iVoyeur
Cloud Gazing

DAVE JOSEPHSEN

Dave Josephsen is the author of *Building a Monitoring Infrastructure with Nagios* (Prentice Hall PTR, 2007) and is senior systems engineer at DBG, Inc., where he maintains a gaggle of geographically dispersed server farms. He won LISA '04's Best Paper award for his co-authored work on spam mitigation, and he donates his spare time to the SourceMage GNU Linux Project.
dave-usenix@skeptech.org

My Mom, an aerospace engineer by trade, married a physicist when I was 13 or so. As a result I spent some time in my youth hanging around some very cool laboratories at places like Honeywell, Hughes, and Raytheon. There is a curious sort of euphoria shared by scientists and engineers in labs like this, places where interesting work is getting done. It's this feeling, I suspect, that would tempt so many of us to take a janitor's position at CERN, if only to be in the building for a while. I'm a little ashamed to admit that I experience the same sort of bliss when I'm at our co-location facility (which is often). A good friend, former coworker, and fellow pipe-smoker runs the NOC there. When I visit, we'll usually sneak out back, behind the generators, and smoke a bowl together.

I enjoy the banter and the opportunity to exchange whatever tobacco mixtures we're experimenting with, but I also relish the walk to the generators, which affords me the opportunity to pass all manner of strange and colorful appliances (NINJABOX); a half-dissected EMC Clarion (what ARE those guys up to?); a huge, distributed commodity storage installation (BLUE LEDs as far as the eye can see!); the battery room (look at the SIZE of those ground wires!); and of course the 10-ton diesel generators. I'll leave it to the reader to imagine a felicity to match a good smoke with a good friend on the concrete foundation of a 10-ton generator.

At any rate, a few weeks ago, on one such walk toward the generators, a new installation drew my curiosity. By our admittedly meager standards, it was quite large. Rack upon rack of 1U commodity boxes of a brand that rhymes with "blooper psycho." Upon being told it was Akamai, it was driven home to me just how much things had changed in the last five or six years. Remember when the guys with money bought Sun or IBM, or at the very least Dell? Now there is no Sun; only Oracle. And, anyway, the really big guys don't even use Oracle. They've all been hacking away on their top secret MapReduce and DHT implementations for years and years. SANs used to come in pretty boxes with fancy labels on them, and whatever happened to "blades"?

What we're seeing in the colo is just the physical manifestation of what we all know to be true. In the past several years the Google Commodity Hardware Model, configuration management engines, machine and storage virtualization, and, most recently, the NoSQL [1] movement have come together in ways that have changed our profession.

New companies of every kind are as likely as not to turn to a cloud provider or two for their server infrastructure needs. At the same time we appear to have accepted

the failure of relational database systems to scale beyond a certain point, and most of the high-end database research is going into distributed key-value type systems. The big iron is gone and from its ashes have sprung a bazillion no-name 1U boxes, most of which play host to some 30-odd virtual machines. Did you know that you can install Cisco IOS on VMware now? The 1U sea of blooper-psycho engulfs even the switches before our very eyes!

I see in this future some fascinating ramifications for systems monitoring, but, alas, being a guy who just runs a bunch of Linux boxes by trade, I'm at the sidelines of most of this spectacle. This month, therefore, instead of subjecting you to my own conjecture, I've decided to borrow from the insight of someone closer to the center of the tempest, Theo Schlossnagle. Theo is the author of *Scalable Internet Architectures* (Sams) and is the founder of OmniTI. Among OmniTI's offerings is a product called Circonus, which is a hosted monitoring service. He was nice enough to answer the following questions for me via email:

[Dave] Let's start with a quick rundown on OmniTI: What sorts of problems do you solve for your clients? What sort of scale are we talking about?

[Theo] OmniTI covers a fairly wide gamut of Internet architecture challenges—everything from datacenter selection and network and systems infrastructure design and management up through software engineering and into design and usability. All of these capabilities are squarely focused on Internet architectures that service millions to hundreds of millions of users. I focus mainly on data, concurrency, and distributed systems issues at that scale.

[Dave] Circonus and Reconnoiter: what are they and what deficiencies were they built to satiate?

[Theo] Reconnoiter evolved out of a long life of pain experienced by our operations team while managing large, distributed architectures. I was very unhappy with the state (and philosophy) of existing monitoring tools, so I set about to fix that. After running Reconnoiter for a while, supporting various high-stakes Internet sites, we started Circonus to bring the features of Reconnoiter and the philosophies of our engineering and operations teams to the rest of the world within the ever-popular and vibrant world of SaaS.

A few key differentiators of our approach include separating data collection from data processing, structured and safe fault remediation workflow, and data agnosticism. What's all that mean? Basically, you don't have to collect metrics in one system to tell you if they are good and bad (paging someone) and then go collect them in another systems to do visualization. It means you can't acknowledge an alert while sleep deprived and subsequently forget that you did so. Perhaps most importantly, it means that you can collect data from other parts of the organization (such as productivity information, finance information, sales and marketing data), so that the engineering and operations teams more directly know how their work impacts KPIs across the organization as a whole.

[Dave] Describe your typical customer. What factors, in your experience, contribute to a company deciding to employ a hosted monitoring system? Do most of your clients use hosted monitoring solutions exclusively?

[Theo] Yet again, we've built a product that doesn't target a market vertically. While frustrating on the sales and position side, it is very rewarding to realize a system that is so useful to every data-driven organization on the planet. Our

customers are those who want more insight into inputs and outputs in every facet of their organization and believe that the great transparency of that data leads to higher performance and greater accountability. In my opinion, that describes every organization on Earth.

[Dave] Can you talk a little bit about where you see monitoring going in the near future? Do you expect to see hosted monitoring systems replace traditional in-house monitoring, or do they merely augment it?

[Theo] Monitoring systems need to be smarter. For the past 15 years (of my experience), these systems have provided excellent information. It's about time they provide some insight too. The in-house vs. SaaS model is an interesting system. As soon as you stop to think about it, it's pretty obvious. The monitoring system is designed to provide information in good times and in bad. In those bad times, the last place you would ever want your monitoring system located is "in the problem." Off-site, externally hosted monitoring systems are the only right way to do it.

[Dave] This is complicated by issues of data security. So, the big companies that need to control the location of their data will spin up their own, privately managed, external monitoring systems. It will work and look like typical SaaS, but will not be multi-tenancy. I believe the world will go SaaS on this front (although some will use private SaaS).

Do you see cloud customers who are working to eschew an IT infrastructure of their own, turning to hosted services to fulfill their monitoring needs, or rolling their own solutions in the cloud?

[Theo] The only consistent thing I see is people not knowing what to do. I think education is paramount in the world of monitoring. We see a good mix of people trying to launch their own monitoring systems in the cloud alongside the rest of their infrastructure and others leveraging SaaS. I think you just need to ask yourself: am I in the monitoring business?

[Dave] I've heard it said that the cents-per-cycle pricing models used by the cloud providers are incentivizing users away from agent-based monitoring tools and toward external polling engines. Do you think that's true? If it is, does it bode poorly for agent-based systems in general in the future?

[Theo] Agent-based systems and polling systems are both very useful. Any monitoring system that doesn't support both is missing core functionality. I think using the term "agent-based" is misleading here. Instead, think of it as active vs. passive. Is the monitoring system asking the resource a question and receiving an answer? Or is the resource notifying the monitoring system unsolicited? Both are valid, both are needed. I think a monitoring system that only supports one of these methods has a dark future indeed.

[Dave] As sites are getting bigger, we're starting to see more interest in monitoring methodologies that employ statistical sampling (sFlow, for example). Do you guys think it will eventually become infeasible or even meaningless to constantly poll a service on every host in a large environment? What does monitoring for service availability look like for large-scale applications in the future?

[Theo] There are a lots of ways of collecting data, including both sampling and polling. Each is useful in its own context. It's a rather simple question that leads one to the right methodology: is basic statistical information (mean, stddev, cardinality, etc.) over the last bit of time enough to answer my questions? or is that little

bit of data misleading? Looking at something like temperature: I can sample that hundreds of times per second. Is knowing only the average and standard deviation of that over the past 60 seconds misleading? The answer to that (in almost every environment) is no. The average is entirely sufficient and I don't need a breakdown of those samples.

On the other hand, let's look at something where the events are highly decoupled, such as database query performance or Web page load time. Thousands of these events happen each second. Can just the average and a few other token statistical aggregates be misleading? Yes. Here, you can do sampling to get an idea of individual events or (what we do in Circonus) collect histograms instead of simple statistical aggregates, which provides much richer insight into the populations that don't follow a normal Gaussian or gamma distribution.

As with all other things, use the right tool for the job.

[Dave] I've noticed several hosted monitoring services that are themselves implemented in EC2 (browsermob, for example). Will you share your thinking on monitoring from within a cloud provider? Are you guys using any cloud services currently?

[Theo] Perhaps I'm a bit old-school, but there's nothing more important than your monitoring systems being up. The reason we don't use *one* provider is for exactly that reason. I want different bandwidth providers, different routes, different SLAs, and different datacenters for my components. The rule of "no single point of failure" is a simple one; just remember that a vendor is a point of failure. This includes some things people rarely think of: disk vendors, equipment suppliers, datacenters, and cloud providers.

[Dave] With the amount of server and network machine virtualization in use at EC2, it isn't difficult to imagine that a good portion of the monitoring traffic destined for it might be redundant and therefore useless. I'm imagining, for a simple example, 20 ICMP packets from four different customers, destined for 20 addresses that all resolve to VHOSTS sharing the same physical NIC. I'm also imagining the apathy a typical cloud customer probably has with regard to their own monitoring overhead. Do you guys anticipate the inter/intra cloud monitoring burden becoming a problem for the service providers? Is there any potential for systems like Circonus to attempt to detect and optimize for redundant or equivalent outbound monitoring traffic?

[Theo] On the active monitoring side (such as ICMP checks or HTTP requests), I don't foresee any undue burden. Circonus already optimizes the high-frequency cases where several people are running the same check for second-to-second monitoring. We do this to limit the burden on our internal systems more than on those being monitored. When comparing the number of packets/requests in/out from duplicitous monitoring versus regular heavy traffic, we find the monitoring burden to be nominal.

[Dave] I really am fascinated by the changes taking place in database technology lately. Key-value and distributed hash table architectures appear to be all the rage, for the simple reason that they scale horizontally. We used to wonder how we were going to get ACID to scale, and now we appear to have thrown it out the window. Most of the distributed data-tier schemes I read about seem to consider only one failure model—that of some number of nodes failing completely. I personally have witnessed all manner of strange quasi-failures in application tier nodes, so I find

this worrisome. Do you guys have any experience with these systems? Can you talk about their reliability versus the old ACID approach? Are they good enough for the banks?

[Theo] There's a lot of confusion between "old ACID" and "new distributed" systems. First, both are quite old. Naming one as old implies that it is less viable. Both systems are very useful. The resiliency profile achievable with well-designed distributed systems will always be better than ACID-compliant systems (at least, we theorize that now). ACID systems will continue to provide a simpler mental model within which engineers can operate. Database failure is far less relevant than service failure (business services, that is). I have seen (and designed) many systems where a failure at the ACID database level causing a complete database outage is not noticeable by the user. These systems are not obsolete. That said, I have a distributed systems background; when you need resiliency, technologies like Riak and Voldemort are really what's needed. Are they good enough for banks? Sure. Both make promises and keep them; the failing is usually with engineers who do not understand what those promises really mean. Distributed systems are hard.

[Dave] With traditional relational database systems, it was easy enough to have Nagios perform a query against the database server, but now how are we to monitor for data consistency in a post-ACID world with 500 Voldemort nodes across multiple cloud providers? What does the future of monitoring look like in the context of the data tier?

[Theo] Monitoring is monitoring. You should care about a vast number of implementation-specific metrics within your database system (relational or non-relational). You also should care about its overall function, quality of service, and availability. If you can measure those, so can your monitoring system. This problem is not difficult.

[Dave] It's conceivable that a modern distributed database infrastructure may grow organically to encompass multiple co-location facilities and/or cloud services. How can we ensure that a given performance metric is meaningful when the database might respond with greater latency to different network ingress paths? How do we monitor to discover performance problems like this?

[Theo] Expectations are evaluated by measuring and analyzing these metrics. Today, these metrics and expectations are set by humans. The mechanics of monitoring these things are very straightforward. Deciding what your expectations/SLA/QoS are for your datacenter distributed system is a much more involved human problem. Again, distributed systems are hard. OmniTI can help with that, if you're struggling.

[Dave] Can you talk a little bit about how Circonus is architected? I'm particularly interested in data storage. Where did you hit the ceiling with RRD, and how do you guys intend to scale the storage of the metric data you collect?

[Theo] Circonus architecture is quite complex, as it provides many services, and the system is highly decoupled to support our rapid development model and fault-tolerance requirements. Data storage is a bit easier to discuss.

Most of the data collection in Circonus follows the path of the open source Reconnoiter system all coming to a head around a set of processes called stratcond. stratcond is simply responsible for securely pulling data from the field and ingesting it into a storage system. Internally, the storage system we use here is called

"snowth." It is a custom-built database that borrows the distributed principles behind Dynamo and storage structures around time-series data so prevalent in the financial services industry. The system is written in C (with extensibility in Lua) and provides zero downtime on instance failure, seamless node addition and removal, complex data analysis server-side in Lua, and a highly optimized on-disk time series data format that never loses granularity.

[Dave] Can you talk a little bit about OmniTI's direction of combining business intelligence with traditional performance monitoring and fault detection? Any interesting statistical models at work here?

[Theo] Interesting is certainly in the eye of the beholder. Basically, every model we've used successfully has also failed and provided misleading results. Statistical models are very dangerous in the hands of anyone but a highly qualified statistician. Mainly, we use statistical models and their outputs heuristically. Each question deserves its own answer, and one should never assume that one model applies equally well to different questions. It may, but the assumption is usually what leads people to irrational conclusions.

[Dave] It seems to me that if you're in the business of collecting metrics for a large base of users, there is some potential to analyze the aggregated data to detect larger problems like DDoS attacks, BGP convergence issues in the Internet, or failures within a given cloud provider. Do you guys see any potential in leveraging the information you're gathering with Circonus to solve larger problems?

[Theo] That's an interesting question and one that has certainly crossed our minds. We treat our customer's data as sacred, on the retention and integrity side as well as on the privacy side. So we are somewhat limited in what we can do there. We are able to observe these things from the metrics we collect on our own systems (collected by Circonus itself, of course). The recent EC2-east services issues were about as obvious as a nuclear explosion next door. There are already very good tools to detect DDoS and BGP convergence issues, and while those are easy to observe in the data flows in Circonus, I think I'd like to go for something more insightful than that. What? That's the question, isn't it? I'm not sure we have the answer to that yet.

We're actively working on some sophisticated intelligence algorithms that can detect anomalistic points in time-series data in the context of all our data streams with all of the metadata we have attached to each metric. It's a sea of interesting data, but I'd be lying if I said we've made good sense of anything but the surface; we're very excited about our future exploration of this sea.

Did I mention we're hiring data analysts with strong signal processing and mathematics backgrounds?