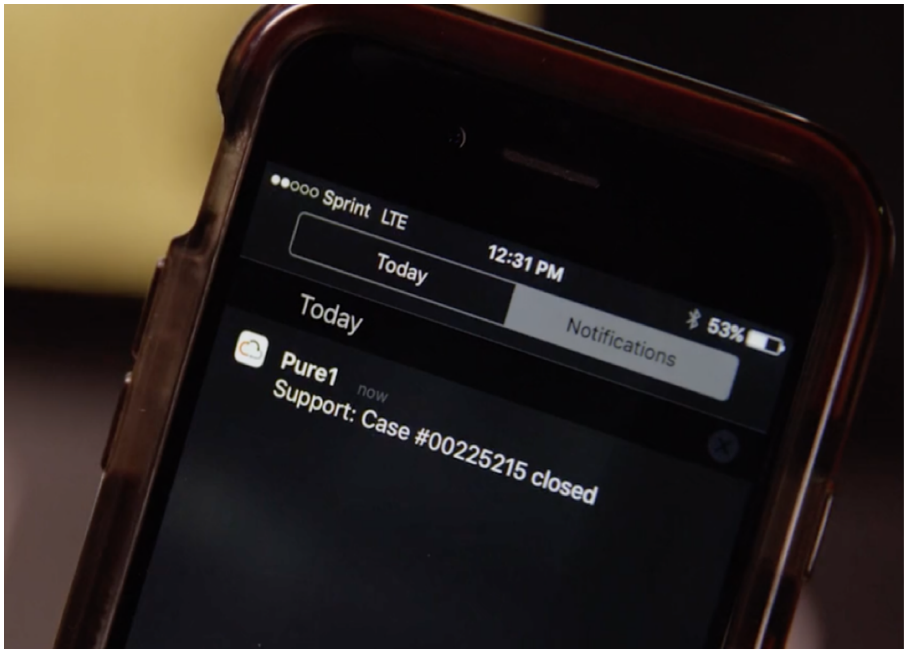
An Inside Look At Big Data Predictive Analytics With Pure1 – Part 1

It’s no secret that our customers absolutely love the refreshing experience that they get with Pure Storage. Our latest Net Promoter Score (NPS) is up to 83 (audited by Satmetrix) and that positions Pure in the top 1% of scores that Satmetrix has seen across consumer and enterprise businesses. One of the key drivers of customer satisfaction is the unique experience delivered with Pure1® support – effortless storage. This means that you can manage storage from anywhere with any device – and with Pure1 predictive support, we work to protect customers before issues happen.

So, as one of the data scientists of Pure1 Global Insight, the capability behind Pure1 predictive support, I wanted to give you a peek inside: how it’s architected, what data is available, how the data is processed, and what results we have achieved.



What Pure1 data is collected?

Pure1 management and predictive support apply both to FlashArray and FlashBlade. In terms of data collection, our arrays constantly generate logs, which capture in great detail the array's hardware and software state. If the array is connected to Pure1, which the majority of arrays are, then it will phone home these logs.

Broadly speaking we can break the logs into two categories:

1. **Frequent telemetry:** this is a JSON object consisting of thousands of data points. This is phoned home every 30 seconds.
2. **Verbose Logs**: the rest of the logs consist of multiple files in different formats with significantly more verbose data about the array. These are phoned home every hour.

This may be a good place to point out that all this data is strictly about the performance and operation of the array. It does not contain any customer data stored on the array or any personally identifiable information.

The Journey: How we got where we are and why?

I joined Pure about 4 years ago as a core developer working to make the Purity OS data structures and algorithms scale for the next big bump in FlashArray capacity. After one year, I rotated onto our escalations team.

The escalations team works seamlessly with the Pure support team, and is an embodiment of Pure's number one value: **Customer First**. A team comprised of core development engineers rotated in from other teams for a period of time in order to rapidly solve any issues that arise for customers. Our customer support has always received rave reviews from customers, and my firsthand experience highlights how we've constantly worked to improve that support – which we don't think can be matched by legacy vendor

arrays and business processes.

As I joined the escalations team, the number of FlashArrays began to increase dramatically and so did the workload. We approached a sink or swim moment where it was clear the old predominantly manual tactics would no longer be enough.

Analysis of what was most time consuming pointed squarely at redundant work. Engineer X got case A assigned to him. It took one week to diagnose the issue and write the code patch. Three weeks later engineer Y had case B assigned to her. Case B was from a different customer but had the same root cause. Since there was no good way to determine this fact, Engineer B spent four days diagnosing the problem only to stumble onto the fix made by the first engineer once she identifies the target block of code. That's a lot of wasted effort.

Our solution was to add one more step to the diagnosis process. Once an issue has been identified, we write a **fingerprint**. A fingerprint is a script that detects the signature of the issue in question from the logs. It returns a binary answer: issue FOUND or issue NOT_FOUND.

Soon we built up a fingerprint library of the most common issues and made a wrapper that would run all fingerprints against one hour of logs phoned home from a given array. We cleverly called it: Mr. Fingerprint. Despite the silly name it was a game changer. Anytime a new case came in Pure support would first run Mr. Fingerprint against the logs. Our initial stats showed that 70-75% of cases that would have gone for investigation to the escalations team were instead caught by fingerprints and solved on the spot.

This had two awesome side-effects:

- The escalations team went from barely keeping their heads above water to having free time to try to improve user experience and array stability.
- Customer experience improved as so many cases were diagnosed on the spot rather than having to go through the escalations team.

My tour on the escalations team was over. I could have gone back to core-dev with a few good stories and scars to show. Instead, a new team was born, the escalations-development team, affectionately known as esc-dev. The charter of esc-dev was to build tools and infrastructure to improve product stability and make customer experience as awesome as possible.

The first project the esc-dev would tackle was very clear: transform Mr. Fingerprint from a reactive tool to a proactive one. Why wait for a customer to initiate a case to run fingerprints? Let's run it all the time and proactively contact customers that have a potential issue flagged.

What followed was a new framework for constantly running fingerprints in the cloud on all arriving logs. What followed after that was learning how to deal with the output of that process. How to filter false positives from real positives. How to minimize false positives and false negatives altogether. How to determine when it is worthwhile to fire an alert because it will be actionable and useful, rather than just noisy.

The analogy we started using is: if your car mechanic calls you and asks you to come into the shop one week before your transmission will fail that is absolutely awesome. But if that same mechanic calls you twice a day with every little thing that may or may not be broken that is both useless and frustrating. The goal was to make sure that Pure1 Global Insight behaves like the former example.

Getting a good balance between signal and noise, making it efficiently run in the cloud and increasing the

number of fingerprints allowed us to deliver Pure1 predictive support a year ago. Since then, we have continued to improve and innovate on its functionality. The fingerprints library has grown to nearly 300. More significantly the capabilities of fingerprints expanded. Fingerprints are able to access the array's results database and assess the history of previous hits or pull stats over multiple-hours, days or weeks. Fingerprints are capable of being built into a sophisticated hierarchy where diagnosing an issue involves hits from multiple different fingerprints spanning some period of time.

An even more powerful tool was the addition of the global metrics database, which is populated with detailed telemetry that is extracted in the process of running fingerprints. We created a giant database of billions of records spanning thousands of different measurements from every customer array over multiple-months. Using data-mining techniques on this database we are able to determine baselines of what is normal and catch deviations. We developed a new class of fingerprints that looks for data anomalies, rather than merely known symptoms. The identification of "there is something weird going on here let's alert the relevant engineering team" created an even more predictive layer in finding issues before they become customer-visible.

Evolution of Diagnosing Issues at Pure1

The journey I have described can be summarized as follows:

Phase 0: (The startup phase)

- Customer reports an issue
- Engineer reads logs to determine root cause

Phase 1: (The first permanent escalations team phase)

- Customer reports an issue
- Engineer uses tools to help read logs faster and determine root cause

Phase 2: (The fingerprints and rapid growth phase)

- Customer reports an issue
- Tools read logs themselves and can find all previously diagnosed issues
- If issue is new: engineer diagnoses and adds to library for tool to find next time

Phase 3: (Predictive support phase)

- Tools scan all customer logs identifying all known potential issues and reaching out to customers before they have an issue
- Customer reports a brand new issue
- Engineer diagnoses and adds to library for tool to find next time

Phase 4: (The near future)

- Tools scan all customer logs identifying all known issues as well as flagging anomalies that may become new issues
- Engineers alerted by tools rather than by customer.
- Fix delivered with 0 customers being impacted

We are in the process of transitioning to phase 4 – again with our focus to improve the customer experience.

Pure1 101: Results

Let's now take a look at some of the results we have achieved in a little over a year:

- **275 real-time issue fingerprints**
- **7,700 incidents automatically identified and resolved**
- **162 severity 1 issues avoided to date**

Stay tuned for part 2, which will explore in more detail the fingerprint execution engine behind Pure1 Global Insight.

In the meantime, check out these two funny videos on Pure1 predictive support that highlight how effortless storage is with Pure when compared to traditional storage:

Support – On Hold

Critical Meltdown