EPISODE 446

[INTRODUCTION]

[0:00:00.6] JM: The MapReduce paper was published by Google in 2004. MapReduce is an algorithm that describes how to do large-scale data processing on large clusters of commodity hardware. The MapReduce paper marks the beginning of the big data movement. The Hadoop project was an open source implementation of the MapReduce. Doug Cutting and Mike Cafarella wrote software that allowed anybody to use MapReduce as long as they had significant server operations knowledge and a rack of commodity servers.

Hadoop got deployed first at companies with the internal engineering teams that could recognize its importance and implement it, companies like Yahoo and Microsoft. The word quickly spread about the leverage Hadoop could provide. Around this time, every large company was waking up to the fact that it had tons of data and didn't know how to take advantage of it.

Billion-dollar corporations in areas like banking, insurance, manufacturing and agriculture all wanted to take advantage of this amazing new way of looking at their data, but these companies did not have the engineering expertise to deploy Hadoop clusters. Three big companies were formed to help bring Hadoop to large enterprises; Cloudera, Hortonworks and MapR. Each of these companies worked with hundreds of large enterprise clients to build out their Haddop clusters and help them access their data.

Tomer Shiran the spent five years at MapR seeing the data problems of these large enterprises and observing how much value could be created by solving these data problems. In 2015, 11 years had passed since MapReduce was first published and companies were still having data problems. Tomer started working on Dremio, a company that was in stealth for another two years. I interviewed Tomer two years ago, and when he still could not say much about what Dremio is doing, we talked about Apache Drill, which was an open-source project related to what Dremio eventually built. I should say is an open source project.

Earlier this year, two of Tomer’s colleagues; is Jacques Nadeau and Julien Le Dem came on to discuss columnar data storage and interoperability, and what I took away from that conversation was that, today, data within an average enterprise is accessible, but the different formats are a problem. Some data is in MySQL, some is in Amazon S3, some is in Elasticsearch, some is in HDFS, stored in parquet files, and also different teams will set up different BI tools and different charts that read from a specific silo of data.

At the lowest level, the different data formats are incompatible. You have to transform MySQL data in order to merge it with S3 data. On top of that, engineers doing data science work are using sparc and pandas and other tools that pull lots of data in the memory, and if the end memory data formats are not compatible, then the data teams can't get the most out of their work. They can't share their datasets with each other.

On top of that, at the highest level, these data analysts that are working with the different data analysis tools creates even more siloing. Now, I understand why Dremio took two years to bring to market. Dremio is trying to solve data interoperability by making it easy to transform datasets between different formats. They're trying to solve data access speed by creating a sophisticated caching system and they're trying to improve the effectiveness of the data analysts by providing the right abstractions for someone who is not a software engineer to study the different datasets across an organization.

Dremio is an exciting project because it's rare to see a pure software company put so many years into upfront stealth product development. After talking to Tomer in this conversation, I'm looking forward to seeing Dremio come to market. It was fascinating to hear him talk about how data engineering has evolved to today, and some of the best episodes of Software Engineering Daily covered the history of data engineering, including an interview that we did with Mike Cafarella, who was the cofounder for Hadoop.

We also did another episode called the history of Hadoop, in which we explored how Hadoop made it from a Google research paper into a multibillion-dollar multi-company industry, and you can find all these old episodes if you download the Software Engineering Daily app for iOS and for android. With the apps, we're building a new way to consume content about software engineering, and they're also open sourced at github.com/software engineering daily. If you're looking to get involved in our community and contribute to the open source projects, we would love to get your help.

With that, let’s get on to this episode.

[SPONSOR MESSAGE]

[0:04:51.1] JM: You are programming a new service for your users, or you are hacking on a side project. Whatever you’re building, you need to send email, and for sending email developers use SendGrid. SendGrid is the API for email trusted by developers.

Send transactional emails through the SendGrid API. Build marketing campaigns with a beautiful interface for crafting the perfect email. SendGrid is trusted by Uber, Airbnb and Spotify, but anyone can start for free and send 40,000 emails in their first month. After the first month, you can send 100 emails per day for free. Just go to sendgrid.com/sedaily to get started.

Your email is important. Make sure it gets delivered properly with SendGrid; a leading email platform. Get started with 40,000 emails your first month at sendgrid.com/sedaily. That’s sendgrid.com/sedaily.

[INTERVIEW]

[0:06:03.9] JM: Tomer Shiran is the CEO of Dremio. Tomer, welcome back to Software Engineering Daily.

[0:06:09.1] TS: Thanks for having me.

[0:06:10.4] JM: The last time we spoke, we were talking about Apache Drill, and in another episode I talked to your colleague, the CTO of Dremio, who is Jacques Nadeau, and we talked columnar data in that episode. In both of these episodes I knew that the two of you were working on this stealth company; Dremio. I didn't know much about what you were building. Now that the product is out, I want to take a top-down approach, and we’ll discuss what Dremio is and then we’ll discuss the technical topics that we discussed in the past two episodes and sort of how they relate to the construction of this product.

To start off, its 2017, and we've got teams of data scientists, data engineers, data analysts, these data teams that are also working with software engineers. They've got tons of data, and they have some problems managing and accessing and visualizing that data. What are some of the specific problems faced by these teams of data engineers and data scientists and software engineers?

[0:07:21.0] TS: Yeah. If you think about our personal lives and how easy it is when we go home and we have a question and we go online and ask us that question on Google and one or two seconds later we have an answer. We have this amazing experience with data in our personal lives, and that extends to smartphones, and we want to book travel, and within two minutes we booked a travel and it's very simple, but then we come to work and it often takes us months to be able to answering a new question or create a new visualization.

Especially when you get to the enterprise where data is distributed all over the place and kind of owned by different teams and a lot of work has to happen in order to make that data available for somebody to be able to ask questions on that data. That's kind of the core problem, and a lot of times you'll see companies go through lots of kind of ETL work where they're extracting and transforming data and they have to figure out some kind of data warehouse where they can load that data into and make it available. It's just a lot of work and that takes months to do. So that's a bit challenge.

[0:08:32.5] JM: Some of these data is sitting in Amazon S3, some of it sitting in Elasticsearch, some of it is sitting in Mongo. Is the data in all of these different places, is it in the right format to be queried by these data teams?

[0:08:48.3] TS: It may or may not be. I think the word has moved to a place where we have lots of different types of data source and each of these data stores is really optimized for building different types of applications, and so developers that build an app on — Web app may choose Mongo, because that's easier to build the app there, or for some other use case, it's Elasticsearch and maybe put the log files on S3.

Really, they're kind of optimizing for what's the best place for me to put the data for the application that I'm trying to build as supposed to for the type of analysis that somebody may later want to do with that data. That's a challenge. If you think of the old world, maybe it was possible to have all my data in one relational database, and I could just as easily query that data and do my analysis directly on that, let's say, Oracle database, but that's obviously no longer the world. With today's kind of volume and variety and complexity of data, it's just way beyond a place where we can just have all our data somehow magically in one relational database and expose that to a bunch of BI tools. That's just not feasible anymore.

[0:09:58.2] JM: Do we want to uniformly turn these datasets into a single access system with consistent latency, consistent formatting? One thing we could talk about is columnar data. I think we will talk about that. Is the goal of Dremio to uniformly turn these datasets into columnar data?

[0:10:23.7] TS: I would describe the goal really as a self-service data. Our goal at Dremio, really, is if you think of this new world where he data no longer can realistically be in one place and one relational database and at the same time you have this growing demand for kind of self-service access to the data everybody from the data scientists to the product manager and the business analyst and so forth, how do we create a way for these people to be self-sufficient, to be empowered to do whatever they want with the data no matter where that data is, how big it is, what structure it's in.

To do that, we have to solve a variety of different problems that the traditional data infrastructure just doesn't deal with, right? If you think about the historically, we've had data in different places. We would then have to ETL that data into maybe some staging area, like a data lake or a Hadoop cluster or something like S3, and then querying directly on that kind of system is more often than not too slow, so companies will tend to ETL a subset of the data maybe the last 30 days or last the — Or some aggregate level data into a data warehouse. That's not fast enough, so they create cubes and they pre-aggregate into other tables in the data warehouse and maybe they extract into the BI servers, and at the end of all that you have 10 different copies of the data and really a lot of manual work that has to be done by engineers every time somebody has a question or want to do something new.

We think that in order to achieve this world where companies really want to leverage data, they want to be data-driven, you have to create a system that empowers the end-users, the data consumer. Whether they're a data scientist who's using pandas or a business analyst using Tableau, how do you empower that user to do everything on their own and get the performance that they need, which is often sub-second response time even when the datasets are even petabytes in size.

[0:12:27.7] JM: All right. I think we understand this from the high level product perspective. What are the features that you need to build in order to make that data access easier? Are we talking about a visualization product? Are we talking about query language? Are we talking about some sort of dashboard with both of those things built into it or are we talking about an API? What are the features that you need?

[0:12:53.4] TS: What Dremio provides — By the way, Dremio is available as an open source project as well as kind of an enterprise edition, and so you can download it. There are basically two aspects to it. On one hand, if you think about most companies, there are different users that want to use different tools to explore and analyze the data, ranging from BI, to Excel, to more advanced things like R and Python.

We don't want create a visualization tool or something that people use to analyze a data. They already have plenty of those tools, but when do you want to provide these data consumers the ability to access and analyze any data at any time, and so we would provide a number of capabilities in that regard and that includes kind of an integrated data catalog where they can find the data that they want and kind of use a search type interface for that and we provide them with a visual interface that they can curate the data and create new virtual datasets and collaborate with their colleagues.

At the end of the day we want to enable their existing tools, whether that's Tableau or PowerBI or Qlik Sense or R or Python to be able to connect to the system and run the query and get a response in less than a second no matter how big the data is or where it's coming from.

When it comes to — For the data consumer, we want them to live in a logical world where they feel that they can do anything with the data at any time. Now, at the same time we have to provide the execution and acceleration capabilities that will actually make that fast, and so that's where underneath the hood there's an entire kind of SQL distributed execution engine leveraging Apache Arrow. There's an acceleration where we've pioneered something called data reflections, which can accelerate quarries by orders of magnitude, and then there's kind of this data virtualization there that knows how to talk to different databases and push down quarries or parts of queries into these underlying databases, whether they're a NoSQL databases, like Elasticsearch and MongoDB, or it's relational databases like Oracle and SQL Server and MySQL.

[0:14:55.6] JM: You talked about a few of the technical concepts there, the reflection and the virtual dataset concept. What's the right order that we should approach these concepts to dive into them?

[0:15:08.8] TS: Right. If you think about what the — Let's say the business analyst who's using the system or the data scientists who's a uses and they want to work with the data. They're never aware of reflections, so I think we should first focus on kind of from their experience, they are dealing with datasets. You have the physical datasets. Those are the things that are in the collections in MongoDB and the index is Elastic and tables in Oracle and the Hive tables in Hadoop. Those are physical datasets, and then we allow this users to create their own virtual datasets. Basically use of the data and they can share them with their colleagues and build on top of each other and so forth. Users always think about the world in terms of datasets and both physical and virtual datasets.

[SPONSOR MESSAGE]

[0:15:59.1] JM: DigitalOcean Spaces gives you simple object storage with a beautiful user interface. You need an easy way to host objects like images and videos. Your users need to upload objects like PDFs and music files. DigitalOcean built spaces, because every application uses objects storage. Spaces simplifies object storage with automatic scalability, reliability and low cost. But the user interface takes it over the top.

I've built a lot of web applications and I always use some kind of object storage. The other object storage dashboards that I've used are confusing, they're painful, and they feel like they were built 10 years ago. DigitalOcean Spaces is modern object storage with a modern UI that you will love to use. It’s like the UI for Dropbox, but with the pricing of a raw object storage. I almost want to use it like a consumer product.

To try DigitalOcean Spaces, go to do.co/sedaily and get two months of spaces plus a $10 credit to use on any other DigitalOcean products. You get this credit, even if you have been with DigitalOcean for a while. You could spend it on spaces or you could spend it on anything else in DigitalOcean. It’s a nice added bonus just for trying out spaces.

The pricing is simple; $5 per month, which includes 250 gigabytes of storage and 1 terabyte of outbound bandwidth. There are no cost per request and additional storage is priced at the lowest rate available. Just a cent per gigabyte transferred and 2 cents per gigabyte stored. There won’t be any surprises on your bill.

DigitalOcean simplifies the Cloud. They look for every opportunity to remove friction from a developer’s experience. I’m already using DigitalOcean Spaces to host music and video files for a product that I’m building, and I love it. I think you will too. Check it out at do.co/sedaily and get that free $10 credit in addition to two months of spaces for free. That’s do.co/sedaily.

[INTERVIEW CONTINUED]

[0:18:17.6] JM: The virtual datasets are these in-memory representations of the datasets, the physical datasets that are probably represented on disk.

[0:18:27.5] TS: Yeah, the physical datasets are represented typically on disk in some source system. The virtual dataset really is — It’s not an in-memory representations. It’s just a logical definition, and the beauty of this is that you can then have a thousand users creating these virtual data. There’s virtually no cost to these things, and they can create as many as they want, and at the end of the day that's important, because kind of in the old world, what happens is that every user wants to get the data into their own exact shape and form that they like before they do their analysis, and that indeed involves downloading the data into a CSV file or a spreadsheet or kind of creating a copy of data. Whereas in Dremio, that's not required. Every user can create the data, kind of massage the data, get it into some other form and save that as a virtual dataset with literally zero overhead in the system. We're not materializing those virtual datasets.

[0:19:27.4] JM: I see. They're essentially saving their queries, and the query, when they decide to run it, becomes a materialized view. Until then, it's just a query, which in a sense is a virtual data.

[0:19:41.9] TS: Right. These virtual datasets are essentially defined by a select statement in SQL. Of course, you can define virtual datasets that are built on top of other virtual datasets. As an example, you may have a virtual dataset that is a join between a Hive table and an Elastic index, and then another virtual dataset that maybe selects only the records from the first virtual dataset and filters them on the city equals a mountain view. You can have that kind of basically data graph evolve over time of these virtual datasets defined based on other virtual datasets.

What's nice is that these virtual datasets are exposed when a BI tool, such as Tableau, for example, connects to Dremio. All of these datasets, whether they're physical or virtual are exposed to Tableau as tables that the Tableau user can then play with. They can start analyzing and they can start visualizing, creating charts and dashboards and stories and so forth.

[0:20:40.0] JM: It sounds like these virtual datasets are kind of like stored procedures, but when we change the way that we're referring to that abstraction, then you can start to build a different product around that. You start to build a product with the idea that this is a virtual dataset. It's not a query. I guess it becomes easier for people to think about merging virtual datasets together rather than running queries on top of one another, or maybe you could help me understand the difference in terminology, because it sounds like a virtual dataset is kind of just like a stored procedure.

[0:21:18.7] TS: It's actually similar to a view in a database. If we were talking about one relational database, then there are views in that system. Now, views are typically defined by like a DBA or somebody who's pretty technical. In the case of Dremio, these virtual datasets can be defined either through a SQL statement, a select statement, or by interacting visually with the data, so as a user interfaces that's similar to Excel and allows them to click on a column and say, "I want to drop this column," and select the zip code in that address column and, say, click on extract, and we figure out how to extract that into a new column. All that's doing underneath the hood is kind of modifying that SQL definition of that dataset. Effectively, you're creating views of the data. These are these virtual datasets.

[0:22:05.9] JM: What's a Dremio reflection?

[0:22:08.3] TS: We've talked about kind of the notion of virtual datasets, and that's what users see in the product. That's what they interact with. That's what they share with their colleagues. That's what they analyze when they connect a BI tool or something like R or Python to Dremio. They [inaudible 0:22:21.6] these virtual datasets.

At the end of the day, these uses expects high-performance. If something is really easy, but it's slow, people don't want to use it. When we looked at, okay, achieving the Holy Grail of analytics, a system basically that allows you to interact and analyze any data at any time. Part of the problem was solving kind of the logical aspect to make it easy for people to find things and collaborate, create new datasets and so forth.

The other aspect is how do you do that fast. There are a lot of challenges in that regard, because often times the data is in a system that just physically won't let you create fast. If the data is in in an Elasticsearch index and you're trying to join two indexes and that requires scans of these indexes. Well, if that system can only do 20,000 records per second per core, it's only so much speed that you can have when you're doing that kind of analysis.

Really, what users wants is they want a response time of up to a few seconds in most use cases regardless of how big the data is, and that's where the reflections come into play. That's basically our kind of unique IP that we've developed to allows us to provide interactive speed queries regardless of the size of the data and the location where the data lives. That's a lot of the magic of the system from a performance standpoint.

[0:23:54.9] JM: I want to ask a little bit about the propriety stuff. I know you probably can't tell me exactly what it's doing. My sense is that it's a complex caching system that does maybe some eager loading in certain situations, something like that.

[0:24:11.2] TS: Yeah, I think that's fair. It's a complex caching system. Now, what we're caching is not just the — A traditional cache is caching kind of copies of the data. In our case, the reasons these are called reflections is because we are caching different reflections or different perspectives of the data. We may cache the data in different shapes and forms, for example. We may cache a given dataset pre-aggregated by various dimensions. We may cache it sorted in specific ways or partitioned in specific ways. These are what we call these data reflections.

One of the hard parts here is when a query comes in, let's say from a BI tool like Tableau or Qlik or Power BI, we then — Our cost-base optimizer has to look at that query, compile it and say, "Okay, how can I reduce the cost of executing this query by leveraging one or more reflections?" Internally, we are rewriting that query plan if it's possible to leverage the reflections rather than scanning the raw data in the source system again and again and again and again. That's kind of how we get the performance.

You can think of it as when you go on Google and you ran a search query, it would be very slow if Google had to then go and scan all the world's webpages. That would be slow. Instead they've created an index and they've created various models where they have already organized the data internally in various shapes that allow them to retrieve answers to specific queries very, very fast, and that's kind of what we do at Dremio. Of course, with different types of data structures that are more suitable for analytical queries.

You could think about in the old world you had things like cubes and projections and indexes and relational databases and all those types of techniques. If you combine those into one system and you put it behind the scenes where the end user doesn't even need to know about it, but it's the system deciding which one of these representations or maybe multiple of them to use at query time. That's kind of how we get that performance.

[0:26:24.3] JM: Let's talk about some of the intelligence that goes into that reflection building. I don't know the way to approach this, but maybe I'm a data analyst, I've got a MySQL database, an Elasticsearch cluster, I've got three or four other data sources, I build some virtual datasets, and if I was naively requesting those datasets at different points throughout the day, it would take forever to build those virtual datasets, because they're just large sets of data and it takes a while to query them and pull them from disk into memory. With the Dremio, if I've got this

Dremio reflection, it's going to intelligently have some of that information cached. It's going to make it accessible to me in my BI tool a little more aggressively. What are you doing that's intelligent that gets that data into something that it gets things going faster? Just give me some guidance for what you're doing.

[0:27:27.6] TS: Yeah. Let's start with where do these reflections live. The reflections live — They live in a persistent store, and we have three options. We can leverage S3 to store these reflections. We can leverage HDFS or Hadoop cluster, any kind of Hadoop cluster for these reflections, and then we can also just leverage the local disks of a Dremio cluster and stripe them across that, and that's good for cases where you don't have Hadoop and you don't have — And you're not running in the cloud. So you can do that as well.

Then that's kind of where the reflections life, and we are maintain them on kind of let's say an hourly basis or whatever your SLA is, so you can kind of define that per data source as well as per data set level and say, "I'm willing for this data to be at most one more still," and then our engine makes sure that we're maintaining that reflection on that schedule and updating it either by doing full updates or incremental updates to maintain these reflections. Then when your query comes in from the BI tool, that query says, "You know what? I want to see all the — Just count the number of events based on city." Per aggregating by city, and one of our reflections in the cache maybe the data already aggregated by city state and neighborhood. We can kind of rule that up and give you the answer that you asked for by going through maybe a million records instead of going through a trillion records, and that of course gives you that many orders of magnitude, faster response time.

[0:29:00.3] JM: You have this scheduled job that pulls data from the virtual dataset into the Dremio reflection, and the Dremio reflection will give you a faster — The Dremio reflection is the kind of — It's like the materialized view of a virtual dataset, right?

[0:29:20.5] TS: That's correct, and a high level. One of the important nuances here is that the reflection doesn't have to — A single reflection could actually accelerate queries on many different virtual datasets, because as you may recall, these virtual datasets could be very much related to each other and they could be derived from one another. At the end of the day, when a SQL query into the system, our optimizer doesn't really care about the virtual dataset. It kind of expands all those definitions and looks at kind of the foundational relational algebra and says, “Okay. How can I kind of massage this query plan? Cononicalize it? Figure out whether there is or there isn’t a reflection or multiple reflections that I can use to satisfy this query more efficiently.” It’s not necessarily a one-to-one between the virtual dataset and a reflection. There could be many reflections associated with a virtual dataset, but even those reflections could also accelerate queries on hundreds of other virtual datasets.

[0:30:22.6] JM: You said there’s a scheduled time or some sort of SLA that where the reflections gets updated with the most recent pieces of data that have been added to the datasets that the virtual datasets are referring to. Does that mean that if I load my reflection into my BI tool, it may not represent the most up-to-date version of the data?

[0:30:50.5] TS: Yeah, that’s correct. Like with any caching system, you’re basically saying, “I’m okay with looking at data that is one hour old.” That’s kind of the tradeoff that you’re making here. Now, for most companies today, it takes them months to create a new visualization. For them, it’s a no brainer, right? We do have some users where that SLA that they’ve defined in our system is one minute. We have, for example, a use case that involves IoT data and it’s a predictive maintenance use case where it really matters that it’s very much up-to-date, and so they do these refreshes on a minute by minute basis. For the most part, people seem to kind of settle on maybe one hour kind of timeframe or something around that range is what they prefer.

[0:31:41.3] JM: It seems like maybe you could also time the reflections to e up-to-date when the data scientists sits down to do their daily or weekly analysis. I guess I don’t know much about how data analysts work, so maybe they don’t work that way or maybe they do more ad hoc inspirational work.

[0:31:58.9] TS: Yeah, you could certainly. If somebody is that — They come in every morning at 9 a.m. and that’s when they do their work for an hour, you could certainly set it up so that you can kind of trigger it or you can use an API call to trigger a refresh at the time that you want.

There are also all sorts of kind of sophisticated capabilities here around how these reflections are maintained. Sometimes you may have a situation where you have different reflections that can be built off of each other rather than each of them being built from the source data. So then we internally maintain this kind of — You can think of it as a graph, dependency graph of reflections and will automatically figure out what's the right order in which we should refresh these reflections so that we're kind of minimizing the amount of load that we put on the — In the case of an operational database, on that operational database.

[0:32:51.1] JM: Tell me a little bit about building a reflection as far as you can go without revealing the secret sauce.

[0:32:57.4] TS: Sure. The reflections are actually stored in a — We use the Apache Parquet format, combination of that and some elements from Apache Arrow, and then added some optimizations on that. That's basically the format on disc of each of these reflections. We actually allow the user, the administrator of the system or somebody who has a good knowledge of the data to go and tune these reflections. They can say, "You know what? This set of queries — I just got a phone call from this business analyst in the marketing department saying that he's running some queries and they're too slow."

As an administrator of the system I could say, "You know what? Let me add a new reflection that's optimized for that type of workload." Once I've defined that and that may take me a minute or maybe two minutes in the system, just a few clicks. That marketing user will now have very, very fast response time and they won't have to change any of their kind of client application or if they're using something like Tableau, they won't have to change the worksheet or their dashboard. It's entirely transparent to the end user. That allows the administrator of the system to be fine tuning these reflections. That's important, because while we have some kind of automated capabilities, we also have kind of a voting system where users can vote for things that they want to be faster. At the end of the day, there are things that we don't know.

One of our customers, for example, the CFO has their own set of reports that they look at every day, and it's just one person. It's not something that's very frequent in the system. There's no way we could have known that that was important. But because they are the CFO, they are naturally important, so we want to provide that kind of flexibility to users to be able to control which reflections exist in the system and they can add and remove them as they want. It's actually very easy.

[0:34:54.2] JM: If I recall, the Apache Arrow project is for in-memory datasets interoperability. It's like you should be able to share your data that is in Python, in-memory in a Python. You're doing something with Python, like pandas data science stuff and it's sitting in memory, you should be able to shift that data to Sparc and do stuff in-memory with Sparc, or just have that data sitting in-memory and you could run Sparc operations on it. You can also run Python operations on it, and Arrow is the format that allows for that interoperability. Is that right? Am I accurately describing Arrow?

[0:35:36.6] TS: Yeah. That's exactly right. Arrow is all about having a columnar in-memory kind of technology for representing data and memory and processing it and leveraging kind of the modern CPUs and GPUs, and it's also a standard, a way for different systems to share the same type of memory structure.

[SPONSOR MESSAGE]

[0:36:04.4] JM: At Software Engineering Daily we need to keep our metrics reliable. If a botnet started listening to all of our episodes and we had nothing to stop it, our statistics would be corrupted. We would have no way to know whether a listen came from a bot or from a real user. That's why we use Encapsula to stop attackers and improve performance.

When a listener makes a request to play an episode of Software Engineering Daily, Encapsula checks that request before it reaches our servers and filters bot traffic preventing it from ever reaching us. Botnets and DDoS attacks are not just a threat to podcasts. They can impact your application too.

Encapsula can protect your API servers and your microservices from responding to unwanted requests. To try Encapsula for yourself, go to encapsula.com/sedaily and get a month of Encapsula for free.

Encapsula's API gives you control over the security and performance of your application. Whether you have a complex microservices architecture, or a WordPress site, like Software Engineering Daily. Encapsula has a global network of over 30 datacenters that optimize routing and cacher content, the same network of datacenters that is filtering your content for attackers is operating as a CDN and speeding up your application.

To try Encapsula today, go to encapsula.com/sedaily and check it out. Thanks again, Encapsula.

[INTERVIEW CONTINUED]

[0:37:53.6] JM: If you've got like data in a MySQL database and an Elasticsearch cluster and three other data sources and you want to get that from a physical dataset that has been earmarked as a virtual dataset that people like and you want to pull it into a reflection, you maybe want to get all that data into the Arrow format so that it's all interoperable, and then you'd get the Arrow format put into Parquet so it's an on-disk consistent format and then you put the on-disk format into a reflection. Is that right?

[0:38:30.5] TS: Yeah. Everything in Dremio, as soon as it leaves the disk or the source system, it becomes Arrow format. All of our internal execution is based on Arrow, and that means that as soon as we read records, a batch of records from Elasticsearch or from HDFS, immediately that becomes a batch of Arrow in memory.

As it's executing in our entire execution engine, it remains, it goes from one Arrow buffer to another Arrow buffer. It's going through kind of the different operators. Then you can also take the results of an execution and you could — The goal really is to be able to use something like Python, and panda supports Arrow as its high-performance memory representation. Now, imagine being able to take the result of a join, whether it's just a join of two tables in Hadoop or as three or join across different systems and be able to do that analysis in Pandas without having to de-serialize and serialize data.

That's really the goal with Arrow and it's something we at Dremio put out, open sourced about, I want to say, a year ago, and have since become kind of the standard memory representation for Pandas and videos GPU data frame and some of the GPU databases now use it as their in-memory representation as well. It's quickly becoming that what we had hoped, which was to create this industry way to represent data in-memory for analytical use cases.

[0:40:02.1] JM: When those reflections get pulled out into the BI tool, are the reflections getting pulled out into Arrow and then being read by BI tool?

[0:40:15.5] TS: One of the things we've done is we've developed a very high-performance kind of translator from Parquet to Arrow. The reflections are stored in Parquet because there are efficiencies in Parquet that we can leverage especially with how we use Parquet to store the data in a very highly compressed way, but once we want to read that data, immediately we read it into kind of the Arrow format.

As more and more of these client applications, such as Python and R, but in the future also various kind of commercial BI tools, embrace Arrow as their way of kind of ingesting and leveraging data and some of them are actually working on that right now. You'll have an extremely high-performance way to move data into those systems as well without having to go through kind of the traditional ODBC protocol.

[0:41:09.9] JM: I see. Today, maybe things are not as fast as they will eventually be, because — Just to refresh people, you've got — The end-to-end explanation; you've got a MySQL database, an Elasticsearch cluster, three other data sources. Everything's got different latency. All the data has different structures. You go into Dremio, you label some of those physical datasets as virtual datasets that like, "Oh, I'm going to want this join between my MySQL database and the data in Amazon S3 or RDS or something," and you get that join labeled as a virtual database on some scheduled basis that virtual dataset gets translated into a Dremio reflection, which is a materialized view that sits on disk so that you can access, basically cache on disk so that you can access it faster so you don't have to do the entire join on the fly. You've just basically got it cached on disk. Then it's cached on disk in Parquet format, which is a columnar format, and people who want to go back and listen to the episode with Jacque to learn more about columnar data, they can do that. We went really into detail on the Parquet and Arrow stuff.

It's sitting on disk and Parquet. When you want to access it from a BI tool today, you need to — You might need to figure out how to translate — Or maybe there's some plug-in. I think you said ODBC. I don't know much about what that is. Open database, something? In order to pull that from the Parquet file into your BI tool, there is some degree of latency or something, because there's not an easy arrow translation. Maybe you could disambiguate that process what you're referring to.

[0:43:03.4] TS: Yeah, today if you're using a client application on your desktop, whether that's, let's say, Excel, or if you're using something like Tablea, those tools typically use an API called — Kind of a local API called ODBC. There's another one called JDBC for Java applications. That's how they want to talk to databases. So then you have kind of an ODBC driver for each database on that client machine, which means that in the case of Dremio, we can maintain everything in Arrow all the way to the client over the network, but then it does have to get translated from Arrow into kind of that ODBC API so that these BI — These kind of traditional BI tools can use it. That doesn't have to happen for things like data scientists who are using, let's say, pandas, because pandas now support Arrow as a native memory representation. Actually, it can actually do operations on that.

Wes McKinney, who's the creator of pandas, is actually one of the primary contributors on the Arrow project. For some tools, like Python, for example, you don't need to go through that translation. While for other tools that are, let's say, Window applications, like Tableau, you need to go through that translation.

Overtime we expect that more and more of these client applications will simply embrace Arrow natively and they'll not have to go through that translation layer that they all have to go through today. That's an ongoing thing. I think depending on the use case, it may or may not matter so much. Often times, for BI tools, because the amounts of data that are being shown in the tool usually aren't very big. At the end of the day it's painting dots that a human eye in the case of something like Tableau, the human eye needs to be able to see all these dots on that report. It doesn't really help that there are a billion dots. You can't really visualize that.

Most of the time, these datasets are much, much smaller because what's happening on the backend is that Dremio is already getting the instructions from Tableau to aggregate the data by city and state and customer, and so we're already doing that aggregation and just sending back a much smaller amount of data.

[0:45:20.9] JM: What's interesting about this project to me is you and I first talked a couple of years ago when we're talking about Drill. We did this show about Drill, and Dremio was just s splash page with very little information, and now you've got a full-fledged project and it makes total sense to me. It's unlike anything I know of. Maybe there's something internally at Google where they use something like this, but it sounds like a pretty cool and differentiated project where you've got a mote in the sense that you've got this caching system that you've developed, and I'm sure that will get more sophisticated overtime.

I'd love to know, did you know that this was the product two years ago, or was there some finagling with the strategy?

[0:46:13.9] TS: We kind of knew at a high level what we wanted to achieve. I had come from kind of — I was one of the early employees at MapR. I was VP of product there. I spent 5-1/2 years, and one of the things I observed over those years kind of the emergence of big data was that it was just way too hard. At the end of the day people wanted to leverage data, but they then had to go hire 20 data engineers and train everybody. It was very hard to get value out of the data especially to enable non-technical users as well to get value out of the data.

When we looked at that and we said, "Why is it so hard for companies to leverage their data?" That was kind of the initial question we were trying to answer. As we started interviewing more and more of these companies, and especially kind of the larger kind of global 2,000 that many of the brands you're familiar with, it became clear that it was just so many different kind of point solution and legacy technology that they had to deal with, and that if somebody could really develop a system that would abstract all of these away and provide performance without making the users have to think about it, that could really be a game changer in analytics, and that could finally enable companies to start capitalizing on the data that they actually have in various places.

That was the goal. Technology is different, definitely evolved overtime. We had to build an entire engine from the ground up for this purpose. We couldn't use any existing SQL engine or Drill or any of those things that none of those kind of met the needs for the performance and the types of things we're doing with reflection. We build that and we ended up building Arrow as part of that and actually open sourcing that along the way.

When we launched the company we also said, “You know what? All of our executive team actually comes from companies like MapR and MongoDB and Hortonworks and has a ton of experience with open source and we think that’s the right strategy in 2017.” Dreamio is itself an open source project now that many companies are downloading every day and putting to use.

[0:48:39.8] JM: The Dremio open source project itself, meaning thing that pulls from Parquet files basically. What is the Dremio open source project?

[0:48:50.3] TS: Yeah, actually everything we talked about today. That’s all open source. You can go to dremio.com or our GitHub page and download the software and run it on the cluster, either your Hadoop cluster or you could run it in the cloud and connect to your different data sources.

[0:49:08.6] JM: Oh, I see. The thing that is not open source is essentially the reflection builder. If you’re not using the Dremio enterprise product, the business model, you’re pulling from your virtual datasets and you have to schedule yourself when are these virtual datasets going to get materialized and when am I going to pull them into memory.

[0:49:32.4] TS: Actually the only thing that’s now open source. The reflections and the acceleration capabilities are all open source as well. They’re all part of the community edition. What’s not in the community edition is, first of all, kind of the enterprise security. The ability to control, to connect to LDAP, for example, for user authentication, the ability to control access to different datasets. Then the second thing that’s not available on the community edition is some of the connectors to things like Teradata and IBM DB2, so some of these technologies that are much more enterprise oriented.

[0:50:07.6] JM: Okay. Those connectors are — Those are like between the reflection and the client side application, or is it between the on-disk physical dataset and the reflection?

[0:50:22.9] TS: It’s actually the — Dremio will connect as a system. The first thing you’ll do is you go to our UI and you’ll say add source and you’ll connect to your different data sources, and that includes your Elasticsearch clusters, your Hadoop cluster, your MySQL database and so forth. You’re connecting to your different data sources. If you’re using the community edition, you can connect to something like 10 different data sources and there are a couple, just a few that are only in the enterprise edition. That's kind of the difference there.

Regardless of which edition you're using, users can log in, they can create new virtual datasets. As an administrator, you can manage the reflections that are happening behind that are being maintained behind the scenes and help with acceleration. We think that any — Certainly, a startup could get started with community edition and use that in production at any scale up to a thousand servers without a problem and not have to pay anything.

[0:51:19.8] JM: I guess I'm still not quite — I fully believe this model makes sense. It's just I'm having a little trouble understanding it. Where does the Teradata and the DB2 connectors or whatever. You basically described the world in which there is a long tail of ways to access your data, and unfortunately you want to be able to joins between your Teradata or your DB2 or whatever databases were used in the COBOL era, maybe that's DB2, and you want to be able to work with these datasets just like your RDS cluster and your MySQL. You want to use them like they're modern datasets. In order to do that, you should have some sort of connector that plugs them into Dremio, and I'm just having a little bit trouble understanding what that connector is, because that's part of the premium offering.

[0:52:15.0] TS: Yeah. Maybe let's take a step back. Dremio has connectors. As part of our software, we ship connectors to different data sources. Data sources are the Teradata, the Oracle, the SQL Server, the Hadoop, S3 and so forth.

We have a collection of different connectors and we're constantly adding more and more of these connectors, and this is what allows our software to connect to different data sources to push down queries into them, to be able to read data from them and so forth. For users of our — Then when a BI tool, for example, like Tableau, or a tool like Pandas in the Python world connects to Dremio, we look like a single relational database to that tool. We're abstracting away all the different data sources and making it all look like one relational database where you can join things together and so forth.

If you're using our community edition, you're able to connect to elastic search and S3 and Hadoop and Mongo and SQL Server and so forth, but if you happen to have a Teradata cluster, you're not going tob e able to connect to that, because that's only reserved to the enterprise edition.

[0:53:25.9] JM: Okay. All right. I think I got it. The motivation for building Arrow or starting the open source Arrow project, that's pretty interesting, because you were — That was basically a way of crowd sourcing the connectors between the on-disk representations of the physical datasets that you could label as virtual datasets. That was a way of crowd sourcing how do we get those virtual datasets into our execution engine so that we can model it however we want and put it into our Dremio reflections.

[0:54:07.9] TS: Yeah, Arrow, when we talk about crowd sourcing, it's a long game, to create a world where everything in the industry uses one memory format. First of all, it's never going to happen entirely, but it's something that's going to take time. Over the last year we've seen a variety of different technologies now embrace Arrow, and that's, of course, helpful for us as well, and just helpful in general for the industry.

[0:54:34.9] JM: No, I didn't mean to phrase it like it's a Machiavellian thing. It was just like what's the — If we do these three things, will that get that community going in a direction that really lifts our boat in the way — Because it's going to raise all boats, but we've built a really big boat —

[0:54:54.7] TS: Exactly. That's exactly right. Our goal was, "Hey, let's provide value —" The only way you're going to get anything adopted by anybody else is if you provide value to them, right? The fact that we provide Arrow, which is a great library for dealing with data in-memory and solves a lot of problems that a bunch of people otherwise would have to solve themselves, and they don't want to. If there's something open source, they can just adopt. We put that out there.

Why does that benefit Dremio? I think it indirectly benefits Dremio, because Arrow is the memory format that we chose as well, of course. By having other systems talk and use that same format, we can then have overtime better and better interoperability with different systems.

[0:55:43.4] JM: Since we're talking about open source now, I think it's worth taking a step back and looking at the history of this, because it's a pretty crazy lineage. I think I understand the lineage. The dreaml project originally came out of Google, and that's like dreaml. That's phonetically similar to Dremio, so I'm assuming dreaml is an ancestor of Dremio.

Dreaml was kind of this columnar — It's like a columnar — It's a system for doing faster analytics. Big query is based off of dreaml. That's Google's big query service. It's very popular. I think Drill had something to do with — Was Apache Drill, was that kind of the open source version of dreaml?

[0:56:26.9] TS: It's hard to say the open source version of something, because Google builds projects internally. They're not really open source them.

[0:56:34.9] JM: Right. They whitepaper.

[0:56:35.5] TS: Yeah, they run at whitepaper.

[0:56:38.2] JM: Until recently.

[0:56:39.0] TS: Drill was a SQL engine. It is a SQL engine primarily focused on kind of Hadoop, and there are a number of others, like Hive and Impala and Presto that kind of serve a similar purpose of being a SQL engine.

Dremio is a very different type of project obviously just from the fact that the whole kind of acceleration of Query is being able to accelerate queries by orders of magnitude and having a way for users to kind of collaborate and curate datasets in the system and then also being able to push down queries into all these different types of systems.

It's a very different system. You're right. The name Dreamio — We were looking for a short company name that had an available domain name, and that was kind of hard to find. We ended up with that, but it', I'd say, the vision and kind of what we're doing is a lot broader than what these kind of SQL engines were doing.

[0:57:36.3] JM: It's catchy. I like the narrow wall. I think it's funny you compare the dreaml strategy or, basically, the big table strategy where Google released these whitepapers and then the community would sort of like shamble slowly towards the Google infrastructure, and everybody is constantly 10 years behind what Google has, because Google just releases these whitepapers. You compare that to today, the Kubernetes and TensorFlow strategies where they open source it and everybody immediately adopts what Google is doing, and then Google gets to kind of similar to the sort of Dremio/Apache Arrow thing. Google has built the biggest ship, so they get — When the rising tide lifts all boats, they get the most valuable out of it. What do you think about that contrast between the whitepaper strategy and the open source strategy that Google — Do you think this is a shift in Google strategy, or do you think they're going to be selective with the whitepaper versus open sourcing strategy?

[0:58:40.8] TS: I think it's definitely a shift, and I think it started when they kind of started investing more in their cloud infrastructure. They realized that if they want to appeal to companies to leverage Google Cloud, they need something to draw them there, versus Amazon being, of course, the first player in the market and the largest player, and Microsoft having kind of owning the enterprise, if you will. What is their strategy? I think they realized that if they can open source some of these technologies, get developers to start using them. If they're the best place to host, that Kubernetes environment or TensorFlow workloads, then that gives them an advantage.

Yeah, I think it's a smart move on their behalf. I think they also observed what happened with some of these whitepapers, right? Where they are going back kind of to, let's say, the MapReduce days. They read a whitepaper on MapReduce and then it was implemented years later in the open source community. of course, when Google came and said, "We want to provide a cloud service," they couldn't then offer their superior map reduce service because it was a different API and everybody had already built apps on the open source version.

[0:59:56.1] JM: I didn't realize that.

[0:59:56.8] TS: Yeah. I think probably lessons learned in terms of let's get people developing on our APIs early on, right?

[1:00:06.3] JM: Hilarious. It sure is a great world for developers these days.

[1:00:12.2] TS: Yeah, it is. There's a lot of free stuff out there and you can download things, and it's also why even if you look at the world where we're in at Dremio where we're actually closer to that business analyst and data scientist where actually a lot of tools at proprietary, we still thought that the right strategy was growing something open source, be it encourages that kind of bottom-up adoption where people can — They can download it, they love that, their ability to get started. They don't have to talk to a sales person. Developers hate that. I hate that.

[1:00:44.6] JM: Did you build a visualization tool, your own visualization tool?

[1:00:48.5] TS: No. We built kind of from a visual standpoint, our UI looks like Google Docs for datasets. People can create new virtual datasets. They can collaborate, share them with their colleagues and kind of build on top of that. Then there's kind of a dataset editor where they can curate the data, they can massage the data, but we don't provide that kind of last model visualization the way a BI tool would. We very much prefer to partner with companies like Looker and Tableau and Qlik and so forth.

[1:01:16.5] JM: But that Google Docs style stuff, all of that is open source?

[1:01:19.3] TS: Yup. That's all open source.

[1:01:21.3] JM: That seems pretty unique. That seems pretty differentiated. No one does that. Looker and Periscope data and stuff, they don't do that stuff, right?

[1:01:28.8] TS: No, that's correct. Looker very much focuses on kind of the analysis as supposed to getting to data read, and so we actually partner very closely with Looker. We actually share a board member with them.

[1:01:40.8] JM: Oh, of course. Okay. Interesting. All right. In that case, let's kind of — I guess we're drawing to an end. To kind of close off with the contextual stuff, what is the most modern data scientist doing with Dremio? You talked a little bit about the Periscope data/Looker set of tools. What are the tools that the most modern data scientist is using today and what are they doing with them?

[1:02:15.0] TS: Yeah. Another way to think about it is there are companies whose goal it is to provide self-service visualization or self-service kind of data science. These are companies like Looker or Tableau or Microsoft Power BI. That's what they do. Dremio's goal is to provide self-service for everything underneath. If in the past you needed to have a data warehouse and a bunch of ETL tools and cube building technologies and pre-aggregating data and extracting data and all that kind of stuff, we wanted to create self-service of the data layer for that entire data platform.

Then you have an entire end-to-end stack that's self-service, both the visualization, and that comes from Looker, Tableau, Power BI, etc., and everything underneath that comes from Dremio. The data scientist today will use — That term data scientist is a little bit — Is very broad. It means different people. For some people, data scientist is somebody who writes SQL queries. For others, if somebody uses a visual interface, like Tableau or Looker, and then for other people it's more kind of a machine learning person who build models and deploys models and that they may be using something like Python or R for that kind of use case.

[1:03:27.8] JM: The Dremio business model, are you modeling yourself after — Because I'm trying to think of an analogue. It's not really like the MapR world that you come from, because the whole MapR — The MapR and Cloudera and whatever — There's a third one also that I'm forgetting.

[1:03:49.8] TS: Hortonworks.

[1:03:50.1] JM: Hortonworks. Right. It was interesting, because that model of company, that it became sort of like a consultancy type of model, but I think part of that was because this was so hard to do and everybody wanted Hadoop, everybody wanted the big data, but nobody had any idea how to set it up, and so you really needed this group of consultants to come in and help you set up and give you enterprise software that made things easier to use. We're kind of in a different world and Dremio does not need to e anything like that, or am I getting it wrong?

[1:04:23.8] TS: You're 100% correct. It was very refreshing to go from the Hadoop world where you sell somebody Hadoop and now it takes them six months before they get any value out of it, and to a world where on the first day that somebody starts using the system, they're already solving kind of really hard business problems that they were having for a long time. The spark that you see when we demo the product to people and they see the actual demo and they love it, that's really refreshing. We don't sell professional services. There's no need for kind of consultants. We can help them on the first day to install it and integrate with their systems. That's really all they need.

When it comes to kind of the open source and the business model, I would say of the three Hadoop vendors, we're most similar to Cloudera probably, and that we could kind of take the community edition, which is kind of an open source, Apache license version, and then we have an enterprise edition, which has additional functionality that people pay for.

[1:05:21.6] JM: Okay. All right. Tomer, this has been a fantastic conversation. Really technical, interesting product that also very subtle business model. I love talking to you Dremio guys. We should continue to do more shows. I'm looking forward to it.

[1:05:36.6] TS: Yeah, thanks so much for hosting me. I really enjoyed the conversation.

[1:05:39.4] JM: Okay. All right. Wonderful.

[END OF INTERVIEW]

[1:05:44.3] JM: Simplify continuous delivery GoCD, the on-premise open-source continuous delivery tool by ThoughtWorks. With GoCD, you can easily model complex deployment workflows using pipelines and you can visualize them end-to-end with its value stream map. You get complete visibility into and control of your company's deployments.

At gocd.io/sedaily, you can find out how to bring continuous delivery to your teams. Say goodbye to deployment panic and hello to consistent, predictable deliveries. Visit gocd.io/sedaily to learn more about GoCD. Commercial support and enterprise add-ons, including disaster recovery, are available.

Thanks to GoCD for being a continued sponsor of Software Engineering Daily.

[END]