BIG DATA AND DATA-CENTRIC SECURITY
EXECUTIVE SUMMARY

You don’t need to be at the reins of a large business to understand the table stakes of data protection and information security. Recent events like the data breach at Neiman Marcus and the point-of-sale infection at Target highlight the far-reaching implications on business reputation, consumer trust, and personal privacy that arise from inadequate safeguarding of private information.

It is important to recognize that when you consolidate disparate data sources to a single location, you are also compounding the risks of adequately protecting that data, and amplifying the stakes of security. The act of satisfying security and compliance requirements has been a major obstacle for successfully rolling Big Data in a production enterprise scenario.

We are faced with a harsh reality: the dissolution of the “trusted zone”. It is no longer the case that there exists a well-defined secure perimeter that we can identify and defend. Because of this, we must rethink our approach to guaranteeing security and privacy.

The best approach for tackling this challenge is Data-Centric Security, in which every field of data has security information embedded within it that determines its access and governance. This approach maximizes security protections while minimizing overhead.

Sqrrl Enterprise is the only Big Data offering with Data-Centric Security, a philosophy we adopted on day one. Sqrrl’s Data-Centric Security implementation includes cell-level security labeling and enforcement, encryption at rest, encryption in motion, an automated data labeling engine, a security policy specification engine, custom secure search indexing, and an audit facility.

Data-Centric Security is the key enabler for comfortably consolidating disparate data sources and developing powerful Big Data applications that reach only the intended audience.
# Table of Contents

- **EXECUTIVE SUMMARY**................................. 2
- **I. INTRODUCTION**.............................................. 4
- **II. SECURITY IN A BIG DATA WORLD**.......................... 4
- **III. DATA-CENTRIC SECURITY**.................................... 5
  - **III.A REFERENCE ARCHITECTURE FOR DATA-CENTRIC SECURITY** 5
  - **III.B REFERENCE IMPLEMENTATION FOR DATA-CENTRIC SECURITY** 6
  - **III.C CELL-LEVEL SECURITY ENFORCEMENT**.......................... 8
  - **III.D AUTOMATED DATA LABELING**..................................... 9
  - **III.E SECURITY POLICY SPECIFICATION**................................ 11
  - **III.F CRYPTOGRAPHY**............................................. 12
  - **III.G SECURE SEARCH**........................................... 13
  - **III.H AUDIT**.................................................. 14
  - **III.I DATA-CENTRIC SECURITY COMPARISON**.......................... 14
- **IV. CONCLUSION**.................................................. 15
I. INTRODUCTION

It has been said that we entered the age of Big Data when the opportunity cost of throwing data away exceeded the monetary cost of keeping it. We collectively decided it is worth it to keep as much data as possible, for fear of missing out on that data diamond in the rough. This decision has led to the emergence of the ‘data lake’ or ‘data hub’ – a massive, centralized, shared data repository that can be accessed on-demand by any user, system, or application.

It is important to recognize that when you consolidate disparate data sources to a single location, you are also compounding the risks of adequately protecting that data, and amplifying the stakes of security. Satisfying security and compliance requirements has been a major obstacle for successfully rolling out a Big Data lake in a production enterprise scenario. This reality is partly on account of immature tooling to address these needs, but also because traditional approaches to data protection are difficult and expensive at scale.

At Sqrrl, we advocate a Data-Centric Security approach by which every field in a data record is individually protected and has security information embedded within it that determines its access and governance. Our Data-Centric Security implementation includes cell-level security labeling and enforcement, encryption at rest, encryption in motion, an automated data labeling engine, a security policy specification engine, custom secure search indexing, and an audit facility.

Data-Centric Security is the key enabler for comfortably consolidating disparate data sources and developing powerful Big Data applications that reach only the intended audience.

II. SECURITY IN A BIG DATA WORLD

You don’t need to be at the reins of a large business to understand the table stakes of data protection and information security. Recent events like the data breach at Neiman Marcus and the point-of-sale infection at Target highlight the far-reaching implications on business reputation, consumer trust, and personal privacy that arise from inadequate safeguarding of private information.

Over the years, security vendors have emerged with products and solutions to protect and monitor critical data assets, and businesses have demonstrated their willingness to invest in these areas. Whether it be on account of corporate governance, regulatory requirements, or simply hedging business risk, security-aware enterprises have implemented policies and procedures for areas such as identity management, access entitlement, encryption, and audit.

Traditional approaches to information security have usually involved a “secure the perimeter” approach, with multiple layers of authorization checks that must succeed before the access request is granted. This model has worked well, and will continue to
have its place in the future; however, in our rapidly evolving world, the “perimeter” is no longer a solid, easily recognizable barrier.

On account of phenomena like Cloud Computing and Bring Your Own Device, businesses no longer have full authority over what data and information flows into their systems. The emergence of sophisticated attacks like Advanced Persistent Threats (APTs), and the occasional malicious actor within a business’ employ, place more strain on monitoring data flows leaving the firewall. Taken together, the reality is that the safe haven of a corporate intranet has disintegrated - there is no longer a “trusted zone”- at least the way we are used to it.

This fact is why a Data-Centric Security approach is so important. Data-Centric Security narrows the trust boundary as much as possible, filling the gap that traditional approaches have difficulty satisfying.

III. DATA-CENTRIC SECURITY

Chances are that you are not in the core business of “security”, but rather, securing your data is a means to safely running business operations. Data-Centric Security offers many benefits to businesses of all types.

Traditional security infrastructure and applications are expensive at scale. This expensiveness measure is not just a monetary one. There are also performance implications and scalability limits to funneling all information through a centralized security checkpoint. Furthermore, there is also process overhead to updating security implementations to adhere to the latest regulatory or compliance requirements.

Additionally, security can sometimes be an obstacle to the intended functionality of a particular system. Techniques such as data masking or obfuscation can detract from the utility of the application, especially in the case of performing analytics. The right approach to Data-Centric Security mitigates these challenges.

Data-Centric Security is a complement to, not a replacement for, traditional security. To the extent that it is possible, securing the perimeter will always be a good idea. At the same time, you can rest easier knowing that there is security information embedded within your data, enforced in a distributed fashion, and available throughout the entire data lifecycle.

III.A REFERENCE ARCHITECTURE FOR DATA-CENTRIC SECURITY

The philosophy behind Data-Centric Security involves narrowing the trust boundary and minimizing the amount of external dependencies to the process of granting access to data. In other words, Data-Centric Security is about embedding security information within the data itself. To that end, one must design for certain characteristics along the data acquisition and data retrieval paths, so that access can be checkpointed, validated, and enforced on every read or write operation.
Figure 1 – Data-Centric Security Reference Architecture

 Appropriately, at the center of the reference architecture is data. Embedded within it are certain components: labels for specifying authorization criteria, indexes for ease of retrieval, and cryptography for an extra layer of protection. The logical store for data should serve as an access/policy enforcement point. A comprehensive Data-Centric Security approach will incorporate the appropriate external factors required to build policy (user attributes, environmental attributes, business requirements, and compliance requirements), provide facilities for implementing policy, and ensure that all operations performed against the system are auditable.

III.B REFERENCE IMPLEMENTATION FOR DATA-CENTRIC SECURITY

Keeping in mind the requirements and flows detailed above, we at Sqrrl believe that a Data-Centric Security solution encompasses the following features:

• **Fine-grained, cell-level security enforcement** – the independent access validation of every field of data individually stored in the system
• **Data labeling capability** – the ability to assign visibility labels to data that specify access policy using a set of rules
• **Policy specification capability** – the ability to grant individual or groups of users entitlements to view data that has a particular set of visibility labels
• **Encryption, at-rest and in-motion** – ensuring that data is always protected cryptographically, whether resident on disk or traversing the network
• **Secure search** – ensuring that data is easily retrievable, and that this convenience does not provide a source of data leakage
• **Auditing** – recording every client operation taken against the system

When designing Sqrrl Enterprise, we took the principles of Data-Centric Security and married them with best practices for Big Data application development, including commodity hardware and open-source software. We then layered our advanced functionality, and it manifested in the following way, with Data-Centric Security being enforced throughout:

---

**Figure 2 – Sqrrl Data-Centric Security Reference Implementation**

---

**Figure 3 – Sqrrl Enterprise Technology Stack**

---
The remainder of this paper will elaborate on the individual components of Data-Centric Security, their importance to data storage and application development, and details on how we at Sqrrl have implemented them into our Enterprise product.

III.C CELL-LEVEL SECURITY ENFORCEMENT

At the core of a Data-Centric Security approach is extremely fine-grained access control, the notion that each field of data is individually protected, irrespective of the protections of neighboring fields or the collection that it’s in.

This foundational aspect of Data-Centric Security allows for data to be collected from disparate sources and aggregated into Entity-centric formats. Only with this technique can you break down data silos and at the same time respect granular variance in the underlying security and compliance requirements of the originating systems and applications.

In Sqrrl Enterprise, this fine-grained enforcement is enabled via the capabilities of Apache Accumulo™, the first NoSQL database with native cell-level security, and the only Big Data platform where cell-level security was an integral part of the design, not bolted on.

On every read or write operation Sqrrl enforces security by inspecting a set of visibility labels that are attached to every field of data in the system, as part of its key. Largely inspired by the Google BigTable architecture, an Accumulo datum record has the following structure:

<table>
<thead>
<tr>
<th>Row</th>
<th>Col. Fam.</th>
<th>Col. Qual.</th>
<th>Visibility</th>
<th>Timestamp</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>John Doe</td>
<td>Notes</td>
<td>PCP</td>
<td>PCP_JD</td>
<td>20120912</td>
<td>Patient suffers from an acute...</td>
</tr>
<tr>
<td>John Doe</td>
<td>Test Results</td>
<td>Cholesterol</td>
<td>JD</td>
<td>PCP_JD</td>
<td>20120912</td>
</tr>
<tr>
<td>John Doe</td>
<td>Test Results</td>
<td>Mental Health</td>
<td>JD</td>
<td>PSYCH_JD</td>
<td>20120801</td>
</tr>
<tr>
<td>John Doe</td>
<td>Test Results</td>
<td>X-Ray</td>
<td>JD</td>
<td>PHYS_JD</td>
<td>20120513</td>
</tr>
</tbody>
</table>

**Figure 4 – Accumulo Data Record**

Data Access Semantics

A “cell” in Accumulo is referenced as a 5-tuple: row key, column family, column qualifier, visibility, and timestamp. The “visibility” component of an Accumulo record key provides assurance that only users authorized to access a given set of labels will be allowed to access the underlying data. In the preceding example, only users who have been granted permission to view data labeled “PCP_JD” (John Doe’s Primary Care Physician) will be able to see the Notes:PCP field. The Notes:PCP notation specifies that “Notes” is the column family, and “PCP” is the column qualifier.

Boolean Logic on Data Labels
Also present in the preceding example are Boolean operators on visibility labels. In particular, only users with the labels “JD” (i.e., John Doe) OR “PCP_JD” (John’s Primary Care Physician) will be able to access the Test Results:Cholesterol field.

Accumulo provides the ability to perform certain Boolean expressions on data visibility labels using AND or OR operators.

**Sqrrl’s Cell-Level Implementation**

Sqrrl’s data model allows the user to interact with JSON-style Document and Graph objects, translating these into Accumulo wide-column key/value pairs under the hood. Sqrrl maintains the cell-level security enforcement capabilities of Accumulo in its higher-level development language and data access API. With the ability to create JSON-style Documents with nested fields, Sqrrl also has the ability to have nested visibility labels. Consider the following:

```
{  
  "children[FAM][IRS]": {  
    "current": [  
      { "name": "Johnny" }  
    ],  
    "expecting[FAM]": [  
      { "name": "Baby Girl" }  
    ]  
  }  
}
```

**Figure 5 – Nested Sqrrl Document**

This example defines a document that represents a family. It has a field, children, that represents the children in the family. children has a set of visibility labels assigned to it, FAM|IRS, which specifies that either people in the family (FAM) OR people in the IRS (IRS) should be able to see the list of children in the family.

Under children are two nested fields, current and expecting. Because current has no set of visibility labels explicitly assigned to it, it inherits labels from its parent, children, which are FAM|IRS. On the other hand, expecting has its own label assigned, FAM. In this case, Sqrrl evaluates the two sets of labels with an AND operator, (FAM|IRS) & FAM, or simply FAM. This visibility condition specifies that only the family, and not the IRS, can see that the family is expecting another child.

This core principle of Sqrrl’s Data-Centric Security is the foundation for defining rules and individually protecting data fields in a Big Data application scenario.

**III.D AUTOMATED DATA LABELING**

Above, we discussed how Sqrrl leverages the mature cell-level security enforcement capabilities of Apache Accumulo to enforce access to data based upon labels contained within that data. One area where Apache Accumulo (and other tools that have similar cell-level protection capabilities) falls short is in providing a way to (1) determine which data is sensitive and requires special attention and (2) automate the application of visibility labels onto data.
Sqrrl Enterprise ships with a Data Labeling Engine and rules syntax that makes these tasks easy and practical. The Label Engine will apply a specified set of labels to data that matches a set of criteria as it is ingested into the system. For example, you could define a rule that specifies that if a string looks like a Social Security Number (i.e., it has the format ‘XXX-XX-XXXX’), that this piece of data should be labeled “pii” for Personally Identifiable Information.

The visibility label rules syntax includes the following capabilities:

- Arithmetic Evaluation and Comparison of values (e.g. “apply this label if this field’s value is greater than 5”)
- Regular Expressions
- Boolean Logic (AND, OR, NOT)
- Variable Substitution (e.g. creating a label which is named after some field’s value)

To demonstrate the capabilities of the Label Engine, consider the following JSON document:

```json
{
  "user-id" : "bob5678",
  "mailbox" : {
    "name" : "Bob E. Example",
    "address" : "bob@example.com",
    "num-messages" : "3",
    "messages" : [
      {
        "message-id" : "99074",
        "message" : {
          "from" : "Alice E. Example <alice@example.com>",
          "subject" : "Party on Sunday",
          "importance" : 1,
          "body" : "Do not forget our party on Sunday"
        }
      },
      {
        "message-id" : "129434",
        "message" : {
          "from" : "Dr. Bob Doctor <drbob@example.com>",
          "subject" : "Test Results",
          "importance" : 10,
          "body" : "Everything came back OK. I will see you in the office on Friday."
        }
      },
      {
        "message-id" : "653812",
        "message" : {
          "from" : "Richard T. Lawyer <rlawyer@example.com>",
          "subject" : "Deposition",
          "importance" : 7,
          "body" : "You need to schedule your deposition."
        }
      }
    ]
  }
}
```
Let's say you want to flag the messages that have the highest importance as `veryimportant`. With the Label Engine, the following rule would achieve that goal:

```
APPLY veryimportant to //mailbox/messages/**[/message WHERE
CHILD importance >= 10
```

After running this rule, the JSON in the Figure above would be transformed to the following:

```json
... snipped ...
{
  "message-id": "129434",
  "message@[veryimportant]" : {
    "from": "Dr. Bob Doctor <drbob@example.com>",
    "subject": "Test Results",
    "importance": 10,
    "body": "Everything came back OK.\n\nI will see you in the office on Friday."
  }
... snipped ...
```

Having a way to automatically assign and manage visibility labels on data is a key part to a Data-Centric Security solution. Please refer to the Sqrrl Enterprise User Guide for more detail on the Data Labeling Engine and rules syntax.

## III.E SECURITY POLICY SPECIFICATION

Once data is loaded into the system and appropriately labeled, the next challenge is figuring out who ought to have which entitlements that grant access to data with a particular label. Sqrrl Enterprise supports this area in a number of different ways.

### Policy Engine

Whereas the Labeling Engine assigns a set of visibility labels to a set of data, Sqrrl Enterprise’s security Policy Engine assigns a set of access entitlements to users on a point-in-time basis, allowing for the configuration of both RBAC and ABAC style policies. This allows for access control flexibility that can evolve alongside changing security and compliance requirements in a dynamic business environment.

### Role-Based Access Control

Sqrrl Enterprise uses [Apache Shiro](http://shiro.apache.org) to manage authentication and authorization. Shiro is capable of plugging into many backends, including LDAP, Active Directory, and Kerberos.

As part of this capability, the user is able to define role mappings, which allows for the translation between a user’s group memberships into labels that ought to be assigned for that user. Role mappings are at the core of [Role-Based Access Control (RBAC)](http://en.wikipedia.org/wiki/Role-based_access_control).
For example, let’s say a user named “Sam” is part of an LDAP group called administrators. The user can then define an RBAC policy that assigns the administrators group access to the “admin” data label, thereby granting Sam access to all data that is labeled “admin”.

This RBAC paradigm has been commonly implemented in technology solutions for decades. RBAC is great for basic levels of protection over collections of data, yet it is static mapping and becomes difficult to configure over time.

**Attribute-Based Access Control**

In addition to RBAC, Sqrrl Enterprise also supports Attribute-Based Access Control (ABAC). NIST defines ABAC as the ability to “grant or deny user requests based on arbitrary attributes of the user and arbitrary attributes of the object [being requested], and environment conditions that may be globally recognized and more relevant to the policies at hand.”

ABAC gives you the ability to key on a user’s attributes, augment them with environmental attributes, and define a rule that says, “Only allow this user access to this piece of data if and only if the user is not a new hire (i.e., has a creationDate in LDAP that is more than 30 days ago), AND the request is during normal business hours, Monday through Friday, 9 to 5.”

Sqrrl leverages Apache Shiro’s attribute translation capabilities and plugin architecture to enable Attribute-Based access rules in Sqrrl Enterprise.

**III.F CRYPTOGRAPHY**

As part of it’s Data-Centric Security guarantees, Sqrrl ships with a number of cryptography capabilities, including encryption at rest, encryption in motion, pluggable encryption algorithms, and pluggable key management. Sqrrl’s enterprise-grade crypto protects data on disk and as it travels across the network. This implementation allows you to narrow the circle of trust for who has data access to only those managing the core Sqrrl system, not those managing other services such as the network or Hadoop.

**Encryption at Rest**

Sqrrl Enterprise ships with transparent server-side encryption for data at rest. Data in Sqrrl and Apache Accumulo is stored in a format called RFile, which is persisted to the Hadoop Distributed File System (HDFS). Each RFile is individually encrypted with a randomly generated key. These file encryption keys are in turn encrypted once more with a key encryption key (KEK). The KEK can then be distributed and protected using best practices for key management.

Sqrrl provides a number of options of handling the KEK, including storing it in HDFS, integrating with external key management solutions such as Gazzang, or implementing a custom connector via our pluggable Key Management interface.
Additionally, Sqrrl and Accumulo use a Write-Ahead Log (WAL) to keep track of recent modifications to data in the system, providing additional atomicity and durability guarantees. Sqrrl Enterprise encrypts the WAL as well, to ensure any data touchpoint is properly protected.

**Encryption in Motion**

It is possible to configure enforcement of data encryption in Sqrrl as it traverses the network, both in server-to-server interprocess communications, as well as client-to-server interactions. We leverage the Secure Sockets Layer (SSL) to encrypt data in motion, and provide utilities that let the user initialize, configure, and manage their SSL certificates.

In Sqrrl Enterprise, starting with version 1.4, it is possible to encrypt the following data paths:

- Sqrrl client to Sqrrl server
- Sqrrl server to Accumulo server
- Accumulo server to Accumulo server

**Pluggable Encryption**

Sqrrl Enterprise uses an industry-standard pluggable framework, the Java Cryptography Extensions (JCE) to build our cryptographic functionality. The cryptographic libraries that plug into this framework can come from a number of providers, including those that have been certified according to the Federal Information Processing Standards Publication 140-2 (FIPS 140-2). These choices include ones from that Open Source Software community, as well as commercial offerings from industry leaders such as IBM and RSA.

**III.G SECURE SEARCH**

More and more Big Data products are introducing real-time search capabilities into their offerings, which is a fantastic development from a customer perspective. At the same time, it is easy to overlook security and privacy when it comes to search.

Consider the following example. You have loaded data into your favorite Big Data database, and this data has a variety of security and compliance requirements surrounding it. You’ve taken the appropriate measures to protect the underlying data, and begin to build applications on top of it. You decide to incorporate search features into your applications, and leverage the database’s indexing capabilities.

This is a fairly common scenario for building search into an application, but what about protecting the index itself? Because it records information about how to match a data record to a search term, a search index can be a source of data leakage – inadvertently revealing information about the presence or the content of certain data fields.
This scenario can be handled in a straightforward way when the data has uniform security and compliance requirements. In other words, the protection of the search index can be implemented similarly to the protection of the underlying data – either grant the user access, or don’t. This makes it practical to wrap a layer of protection around the search index, and synchronize it with the security policy for the dataset at large.

However, more and more, especially in the Big Data space, there are fields of data with different security requirements co-resident in the same logical store. When this is the case, traditional security management practices are insufficient to adequately protect data and at the same time enable users and applications that ought to access it.

Sqrrl Enterprise ships with Secure Search capabilities that respect the underlying high fidelity, granular security labeling rules. Only Sqrrl has term-level security on search indexes, which allows you to individually protect every entry within the index. Sqrrl’s term-level Secure Search supports full-text search, numeric search, and Graph search operations.

III.H AUDIT

Last but not least, to complete the Data-Centric Security picture, you should have the ability to inspect and verify all actions that occur within the system. Sqrrl Enterprise ships with Audit capabilities, which securely log every client action against the system, including information about the request and detail on the each of security operations attempted during execution.

Sqrrl Enterprise gives Compliance Officers the ability to audit every action taken within the system, both successful and unsuccessful operations. This is especially important in a Big Data system, where many users potentially have access to many sources of data. With the level of log detail and the ability to integrate with Enterprise systems, Sqrrl fulfills many data compliance requirements such as full audit, history, and access review.

III.I DATA-CENTRIC SECURITY COMPARISON

Sqrrl has taken great lengths to ensure security, compliance, and privacy of our Big Data solution via best practices in Data-Centric Security. Below is a chart summarizing how Sqrrl stacks up with other NoSQL systems today from a security perspective.
IV. CONCLUSION

Security is a crucial, necessary piece to any business, and soundly achieving it at scale is a difficult challenge. Providing reliable security at the perimeter becomes ineffective when that perimeter is easily circumvented. Further, as silos are broken down and data is gathered in a central, multitenant landing zone, security becomes even more important than before. For these reasons, many have already begun to rethink their approach.

As Big Data technology solutions continue to gain traction in the Enterprise and enter the mainstream, they will be expected to embrace the principles of Data-Centric Security in order to be safely and successfully deployed. Data-Centric Security reduces the threat of breach or exposure by maximally narrowing the trust boundaries of your data infrastructure.

It is important that tools and processes that enforce security do not become an impediment to the business. Only with a Data-Centric Security approach can you build enterprise-class applications that reach only the intended audience. Sqrrl Enterprise provides the most mature, battle-tested, and comprehensive way to protect your Big Data and still maintain the usefulness of your applications and integrity of your analytics.