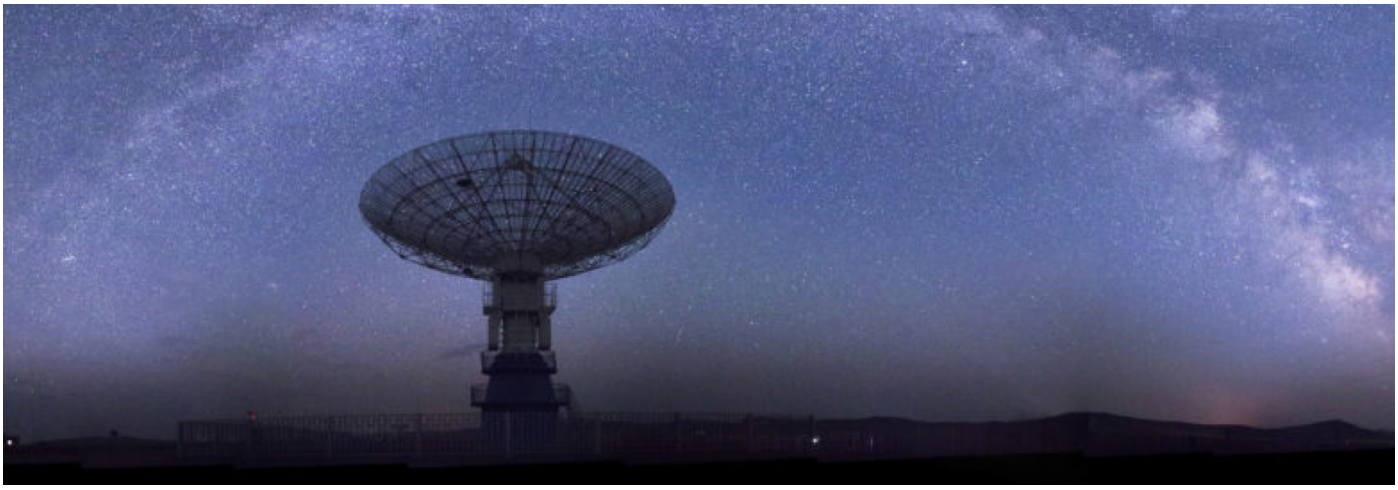
TECHNICAL WHITE PAPER

Defense in Depth with Polaris Radar

TABLE OF CONTENTS

DEFENSE IN DEPTH WITH POLARIS RADAR



There's a strong chance that you, a colleague, or a peer at another company has been hit by a ransomware attack. This means that someone penetrated your perimeter defense, likely through human phishing methods or insecure external access (such as RDP), and has landed malicious code within a permissive zone of your production environment. The outcome of these attacks come in the form of encrypted content (files, folders, operating systems, etc.) that require cryptocurrency payment(s) to make it accessible once more.

This pain can hit hard on several fronts:

- Identifying where the malicious code exists to remove or neuter it.
- Scoping out the damage and timeline of the attack.
- Knowing when the encryption occurred to determine what point in time to restore data.
- Determining how to prevent the intrusion from repeating, if possible.

Fortunately, we at Rubrik understand this pain all too well. One of our earliest customers - Langs Building Supplies - had their production environment hit by a ransomware snag back in 2016. Their team acted quickly and used the immutable nature of Rubrik's backups to recover the encrypted data without paying the ransom.

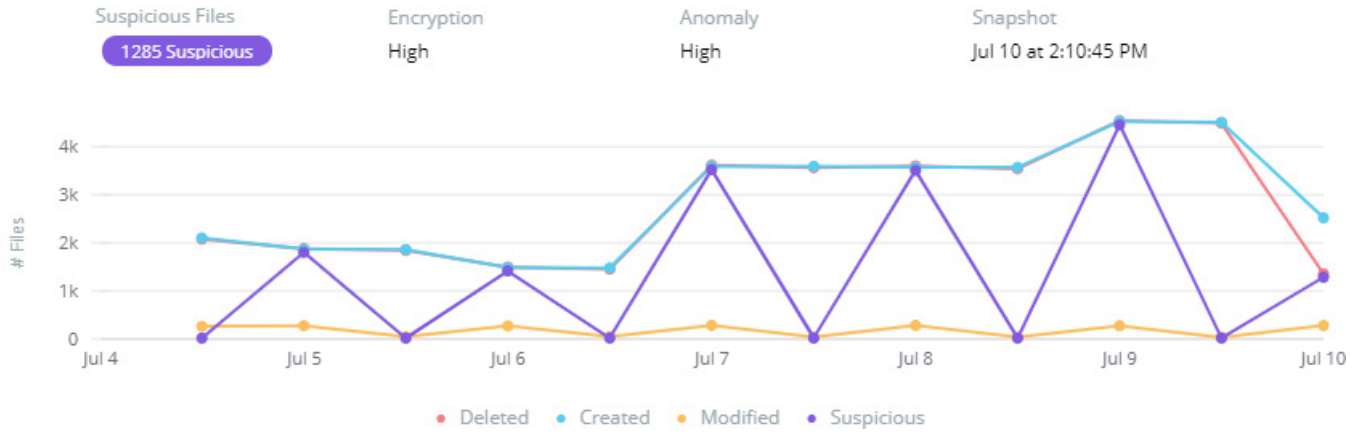
Since then, we've taken the state of the art to a new frontier with the release of Radar, an application that lives on our Polaris SaaS platform, to detect anomalies, analyze the threat, and accelerate recovery with a few clicks. This is the same application that received two Best of VMworld Europe 2018 Awards - Best of Show and Best Data Security and Data Protection Project - recognizing the exceptional impact of Rubrik's Radar ransomware threat protection for ASL Airlines. Their experience thus far has been glowing around the pain points we are working together to remediate:

> *"We experience a minimum of 1 ransomware attack per month. Before Radar, the team spent 15 hours to recover from a minor ransomware attack. If we had been hit with a major attack, I fear recovery could've taken weeks."*

In this paper, I'm going to touch on the more technical pieces of how Radar works, how our immutable filesystem is leveraged for recovery, and then dive into the real-world training and testing we've done to ensure our solution is able to detect file encryption and filesystem metadata encryption that typically accompany a ransomware attack.

HOW POLARIS RADAR WORKS

Consider the fact that every backup "snapshot" you take is a linear data point. Each snapshot contains a wealth of metadata about the source that you've protected - such as a VM, its characteristics, and all of the content that lives within the virtual disk(s). This metadata can be securely collected from an on-premises or cloud environment and sent back to Radar for inspection via a Metadata Sync (MDS) service that can retrieve information from numerous Rubrik clusters.



A series of data points for a workload with suspicious activity.

In the next section I will go deeper into the analysis pipeline and showcase what metadata is being sent to Radar below.

FILESYSTEM BEHAVIOR ANALYSIS PIPELINE

We begin with the local Rubrik cluster that is protecting one or more workloads. Assuming that Radar is associated with the Rubrik cluster, extremely small Metadata Files (MDFs) are generated that contain a payload of metadata "diffs" related to filesystem changes since the last snapshot was taken. These files are then sent up to the Polaris platform where a poller will detect new content to crack open and begin the process of behavior analysis.

The initial task is to perform a preliminary analysis for ransomware by examining the MDFs for changes in the filesystem itself. The information examined is given by metadata detailing the changes that has occured in a filesystem.

Name	Changes	Suspicious	Change In Size	Total Size	Last Modified
Engineering Department	10 Added, 10 Deleted	10 Suspicious	270 B	62.79 MB	10:01:08 AM
Finance Department	10 Added, 10 Deleted	10 Suspicious	284 B	63.03 MB	10:01:06 AM
HR Department	263 Added, 263 Deleted	263 Suspicious	8.09 kB	920.81 MB	10:01:08 AM
IT Department	138 Added, 141 Deleted, 3 Modified	138 Suspicious	54.61 kB	627.59 MB	10:01:08 AM
Legal Department	157 Added, 157 Deleted	157 Suspicious	5.2 kB	516.05 MB	10:01:08 AM
Marketing Department	201 Added, 206 Deleted, 5 Modified	201 Suspicious	247.76 kB	931.66 MB	10:01:08 AM
Public Share	394 Added, 396 Deleted, 2 Modified	394 Suspicious	70.13 kB	1.44 GB	10:01:08 AM
Sales Department	112 Added, 112 Deleted	112 Suspicious	3.29 kB	680.94 MB	10:01:08 AM

File changes are displayed across all items in the filesystem.

This includes, but is not limited to:

- Path
- Size
- ACL details
- UIDs
- GIDs
- Attributes

These MDFs tend to be quite small, ranging from 1-2 kilobytes in size when represented in a binary encoding. The next step is to scan for any anomalous patterns that can be observed in the metadata file when compared to historical data points. For example, a large number of add-file or move-file operations (outside of standard usage norms) might be indicative of a malware infection. The compute requirements for this stage are kept minimal in order to limit compute.

APPLYING MACHINE LEARNING (ML) MODELS

As each snapshot's metadata is collected by Polaris, we leverage a Deep Neural Network (DNN) to build out a full perspective on what is going on with the workload through analysis. The network is trained to identify trends that exist across all samples and classify new data by their similarities without requiring human input. The analysis is largely based off of file system behavior and content analysis.

The detection pipeline has two parts:

1. **File System Analysis** - Performs behavioral analysis on the file system metadata information looking at items like number of files added, number of files deleted, and so forth.
2. **File Content Analysis** - Once outlier behavior is detected, further analysis can be performed on that snapshot.

Overall, this pipeline excels at creating a historical baseline that gets refined over time through machine learning algorithms. This information is used to detect anomalies in behavior for future scans and compares for encryption performed by ransomware. It is worth noting that the method of the analysis is a lightweight process and is designed to efficiently use CPU cycles to lower compute overhead.

If an anomaly alert is generated, Radar is able to go deeper into the content of the files to look for signs of encryption and compute an encryption probability using a statistical model. This allows the analysis pipeline to compute entropy features to measure the level of encryption in the filesystem without the wastefulness of a "brute force" workflow.

USING IMMUTABLE DATA FOR RECOVERY

The key characteristic to Rubrik's approach to handling ransomware and other anomalous behavior is being able to quickly and efficiently restore data to a known-good state. We rely on the immutable nature of our filesystem to execute this restore. In this case, we use the term immutable to mean that it is not possible to change this data collected from protected objects. At a high level, this means that malicious actors are not able to modify snapshot data held within the Rubrik cluster due to the nature of the filesystem design.

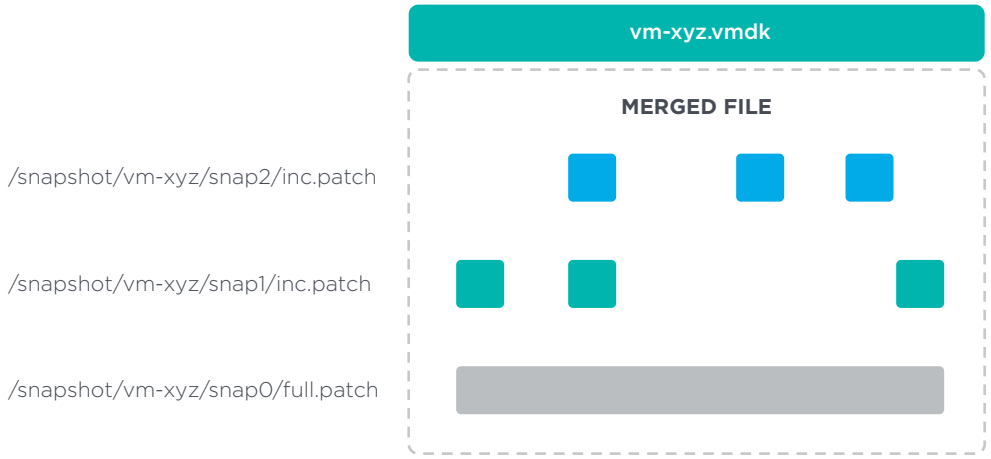
In this section, we will dive deeper into the concepts behind the components that drive our immutable approach to data protection: patch file chains, merge files, stripes, chunks, and checksums.

PATCH FILES, FINGERPRINT FILES, AND MERGED FILES

When Rubrik generates a new snapshot from a protected object, the data is written in a proprietary sparse file format called Patch File. These are divided into blocks called Patch Blocks. As Patch Files are written, a checksum for each Patch Block is computed and written to a Fingerprint File stored alongside the Patch File.

> **Note:** Checksums refer to a compact signature of a larger piece of data which can be used to verify its integrity. This is also known as CRC (cyclic redundancy check).

To restore an object, such as a virtual disk, we use another proprietary mechanism called a Merged File. This uses the concept of a Patch File chain to construct the original object to the requested snapshot.



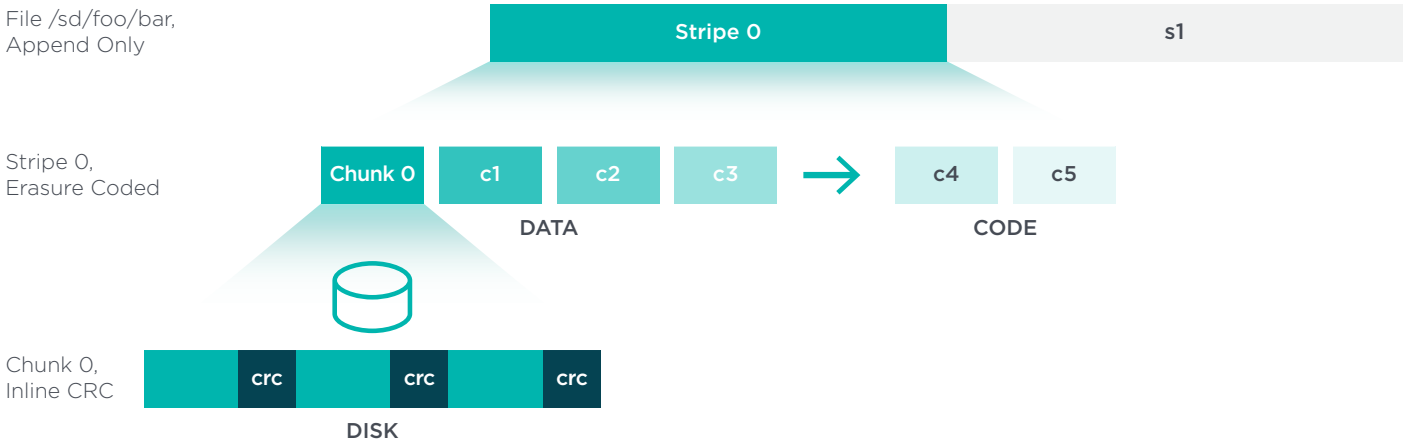
Patch File chain stored under /snapshot is restored via Merged File

Integrity is verified by a background process that is periodically ran to verify the Patch Blocks against their checksums to ensure data integrity at the logical Patch Block level. Patch Files are not exposed to any external systems or user administrator accounts. Simply modifying a Patch File in place (which is not possible) would likely result in making the internal file format to be invalid, allowing the Patch File to catch such a modification on the read path.

Both the Patch and Fingerprint files are guaranteed to be immutable by Atlas, the purpose-built underlying storage system created by Rubrik's engineering team that is covered in the next section.

STORING IMMUTABLE DATA IN ATLAS

Atlas is the distributed file system into which Rubrik writes snapshot data. Patch Files are written and read through an abstraction called the Append Only File (AOF). Both the metadata and physical data are self describing in this regard. AOFs will refuse writes at the API level which are not append only, such as situations where the write offset value does not equal the file size.



A view of the filesystem components in Atlas.

AOFs are logically divided into fixed length segments called Stripes. As Stripes are written, the AOF computes a Stripe level checksum which it stores within each Stripe Metadata.

The following behavior is followed:

1. When a Stripe fills up it is marked CLOSED and cannot be reopened.
2. When an AOF is written in its entirety it is marked FINALIZED and the final, potentially partial, Stripe is CLOSED.
3. A FINALIZED file will contain only CLOSED stripes, none of which can be reopened for write.

Note: A partial stripe can be reopened but only for append operations.

PHYSICAL CHUNKS

Stripes are further divided into physical Chunks stored on physical disks held within the Rubrik cluster. Activities such as replication and erasure coding occur at the Chunk level. Just as with Patch Files, as each Chunk is written a Chunk checksum is computed and stored in the Stripe Metadata alongside the list of chunks. These checksums are periodically recomputed as part of Atlas' background scan by reading the physical Chunks and comparing against the checksums in the Stripe Metadata. Additionally, if a data rebuild is needed, the resiliency provided by erasure coding is automatically leveraged in the background.

In addition to the logical chunk checksum, Chunks themselves are written to disk in a proprietary format with inline checksums. These checksums are used on the read path to validate the chunk content matches what was written. If checksum violations are detected, data will be automatically repaired from other copies in the Rubrik cluster. While the inline Chunk checksums are meant to protect against bit rot and cosmic bit flips, they also serve to protect against malware.

In addition to Rubrik's internal `/snapshot` directory consisting of exclusively Append Only Files, this directory is **never** exposed or accessible outside of the Rubrik cluster using protocols like NFS or SMB. Even if a malicious actor wanted to modify or overwrite a `/snapshot` file, they would not be able to access it.

DATA FINGERPRINTS

Fingerprinting algorithms, in which a large data item is mapped to a shorter bit string, are employed as a more rigorous end-to-end check and created at the data management level.

Fingerprints are leveraged in multiple ways:

1. **Data Ingest** - fingerprints are computed the first time the data enters the system. From this point on, the system ensures that content does not change (i.e. fingerprints are validated) and always does a fingerprint check before committing any data transformations. The fingerprints are compared and validated during ingest to avoid duplicate data being written. This is also known as deduplication. If fingerprint mismatch is detected, the data transformation operation is rolled back.
2. **Data Replication** - fingerprints are compared and validated as part of the replication process between two Rubrik clusters to avoid duplicate data being sent over the network.
3. **Data Archive** - fingerprints are compared and validated as part of the cloud archive process to avoid duplicate data being sent to and stored with a cloud provider.
4. **Data Immutability** - beyond the append-only nature of the Atlas filesystem, fingerprints are used to guarantee data immutability.
5. **Background Scan** - similar to checksums, Rubrik runs periodic fingerprint checks to ensure data integrity.

TESTING KNOWN LIVE RANSOMWARE SAMPLES

While all of the conceptual design behind Radar may sound great, how do we know it works? After all, it's not like you are going to let loose various ransomware variants into your production environment just to see if Radar sends over an alert, right?

The data scientists at Rubrik have your back. In order to ensure that Radar's detection model adequately defends customers environments from cyber threats like ransomware, extensive testing was required.

Note: One approach the team could have taken was to write in-house simulators whose behavior was modeled after known ransomwares. While this would have been a good first step, it is not fully representative of live ransomware behavior. As such, the team opted to rely on a worldwide security community platform to find live samples of ransomware. The richness of these community sourced samples allow us to continuously adapt our supervised machine learning model to account for new and emerging ransomware behavior.

TRAINING THE MODEL

Radar's detection model was trained, validated and tested against a large amount of real world labeled data containing a diverse mix of snapshots from real world usage, simulated usage, and snapshot changes caused by various ransomware and malicious activities.

We followed a standard ML practice of segmenting the labeled data into 3 categories: training, validation and testing. This enabled us to ensure that the model was not overfit to the testing data; training and validation sets are used to tune the model, while testing data is used to evaluate the model on unseen data.

The result was a test accuracy of 99.90%, a false positive rate of 0.073%, and a true positive rate of 99.84%.

PRODUCTION RESOURCES TESTED

We took a sample of 100 resources that consist of:

- Rubrik filesets of file servers
- Linux application servers
- Linux database servers
- Windows application servers
- Windows database servers
- Windows desktops
- Windows file servers

These objects have varying file system sizes ranging from gigabytes to terabytes. This range of applications and sizes served as a sample size representative of our broader customer base.

RANSOMWARE STRAINS TESTED

We started with the top ransomware by frequency of outbreak in order to prioritize the most likely types of infection. Unfortunately, some of these variants required communication with a functioning command & control center in order to initially activate, so the dormant samples were excluded from our testing.

The samples we obtained were:

- Cryptolocker
- Wannacry
- Locky
- Cryptxxx
- Cerber
- Petya

We observed two general types of ransomware behavior across these samples:

1. **File Encryption**: The vast majority of the ransomware exhibited behavior where it targeted specific files with a known set of extensions. Once a file match was found, the file was copied, encrypted, renamed with a new extension, and the original good copy was deleted.

2. **Filesystem Metadata Encryption**: Petya overwrote the bootloader and encrypted the NTFS filesystem metadata table, causing the filesystem to become unavailable.

TESTING FILE ENCRYPTION RANSOMWARE

Radar correctly flagged every ransomware matching the first type of behavior outlined above.

Strain	Samples	Positive Detection	Missed Detection	Accuracy
Cerber	10	10	0	100%
Cryptolocker	9	9	0	100%
Cryptxxx	9	9	0	100%
Locky	7	7	0	100%
Wannacry	10	10	0	100%

Overall, Radar performed excellently against all of the file encryption ransomware, successfully detecting each and every variant it was tested against with 100% accuracy.

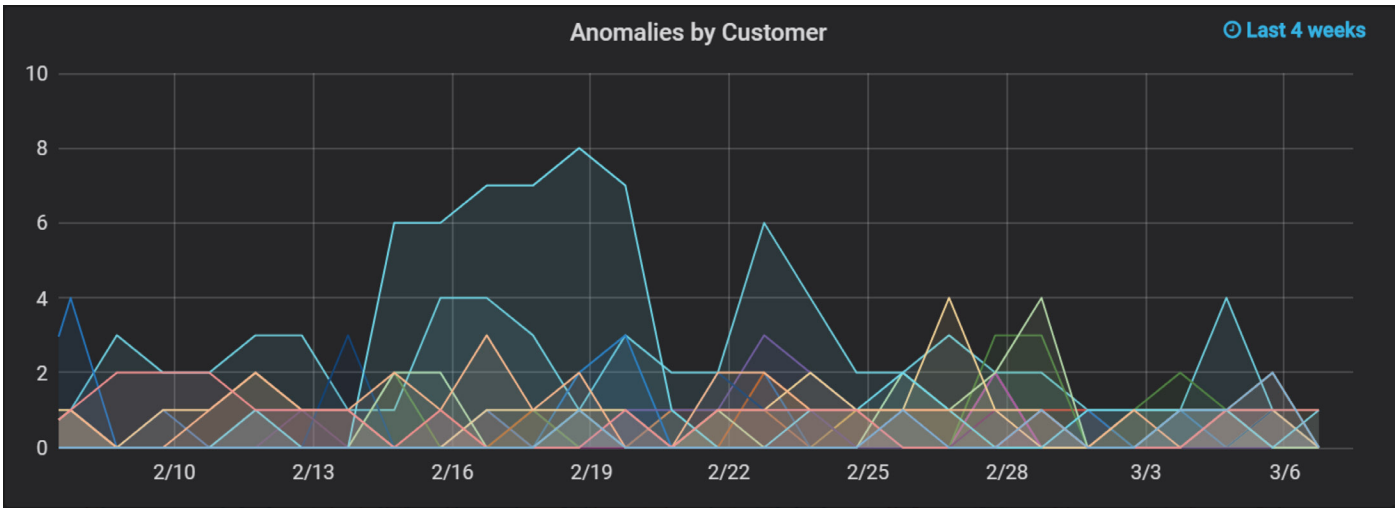
TESTING FILESYSTEM METADATA ENCRYPTION RANSOMWARE

Petya had a unique behavior amongst our samples in that it overwrote the bootloader and encrypted the NTFS filesystem metadata table. This lead to filesystem corruption of the source VM, which prevented indexing of the machine. Rubrik enables full volume recovery of both VMs and physical systems that have been infected with Petya ransomware.

For more on decoupling data from physical servers and providing an image level, or Bare Metal Recovery (BMR), restore process, see this excellent post from Mike Preston.

CONCLUSIONS

Three years ago, we set out on a mission to solve for customer pain related to encryption, insider threats, and ransomware. Our lead security engineer was quoted saying "With an effective backup solution, ransomware can ideally be reduced to a minor inconvenience." Today, we can see that many of our customers are finally able to gather a clear picture into the anomalies that impact their environment on a regular basis.



Anomalies are tracked across customers by the Rubrik data scientists.

As ransomware becomes increasingly sophisticated, successful attacks are more prevalent. To respond quickly, enterprises are adopting a holistic ransomware response strategy. The introduction of Polaris Radar to our SaaS platform has expanded upon that idea to accelerate recovery from ransomware with minimal business disruption and data loss. You know that the investment made into our immutable filesystem that contains your critical backup data is there to be leveraged when time is at a premium without resorting to legacy, filesystem scanning approaches that eat up valuable time.

ABOUT THE AUTHORS

Chris Wahl is the Chief Technologist at Rubrik. He is the author of the award winning Wahl Network blog and host of the Datanauts Podcast. Chris focuses on creating content that revolves around hybrid clouds, CI pipelines, workflow automation, building operational excellence, and evangelizing solutions that benefit the technology community. In addition to co-authoring "Networking for VMware Administrators" for VMware Press, he has published hundreds of articles and continues to enable others on a plethora of open source initiatives, including Rubrik Build. You can reach him on Twitter via @ChrisWahl.

Adam Gee is a Senior Director of Engineering and Founding Engineer at Rubrik. He was a former Staff Software Engineer in Google Search Infrastructure and helped create Colossus, the second generation / replacement for the Google File System. Adam graduated from Stanford University with a Bachelor of Science in Computer Systems Engineering. You can reach him on LinkedIn.

rubrik

Global HQ
1001 Page Mill Rd., Building 2
Palo Alto, CA 94304
United States

1-844-4RUBRIK
inquiries@rubrik.com
www.rubrik.com

Rubrik delivers a single platform to manage and protect data in the cloud, at the edge, and on-premises. Enterprises choose Rubrik's Cloud Data Management software to simplify backup and recovery, accelerate cloud adoption, and enable automation at scale. As organizations of all sizes adopt cloud-first policies, they rely on Rubrik's Polaris SaaS platform to unify data for security, governance, and compliance. For more information, visit www.rubrik.com and follow @rubrikInc on Twitter.

20190717_v3