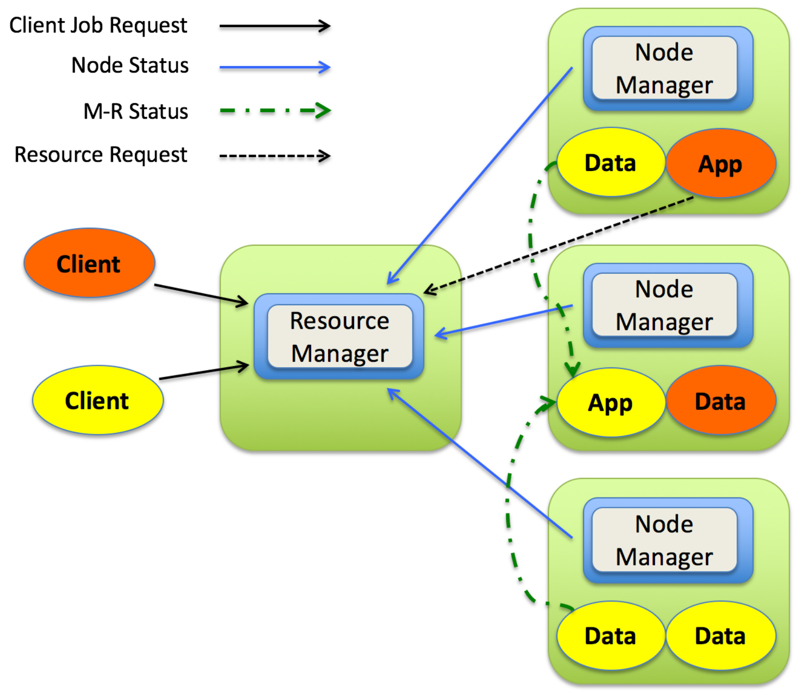
There remain gaps in monitoring and assessment capabilities, but Hadoop has (mostly) reached security parity with the relational platforms of old, and that's saying a lot given their 20-year head start. Because of this rapid advancement, a fresh review of Hadoop security is in order. The reminder of this paper will provide a brief overview of Hadoop's architecture, and highlight security challenges for those not yet fully versed in how clusters are built. We will then offer a fairly technical set of strategic and tactical responses to address these challenges. Our goal is to help those tasked with Hadoop security address risks to the cluster, as well as build a governance framework to support operational requirements.

Architecture and Composition

Our goal for this section is to succinctly outline what Hadoop clusters look like, how they are assembled, and how they are used. This facilitate understanding of the security challenges, along with which sort of protections can secure them. Developers and data scientists continue to stretch system performance and scalability, using customized combinations of open source and commercial products, so there is really no such thing as a 'standard' Hadoop deployment. For example, some place all of their data into a single Hadoop 'Data Lake' supporting dozens of applications, while others leverage multiple clusters, each tuned for the specific business requirement.

Some of you reading this are already familiar with the architecture and component stack of a Hadoop cluster. You may be asking, "Why are we reviewing these basics?" To understand threats and appropriate responses, one must first understand how all the pieces work together. Not everyone reading this guide is familiar with the Hadoop framework, and while it's not essential for readers to understand what each component does, it is important to understand each component interface as an attack target. Each component offers attacker a specific set of potential exploits, while defenders have a corresponding set of options for attack detection and prevention. Understanding architecture and cluster composition is the first step to putting together your security strategy.

The following is a simplified view of Apache Hadoop's underlying MapReduce system:



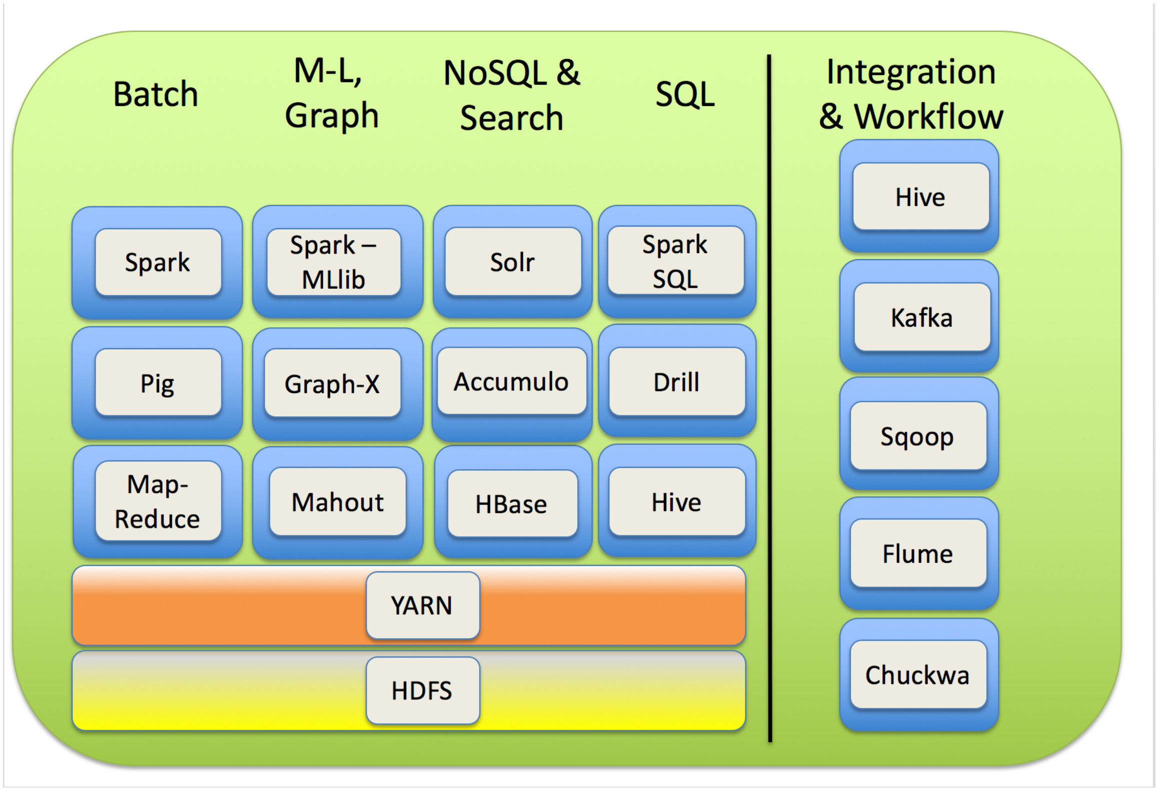
Architecture and Data Flow

Hadoop has been wildly successful because it scales extraordinarily well, can be configured to handle a wide variety of use cases, and is incredibly inexpensive compared to older — mostly proprietary — data warehouse alternatives. Which is all another way of saying Hadoop is cheap, fast, and flexible. To see why and how it scales, take a look at a Hadoop cluster architecture, illustrated in the above diagram.

This architecture promotes scaling and performance. It supports parallel processing, with additional nodes providing 'horizontal' scalability. This architecture is also inherently multi-tenant, supporting multiple client applications across one or more file groups. There are several things to note here from a security perspective. There are many moving parts — each node communicates with its peers to ensure that data is properly replicated, nodes are on-line and functional, storage is optimized, and application requests are being processed. Each line in the diagram is a trust relationship utilizing a communication protocol.

The Hadoop Framework

To appreciate Hadoop's flexibility you need to understand that clusters can be *fully* customized. For those of you new to Hadoop it may help to think of the Hadoop framework as a 'stack', much like a LAMP stack, only on steroids. With a jetpack and night-vision googles. The extent to which you can mix and add components is incredible. While HBase is often used on top of HDFS, you might choose a different style of search, such as Solr. You can use Sqoop to provide relational data access, or leverage Pig for high level MapReduce functions. You can select different SQL query



engines — with Spark, Drill, Impala, and Hive all accommodating SQL queries. This modularity offers great flexibility to assemble and tailor clusters to perform exactly as desired.

It's not just that Hadoop can handle data processing in different ways, or that you can adjust its performance characteristics, alter scheduling, or add input data parsing. A Hadoop cluster can be tailored to fit the exact needs of your business or applications. Need something different? Add a new module. You can design a cluster to satisfy your usability, scalability, and performance goals. You can tailor it to specific types of data, or add modules to facilitate analysis of certain data sets.

But flexibility brings complexity: A dynamic framework of features, functions, and access points makes security more difficult. Each option brings its own security options and issues. Each module runs a specific version of code, has its own configuration, and may require independent authentication to work in the cluster. Many pieces must work in tandem here to process data, so each requires its own security review.

The good news is that security, auditing, governance, and configuration management can *also* be added as modules.

Systemic Security

Now that we have sketched out the elements a Hadoop cluster, and what one looks like, let's talk threats. We need to consider both the infrastructure itself and the data under management. Given the complexity of a Hadoop cluster, the task is closer to *securing an entire set of applications* than a something like a simple relational database. All the features that provide flexibility, scalability, performance, and openness create specific security challenges. So here are some specific facets of clustered systems attackers will target.

- **Data access & ownership:** Role-based access is central to most RDBMS and data warehouse security schemes, and Hadoop is no different. Relational and quasi-relational platforms include roles, groups, schemas, label security, and various other facilities for limiting user access to subsets of available data. Today Hadoop offers full integration with identity stores, along with role-based facilities to divide up data access between groups of users. That said, authentication and authorization require cooperation between the application designer and the IT team managing the cluster. Leveraging existing Active Directory or LDAP services helps tremendously with defining user identities, and predefined roles may be available for limiting access to sensitive data.
- **Data at rest protection:** The standard for protecting data at rest is encryption, which protects against attempts to access data outside established application interfaces. With Hadoop systems we worry about people stealing archives or directly reading files from disk, and encryption at the file or HDFS layer ensures files are protected against direct access by users as only the file services are supplied with the encryption keys. Apache offers HDFS encryption as an option; this is a major advance, and is bundled with the Hadoop distribution. Some commercial Hadoop vendors, as well as commercial third parties products, have advanced the state of the art in transparent encryption options for both HDFS and non-HDFS file formats. These solutions provide key management as well.
- **Multi-tenancy:** Hadoop is commonly used to serve multiple applications and 'tenants', each of which may be from different groups with one firm, or altogether different companies. Typically one tenant's data is not shared with other tenants, but you must implement a security control to ensure privacy. Some firms use Access Control Entries (ACE) or Access Control Lists (ACL) — both essentially file permission constructs— to ensure one tenant cannot read another's data. Still others leverage what are called 'encryption zones' built into native HDFS and some third-party transparent encryption products. Essentially each tenant has defined zones — file groups or even individual files — where each defined zone is encrypted with a different key to ensure data privacy.

- **Inter-node communication:** Hadoop and the vast majority of distributions (Cassandra, MongoDB, Couchbase, etc.) don't communicate securely by default — they use unencrypted RPC over TCP/IP. TLS and SSL capabilities are bundled in big data distributions, but not always used between client applications and the cluster resource manager, and seldom for inter-node communication. This leaves data in transit, along with application queries, accessible for inspection and tampering.
- **Client interaction:** Clients interact with the resource manager and nodes. Gateway services can be created to load data, but clients communicate *directly* with both resource managers and individual data nodes. Compromised clients may send malicious data or link to services. This facilitates efficient communication but makes it difficult to protect nodes from clients, clients from nodes, and even name servers from nodes. Worse, the distribution of self-organizing nodes is a poor fit for security tools such as gateways, firewalls, and monitors.
- **Distributed nodes:** One of the key advantages of big data is the old truism: "Moving computation is cheaper than moving data." Data is processed wherever resources are available, enabling massively parallel computation. Unfortunately this produces complicated environments with lots of attack surface. With so many moving parts it is difficult to verify consistency or security across a highly distributed cluster of (possibly heterogeneous) platforms. Patching, configuration management, node identity, and data at rest protection — and consistent deployment of each — are all issues.

Operational Security

Beyond the systemic Hadoop issues discussed in the last section, IT teams expect some common tools and services which are common across enterprise applications. That includes "turning the dials" on configuration management, vulnerability assessment, policy management, and maintaining patch levels across a complex assembly of supporting modules. The day-to-day processes IT managers follow to ensure typical application platforms are properly configured have evolved over years — core platform capabilities, community contributions, and commercial third-party all help to fill in gaps. IT folks expect best practices, checklists, and validation tools (to verify things like suitable administrative rights and nodes patch status). Hadoop security has come a long way in just a few years, and most of the common issue can now be addressed with some time and effort on the part of IT and security teams. That said, some practitioners still feel available support still lacks maturity in day-to-day operational offerings, and this is where we see most firms struggling.

The following is an overview of the most common threats to Hadoop (and data management systems in general), along with operational controls offering preventative security to close off common attacks.

- **Authentication and authorization:** Identity and authentication are central to any security effort — without them we cannot determine who should have access to data. Fortunately the greatest gains in Hadoop security have been in identity and access management. This is largely thanks to providers of enterprise Hadoop distributions, who have performed much of the integration and setup work. We have evolved from default configurations offering *no* authentication options to fully integrated LDAP, Active Directory, Kerberos, and X.509 based options. By leveraging these capabilities we can use established roles for authorization mapping, and sometimes extend it to fine-grained authorization services like Apache Sentry, or custom authorization mapping controlled from within the calling application.

- **Administrative data access:** Most organizations have platform (*i.e.:* OS) administrators and Hadoop administrators, both with access to the cluster's files. To provide separation of duties — to ensure administrators cannot view content — a facility is needed to segregate administrative roles and restrict unwanted access to a minimum. Direct access to files or data is commonly addressed through a combination of role based-authorization, access control lists, file permissions, and segregation of administrative roles — such as with separate administrative accounts bearing different roles and credentials. This provides basic protection, but cannot protect unauthorized access to archived or snapshot content. Stronger security requires a combination of data encryption and key management services, with unique keys for each application or cluster, as provided with transparent file or HDFS encryption.

- **Configuration and patch management:** With a cluster of servers, which may have hundreds of nodes, it is common to unintentionally run different configurations and patch levels at one time. As nodes are added we see configuration skew. Keeping track of revisions is difficult, and existing configuration management tools only cover the underlying platforms. Issuing encryption keys, certificates, keeping open-source libraries up to date, avoiding *ad hoc* configuration changes, ensuring file permissions are set correctly, and ensuring TLS is correctly configured are some of the common issues. NoSQL systems do not yet have counterparts for the configuration management tools available for other IT platforms, and while Hadoop now has management tools, recommended configurations and pre-deployment checklists are scant, and commercial assessment scanners don't include Hadoop specific checks.

- **Software bundles:** Application and Hadoop stacks are each assembled from many different components. Underlying platforms and file systems also vary — each with its own configuration settings, ownership rights, and patch levels. We see organizations increasingly using source code control systems to handle open source version management and application stack management. Container technologies also help developers bundle up consistent application deployments.

- **Authentication of applications and nodes:** If an attacker can add a new node they control to the cluster, they can exfiltrate data. To authenticate *nodes* (rather than *users*) before they can join a cluster, most firms we speak with either employ X.509 certificates or Kerberos. Both systems can authenticate users as well, but we draw this distinction to underscore the threat of rogue applications or nodes being added to the cluster. Deployment of these services brings risks as well. For example if a Kerberos keytab file can be accessed or duplicated — perhaps using credentials extracted from virtual image files or snapshots — a node's identity can be forged. Certificate-based identity options implicitly complicate setup and deployment, but properly deployed they can provide strong authentication and improve security.

- **Audit and logging:** If you suspect someone has breached your cluster, can you detect it, or trace back to the root cause? You need an activity record, usually from an event log. A variety of add-on logging capabilities are available, both open source and commercial. Scribe and LogStash are open source tools which integrate into most big data environments, as do a number of commercial products. You can leverage the cluster to store its own logs, but many security professionals worry an attacker can cover their tracks by deleting or modifying log entries. For this reason most firms leverage other dedicated platforms like a SIEM or Splunk to quickly stream logs to another repository. Also note some logging options do not provide sufficient information for an auditor to determine exactly what actions occurred. You will need to verify that your logs are configured to capture both the correct event types and sufficient information to determine user actions. A user ID and IP address are insufficient — you also need to know which queries were issued.

- **Monitoring, filtering, and blocking:** There are no built-in monitoring tools to detect misuse or block malicious queries. There isn't even a consensus yet on what a malicious big data query *looks like* — aside from crappy queries written by bad programmers. We are just seeing the first viable releases of Hadoop activity monitoring tools. No longer the "after-market speed regulators" they once were, current tools typically embedded into a service like Hive or Spark to capture queries. Usage of SQL queries has blossomed in the last couple years, so we can now leverage database activity monitoring technologies to flag or even block misuse. These tools are still very new, but the approach has proven effective on relational platforms, so NoSQL implementations should improve with time.

- **API security:** Big data cluster APIs need to be protected from code and command injection, buffer overflow attacks, and all the other usual web service attacks. This responsibility typically lands on the applications using the cluster, but not always. Common security controls include integration with directory services, mapping OAuth tokens to API services, filtering requests, input validation, and managing policies across nodes. Some people leverage API gateways and whitelist allowable application requests. Again, a handful of off-the-shelf solutions can help address API security, but most options are based on a gateway funneling all users and requests through a single interface (choke-point). Fortunately modern DevOps techniques for application stack patching and pre-deployment validation are proving effective at addressing application and cluster security issues. There are a great many API security issues, but a full consideration is beyond our scope for this paper.

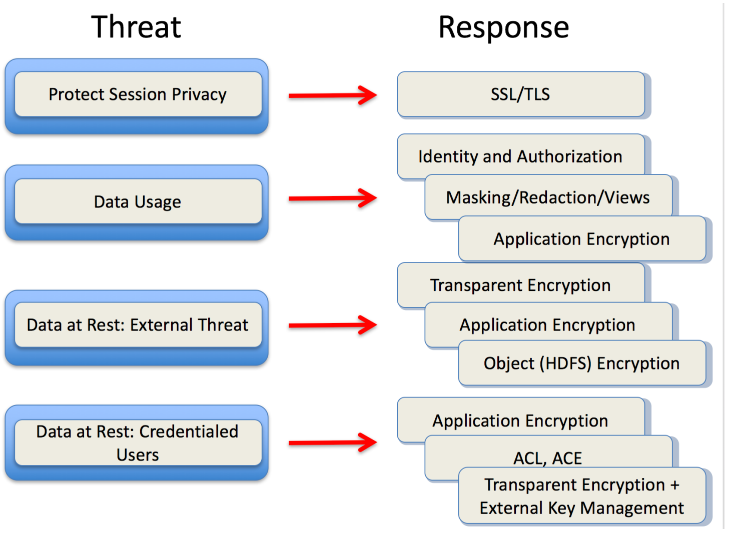
Architecting for Security

Members of the open source community and the vendors of commercial Hadoop distributions talk about security as a collection of basic capabilities. Authentication, authorization, encryption, key management, and logging are the commonly cited 'pillars' upon which you build cluster security. Indeed these are the foundational elements of a good Hadoop security model. And it's likely that you'll implement all of these functions to some degree. But assembling these technologies into a cohesive security strategy requires additional planning. It's not enough to have these tools installed, but how and where you deploy them is important. And you need to understand what security gaps remain after you have the basic capabilities in place, if these gaps must be addressed, and what tools meet your requirements.

The easiest way to communicate security strategies is to illustrate what specific security technologies are used for. We find mapping a problem to potential solutions enables people to understand which security pieces they need to meet particular challenges. It really helps organizations piece together a complete strategy — typically choosing to leverage what they have and are familiar with first, then adding the missing bits later.

Systemic Threat-Response Models

The following diagram shows specific options at your disposal to help you select 'preventative' security measures.

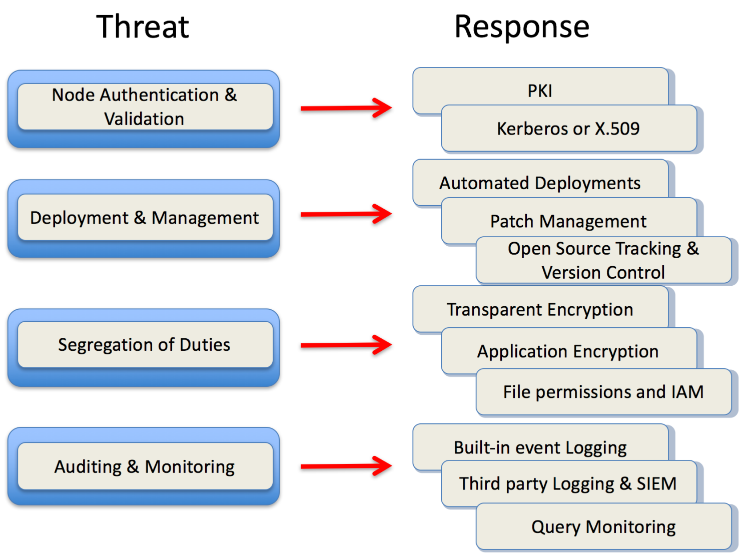


One or more security countermeasures are available to mitigate each threat identified in our discussion of threats facing Hadoop clusters. If your goal is to protect session privacy — whether between clients and data nodes or for inter-node communication — Transport Layer Security (TLS) is your first choice. This usage was unheard of in 2012, but since then about 25% of the companies we spoke with have implemented SSL or TLS for inter-node communication — not just between applications and name servers. Transport encryption protects all communication from access or modification by attackers, but we won't lie — it is a huge pain to implement and get certificate management right. Some firms instead use network segmentation and firewalls to ensure that attackers cannot access network traffic. This is less robust but much easier. Some clusters are deployed to third-party cloud services where virtualized network services make sniffing nearly impossible — these companies typically chose not to encrypt internal cluster communications running within a specific AWS security zone.

Full integration with existing Active Directory and LDAP identity stores, something that was difficult to do in 2012, has helped IT get a handle on managing who get's access to what data in Hadoop clusters. Leverage existing roles and management infrastructure not only aids role based access control, but requires no additional learning curve for admins. And with other tools at our disposal, such as Apache Ranger or Sentry, we have a means to implement more precise authorization controls over sensitive data.

Operational Threat-Response Models

Let's switch to our second set of Threat/Response models, focused on the day-to-day management issues facing Hadoop clusters. This graphic is hardly complete, but should provide a good overview of what we are trying to accomplish, helping you review the options at your disposal. Our goal is to ensure you are aware of the risks, and to point out that you have choices for addressing each specific threat. Each option offers different advantages, with different costs to fully implement.



There are a couple important points to note on the second graphic. First, we don't see it often, but a handful of organizations encrypt sensitive data elements at the application layer, storing information as encrypted elements. This way the application manages decryption and keys, and can offer additional controls over who can see which information. If your application already approaches encryption in this way, great. However, retrofitting application-layer encryption into an existing platform is so difficult we usually steer companies away from this option. If you need greater control than file based encryption provides, dynamic masking, tokenization and labeling are three models which can protect data from unwanted inspection and still provide users with meaningful results. These technologies offer precise control over which data is displayed to which users, and can be easily built into existing clusters to enforce security and compliance.

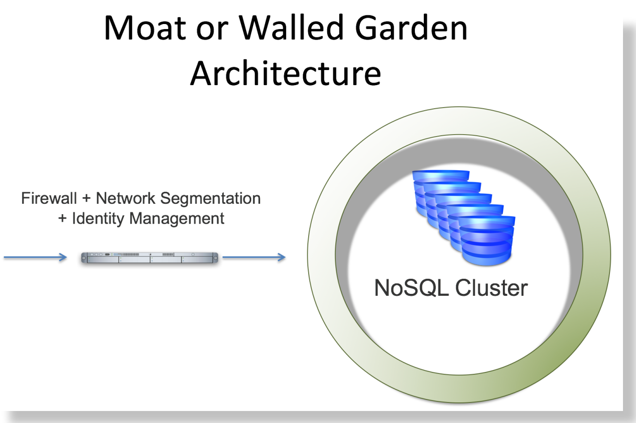
For deeper technical analysis on the technologies listed above — see our other technical papers Understanding Database Encryption (which covers both Hadoop clusters and relational stores), Understanding Data Masking, and Understanding and Selecting a Key Management Solution.

Security Architectures

The following security architectures are very helpful for conceptualizing how you want to approach cluster security. And they are *very* helpful to get a handle on resource allocation: which approach is your IT team comfortable managing, and which tools do you have funds to acquire? That said, the reality is that firms no longer adhere wholly to any single model — most use a combination of two. Some firms we interviewed use application gateways to validate requests, and IAM and transparent encryption to provide administrative segregation of duties on the back end. In another case, the highly multi-tenant nature of the cluster meant they relied heavily on TLS security for session privacy, and implemented dynamic controls (masking, tokenization, and redaction) for fine-grained control over data.

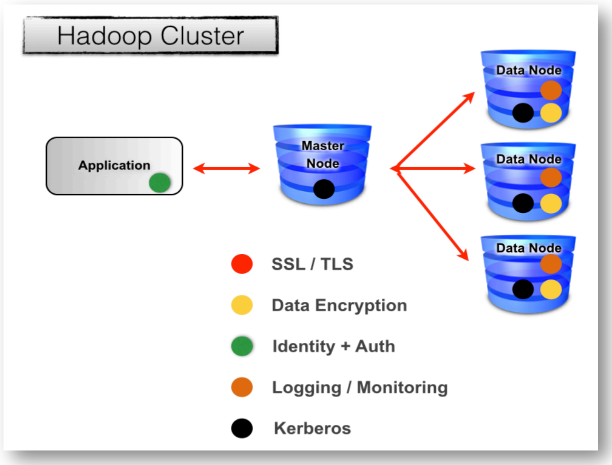
Walled Garden

The most common approach today is a "walled garden" security model, similar to the 'moat' model from mainframe security: place the entire cluster on its own network, and tightly control logical access through firewalls or API gateways, using access controls for user or application authentication. In practice this model provides virtually no security *within* the Hadoop cluster — data and infrastructure security depend on the outer "protective shell" of the network and applications that surround it. The advantage is simplicity: any firm can implement this model with existing tools and skills, without performance or functional degradation to the Hadoop cluster. On the downside security is fragile: once a failure of the firewall or application occurs, the system is exposed. This model also does not prevent credentialed users from misusing the system or viewing/modifying data stored in the cluster. For organizations not particularly worried about security, this is a simple and cost-effective approach.


Moat or Walled Garden Architecture
Firewall + Network Segmentation + Identity Management
NoSQL Cluster

Cluster Security

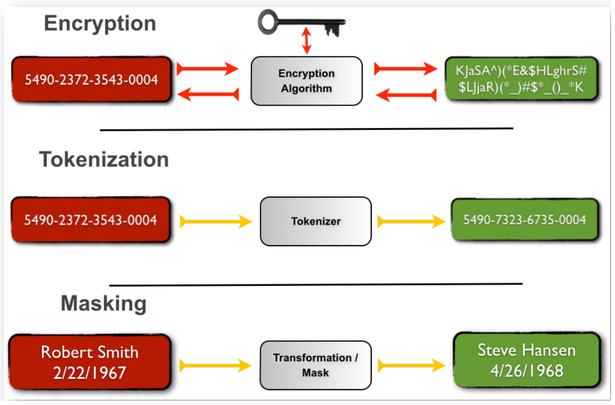
Unlike relational databases which function like black boxes, Hadoop exposes its innards to the network. Inter-node communication, replication and other cluster functions take place between many machines, using different types of services. For the most effective protection, building security into cluster operations is critical. This approach leverages security tools built into – or third-party products integrated into – the NoSQL cluster. This security is systemic and built to be part of the base cluster architecture.



Tools may include SSL/TLS for secure communication, Kerberos for node authentication, transparent encryption for data-at-rest security, and identity and authorization (groups & roles) management, just to name a few. This approach is more difficult because there are a lot more moving parts and areas where some skill is required. Setting up multiple security functions targeted at specific risks takes time. And third-party security tools can be expensive. But they can effectively secure clusters from attackers, rogue administrators, and witless application programmers. It's the most effective and comprehensive approach to Hadoop security.

Data Centric Security

Big data systems typically share data from dozens of sources. Firms do not always know where their data is going, or what security controls are in place once it is stored, they have taken steps to protect their data regardless of where it is used. This model is called data-centric security because the controls are part of the data, or in some cases of the query-processing layer.



The three basic tools that support data-centric security are tokenization, masking, and data element encryption – the later commonly implemented as Format Preserving Encryption.

You can think of a data token like a subway or arcade token: it has no cash value but can be used to ride a train or play a game. In this case a data token is provided in lieu of sensitive data – credit card processing systems often substitute tokens for real credit card numbers. A data token has no intrinsic value — it only references the original value in a token store. Masking is another popular tool used to protect data elements while retaining the aggregate value of a data set. For example some firms substitute an individual's Social Security number with a random number, or replace customer names with random names from a phone book, or replace date values with random dates within a range. This way the original *sensitive* data value is removed entirely from query results, but the value of the *data set* is preserved for analytics. Alternatively data elements can be encrypted and passed without fear of compromise; only legitimate users with the right encryption keys can view the original values.

The data-centric security model provides a great deal of security when the systems that process data cannot be fully trusted, or in cases we don't want to share data with users. For example, when valid users are on mobile devices or information is being shared with partners. Dynamic forms of data centric security alters what information is presented based upon external factors, such as geo-location, device type, or even time of day. Static masking can be used to completely remove sensitive data from clusters yet still allow for meaningful data analysis. But a data-centric security model requires careful planning and tool selection — it is more about *information lifecycle management*. You define the controls over data usage, providing a surrogate when conditions are suspect. Short of deleting sensitive data, this is the best model when you must populate a big data cluster for analysis work but cannot guarantee its security.

Enterprise Security Options

Security vendors will tell you both attacks on corporate IT systems and data breaches are prevalent, so with gobs of data under management, Hadoop provides a tempting target for 'hackers'. All of which is true, but we have not yet seen Hadoop blamed in a major data breach. So this sort of FUD carries little weight with IT Operations. But *security is still a requirement*! As sensitive information, customer data, medical histories, intellectual property, and just about every type of data used in enterprise computing is now commonly used in Hadoop clusters, the 'C' word (Compliance) has become part of the vocabulary. One of the major changes we have seen over the last couple years has been Hadoop becoming business-critical infrastructure. Another, which flows directly from the first, is IT being tasked with bringing *existing* clusters in line with enterprise compliance requirements.

This is challenging because a fresh install of Hadoop suffers all the same weak points as traditional IT systems, so it takes work to establish security. And more to create policies and reports for the compliance team. For clusters already up and running, you need to choose technologies and a deployment roadmap that do not disturb ongoing operations. Additionally, the in-house tools you use to secure things like SAP, or the SIEM infrastructure you use for compliance reporting, may be inadequate or unsuitable for NoSQL.

Embedded Security

The number of security solutions compatible with or even designed for Hadoop has been the biggest change since 2012. All the major security pillars — authentication, authorization, encryption, key management and configuration management — are available and viable; in many cases suitable open source options exist alongside with the commercial ones. The biggest advances came from firms providing enterprise distributions of Hadoop. They have purchased, built, and in many cases contributed back to the open source community, security tools which provide the basics of cluster security. Reviewing the threat-response models discussed previously, *there exists a* compensating security control for each threat vector. Better still, the commercial Hadoop vendors have done a lot of the integration legwork for services like Kerberos, taking much of the pain out of deployments.

Here are some components and functions that were not available — or not truly viable — in 2012.

- **LDAP/AD integration** — AD and LDAP integration existed in 2012, but both options have been greatly improved, and they are easier to integrate. This area has received perhaps the most attention, and some commercial platforms make integration as simple as filling in a setup wizard. The benefits are obvious — firms now leverage existing access and authorization schemes, and defer user and role management to external sources.

- **Apache Ranger** — Ranger is a policy administration tool for Hadoop clusters. It includes a broad set of management functions, including auditing, key management, and fine grained data access policies across HDFS, Hive, YARN, Solr, Kafka and other modules. Ranger is one of the few tools to offer a single, central management view for security policies. Better still, policies are context aware, so it understands to set file and directory policies in HDSF, SQL policies in Hive, and so on.

This helps with data governance and compliance because administrators can now define how data may be accessed, and how certain modules should function.

- **HDFS Encryption** — HDFS offers 'transparent' encryption embedded within the Hadoop file system. This means data is encrypted as it is stored into the file system, transparently, without modification to the applications that use the cluster. HDFS encryption supports the concept of encryption zones; essentially these zones are directories in HDFS where all content, meaning every file and subdirectory in it, is encrypted. Each zone can use a different key if desired. This is an important feature to support tenant data privacy in multi-tenant clusters. HDFS can be used with Hadoop's Key Management Service (KMS), or integrated with third party key management services.
- **Apache Knox** — You can think of Knox as a Hadoop firewall. More precisely it is an API gateway. It handles HTTP and RESTful requests, enforcing authentication and usage policies on inbound requests and blocking everything else. Knox can be used as a virtual 'moat' around a cluster, or combined with network segmentation to further reduce network attack surface.
- **Apache Atlas** — Atlas is a proposed open source governance framework for Hadoop. It enables annotation of files and tables, establishment of relationships between data sets, and can even import metadata from other sources. From a compliance perspective, these features are helpful for reporting, data discovery and access control. Atlas is new and we expect to see significant maturation in coming years, but it already offers valuable tools for basic data governance and reporting.
- **Apache Ambari** — Ambari is a facility for provisioning and managing Hadoop clusters. It helps administrators set configurations and propagate changes to the entire cluster. During interviews we only spoke to two firms using this capability, but received positive feedback from both. Additionally we spoke with a handful of companies who had written their own configuration and launch scripts, with pre-deployment validation checks, mostly for cloud and virtual machine deployments. Homegrown scripts are more time-consuming but offer broader and deeper capabilities, with each function orchestrated within IT operational processes (*e.g.,* continuous deployment, failure recovery, & DevOps). For most organizations Ambari's ability to get up and running quickly, with consistent cluster management, is a big win and makes it a good choice.
- **Monitoring** — Monitoring takes the concept of logging two step further, performing real time analysis on events, and alerting when misuse is detected. Hive, PIQL, Impala, Spark SQL and similar modules offer SQL or pseudo-SQL syntax. This enables you to leverage activity monitoring, dynamic masking, redaction, and tokenization technologies originally developed for relational platforms. The result is that we can both alert and block on misuse, or provide fine-grained authorization (beyond role-based access control) by altering queries or query result sets based on user metadata. And because these technologies examine queries they offer an application-centric view of events which log files may not capture.

Your first step in addressing these compliance concerns is mapping your existing governance requirements to a Hadoop cluster, then deciding on suitable technologies to meet data and IT security requirements. Next you will deploy technologies that provide security and reporting functions, and setting up the policies to enforce usage controls or detect misuse. Since 2012 many technologies have become available to address common threats *without* killing scalability and performance, so there is no need to reinvent the wheel. But you will need to assemble these technologies into a coherent system.

Technical Recommendations

Following are our recommendations for addressing security issues with Hadoop. Last time we joked that many security tools broke Hadoop scalability — they secured clusters by making them unusable. Fast forward four years, and both commercial and open source technologies have advanced considerably — to address the threats you're worried about and fit into the Hadoop architecture. This makes it less likely a security tool will compromise cluster performance or scalability, and puts the integration hassles of old largely behind us.

The rapid technical advancement of open source projects have caused an about-face in where we look for security capabilities. We are no longer focused only on commercial third-party security tools — it is now also important to consider open source, which has helped to close the major Hadoop security gaps. That said, many of these new capabilities are still somewhat immature. You still need to work through a tool selection process based on your needs, and then integrate and configure.

Selection Requirements

Security in and around Hadoop is still relatively young, so not all security tools will work within your clustered environment. We still see vendors proudly showing off old products for other back-office systems and relational databases, updating their banners with Sharpie to say "Big Data Ready!" To ensure you are not duped by security vendors, you still need to do your homework: evaluate products to ensure they are architecturally and environmentally consistent with the cluster architecture, rather than in fundamental conflict with Hadoop's architecture.

Any security control used for Hadoop must meet the following requirements:

1. It must not compromise the basic functionality of the cluster.
2. It should scale in the same manner as the cluster.
3. It should address a security threat to the Hadoop cluster or data stored within it.

Our Recommendations

Our big data security recommendations boil down to a handful of standard tools which can establish a secure baseline Hadoop environment.

1. **Use Kerberos for node authentication:** At the outset of this project we thought we would no longer recommend Kerberos. We expected implementation and deployment challenges to drive customers in a different direction. We were completely wrong. Our research shows that adoption has increased considerably over the last 24 months, specifically in response to enterprise distributions of Hadoop streamlining Kerberos integration, making it reasonably easy to deploy. Now, more than ever, Kerberos is used as a cornerstone of cluster security. It remains effective for validating nodes, and in some organizations for authenticating users. It takes time to set up, and you'll need to protect the keytab files, but the investment in time is worth it — both for the security it directly provides and because other security controls piggyback off Kerberos as well. It is one of the most effective security controls at our disposal, it's built into the Hadoop infrastructure, and enterprise bundles make it accessible, so we recommend you use it.
2. **Use file layer encryption:** Simply stated, this is how you will protect data at rest. File encryption protects against two tactics for circumventing application security controls. It prevents administrators and malicious users/tenants from accessing data nodes and directly inspecting files, and renders stolen files and copied disk images unreadable. And encryption is mandatory if you need to satisfy compliance or data governance requirements. While it may be tempting to rely on encrypted SAN/NAS storage, it cannot provide protection from credentialed user access, granular file protection, or multi-key support. File layer encryption provides consistent protection across different platforms regardless of OS, platform, and storage type — with some products even protecting encryption operations in memory. Just as important, encryption meets our requirements for big data security — it is transparent to both Hadoop and calling applications, and scales out as the cluster grows. But you must choose between open source HDFS encryption, OS variants like Linux block encryption, or third-party commercial file and/or HDFS products. Open source products are freely available and offer open source key management support. But HDFS encryption can only protect data on HDFS, leaving other files exposed. Commercial variants which work at the file system layer secure all files. Additionally, open source products offer weaker support for external key management, trusted binaries, and full support contracts than commercial products. Free is always nice, and the quality of the open source tools is not in doubt, but many enterprise customers prefer complete coverage and support. Regardless of which option you choose, this is a mandatory security control.
3. **Use key management:** File encryption is ineffective if an attacker can access the encryption keys. Many big data cluster administrators store keys on local disk drives because it's quick and easy, but that is insecure because keys can be collected by the platform administrator or an attacker. Use a key management service to distribute keys and certificates; this is especially effective when combined with HDFS encryption zones and different keys for each tenant, application, and user. This requires additional setup, and possibly a commercial key management product to scale with your big data environment, but it is critical. Most of the encryption controls we recommend depend on key/certificate security.

4. **Use Apache Ranger:** In our original research we worried most about combining a dozen Hadoop modules, all deployed with *ad hoc* configurations, obscured within the complexities of the cluster, each exposing its own unique attack surface to adversaries. Deployment validation remains at the top of our list of concerns, but Apache Ranger provides a consistent management plane for establishing configurations *and usage* policies to protect data within the cluster.

5. **Automate deployment:** You'll still need to address patching the Hadoop stack, application configuration, managing trusted machine images, and platform discrepancies. Some organizations use automation scripts and source code control; others leverage traditional patch management systems to track revisions; still others have a management nightmare on their hands. We also recommend use of automation tools, such as Chef and Puppet, to orchestrate pre-deployment configuration tasks, assembly from trusted images, patching, issuing keys, and even running tools like vulnerability scanners prior to deployment. Building the scripts and setting up these services takes time up front, but pays for itself in reduced management time and effort later, and ensures that each node comes online with baseline security in place.

6. **Use logging and monitoring:** To perform forensic analysis, diagnose failures, or investigate unusual behavior, you need an activity record. You can leverage built-in Hadoop functions to create event logs, and even use the cluster itself to store events. Tools like LogStash, Log4J and Kafka help with streaming, management, and searching. Plugins are available to stream standardized `syslog` feeds to supporting SIEM platforms or even Splunk, and when done in real time, help protect logs from tampering. We also recommend usage of context-aware monitoring tools — in 2012 no activity monitoring tools worked with big data platforms, but now they do. These capabilities usually plug into supporting modules like Hive and collect all queries, their parameters, and details about the user and/or application issuing the query. This approach goes beyond basic logging to detect misuse and even alter the results users see. These tools can also feed events into native logs, SIEM, or even database activity monitoring tools.

7. **Use secure communication:** Implement secure communication between nodes, and between nodes and applications. This requires an SSL/TLS implementation that actually protects *all* network communications, rather than just a subset. This imposes a small performance penalty on transfer of large data sets, but the burden is shared across all nodes. The real issues are setup, certificate issuance, and configuration.

Encryption, authentication, and platform management tools are greatly improving the security of Hadoop clusters, closing off all the easiest paths attackers have used to steal information or compromise functionality. For some challenges, such as authentication, Hadoop provides excellent integration with Active Directory and LDAP services. Authorization modules and services offer fine-grained control over data access, thankfully moving beyond simple role-based access controls to make application developers' jobs far easier. The Hadoop community has largely embraced security, offering all of the basic security functions one would expect, and done so far faster than we imagined possible in 2012.

Unfortunately when we speak with Hadoop architects and IT managers, we *still* hear that the most popular security model is to hide the entire cluster with network segmentation, and then hope attackers can't get through the application. Hard on the outside, soft and chewy on the inside — great for candy bars and French bread — but not for security. The good news is that almost everyone we spoke with has evolved cluster security to some degree — mostly in response to compliance requirements. Like Hadoop itself, administrators and cluster architects are getting far more sophisticated about security. Most of the people we spoke with have mapped out all these recommended controls, and are taking the next step to satisfy their compliance obligations. Consider the recommendations above a minimum set of preventative security measures. These are easy to recommend — they are simple, cost-effective, and scalable, and they addresses real security deficiencies with big data clusters. Nothing suggested here harms performance, scalability, or functionality. They are more work to set up, but relatively simple to manage and maintain.

We hope you find this research helpful. If you have any questions on this topic, or want to discuss your situation specifically, feel free to send us a note at info@securosis.com or ask via the Securosis blog.

About the Analyst

Adrian Lane, Analyst/CTO

Adrian Lane is a Senior Security Strategist with 25 years of industry experience. He brings over a decade of C-level executive expertise to the Securosis team. Mr. Lane specializes in database security, secure application development and data security. With extensive experience as a member of the vendor community (including positions at Ingres and Oracle), in addition to time as an IT customer in the CIO role, Adrian brings a business-oriented perspective to security implementations. Prior to joining Securosis, Adrian was CTO at database security firm IPLocks, Vice President of Engineering at Touchpoint, and CTO of the secure payment and digital rights management firm Transactor/Brodia. Adrian also blogs for Dark Reading and is a regular contributor to Information Security Magazine. Mr. Lane is a Computer Science graduate of the University of California at Berkeley with post-graduate work in operating systems at Stanford University.

About Securosis

Securosis, LLC is an independent research and analysis firm dedicated to thought leadership, objectivity, and transparency. Our analysts have all held executive level positions and are dedicated to providing high-value, pragmatic advisory services. Our services include:

- **The Securosis Nexus**: The Securosis Nexus is an online environment to help you get your job done better and faster. It provides pragmatic research on security topics that tells you exactly what you need to know, backed with industry-leading expert advice to answer your questions. The Nexus was designed to be fast and easy to use, and to get you the information you need as quickly as possible. Access it at <https://nexus.securosis.com/>.

- **Primary research publishing**: We currently release the vast majority of our research for free through our blog, and archive it in our Research Library. Most of these research documents can be sponsored for distribution on an annual basis. All published materials and presentations meet our strict objectivity requirements and conform to our Totally Transparent Research policy.

- **Research products and strategic advisory services for end users**: Securosis will be introducing a line of research products and inquiry-based subscription services designed to assist end user organizations in accelerating project and program success. Additional advisory projects are also available, including product selection assistance, technology and architecture strategy, education, security management evaluations, and risk assessment.

- **Retainer services for vendors**: Although we will accept briefings from anyone, some vendors opt for a tighter, ongoing relationship. We offer a number of flexible retainer packages. Services available as part of a retainer package include market and product analysis and strategy, technology guidance, product evaluation, and merger and acquisition assessment. Even with paid clients, we maintain our strict objectivity and confidentiality requirements. More information on our retainer services (PDF) is available.

- **External speaking and editorial**: Securosis analysts frequently speak at industry events, give online presentations, and write and speak for a variety of publications and media.

- **Other expert services**: Securosis analysts are available for other services as well, including Strategic Advisory Days, Strategy Consulting engagements, and Investor Services. These tend to be customized to meet a client's particular requirements.

Our clients range from stealth startups to some of the best known technology vendors and end users. Clients include large financial institutions, institutional investors, mid-sized enterprises, and major security vendors.

Additionally, Securosis partners with security testing labs to provide unique product evaluations that combine in-depth technical analysis with high-level product, architecture, and market analysis. For more information about Securosis, visit our website: <http://securosis.com/>.