1. Nagios (& Icinga)

Infrastructure monitoring is a field that has so many solutions… from Zabbix to Nagios to dozens of other open-source tools. Despite the fact that there are now much newer kids on the block, Nagios is a veteran monitoring solution that is highly effective because of the large community of contributors who create plugins for the tool. Nagios does not include all the abilities that we had wanted around the automatic discovery of new instances and services, so we had to work around these issues with the community’s plugins. Fortunately, it wasn’t too hard, and Nagios works great.

We also looked into Icinga, which was originally created as a fork of Nagios. Its creators aim to take Nagois to the next level with new features and a modern user experience. There is a debate within the open source community about the merits of Nagios and its stepchild, but for now we are continuing to use Nagios and are satisfied with its scale and performance. The switch to newer technology, such as Icinga, may be appropriate in the future as we progress.

2. Monit

monitSometimes the simplest tools are the most useful, as proven by the simple watchdog Monit. Its role is to ensure that any given process on a machine is up and running appropriately. For example, a failure occurs in Apache, Monit will help to restart the Apache process. It is very easy to setup and configure and is especially useful for multi-service architecture with hundreds of micro-services. If you are using Monit, make sure to monitor the restarts that it executes in order to surface problems and implement solutions (rather than just restarting and ignoring the failure). You can do this by monitoring Monit’s log files and ensuring that you are alerted to every restart.

3. ELK – Elasticsearch, Logstash, Kibana – via Logz.io

logz.ioThe ELK Stack is the most common log analytics solution in the modern IT world. It collects logs from all services, applications, networks, tools, servers, and more in an environment into a single, centralized location for processing and analysis. We use it for analytical purposes (e.g., to troubleshoot problems, monitor services, and reduce the time it takes to solve operational issues). Another use for this tool is for security and auditing (e.g., to monitor changes in security groups and changes in permissions). After receiving alerts on these issues, it is easy to act on unauthorized users and activities. We also use ELK for business intelligence, such as monitoring our users and their behavior. You can set up your own ELK or buy it as-a-service. We’ve written a guide for the community on using ELK to monitor your application performance.

Disclaimer: Logz.io is our ELK-as-a-service that we use in our own environment. You can say that we eat our own dog food.

4. Consul.io

consulConsul is a great fit for service discovery and configuration in modern, elastic applications that are built from microservices. The open-source tool makes use of the latest technology in providing internal DNS names for services. It acts as a kind of broker to help you sign and register names, enabling you to access service names instead of specific machines. If, for example, you have a cluster of multiple machines, you can simply register them as a single entity under Consul and access the cluster easily. We praise this tool for its efficiency, although we still feel there is more that can be done with it. If you also use it, it would be great to hear about your own use case.

5. Jenkins

jenkinsEveryone knows Jenkins, right? It’s not the fastest or the fanciest, but it’s really easy to start to use and it has a great ecosystem of plugins and add-ons. It is also optimized for easy customization. We have configured Jenkins to build code, create Docker containers (see the next item), run tons of tests, and push to staging/production. It’s a great tool, but there are some issues regarding scaling and performance (which isn’t so unusual). We’ve explored other cool solutions such as Travis and CircleCI, which are both hosted solutions that don’t require any maintenance on our side. For now, however, since we’ve already invested in Jenkins, we’ll continue with it.

6. Docker

dockerEverything that can be said about how Docker is transforming IT environments has already been said. It’s great…life changing, even — (although we’re still experiencing some challenges with it). We use Docker in production for most services. It eases configuration management, control issues, and scaling by allowing containers to be moved from one place to another.

We have developed our SaaS solution with a twelve-layer pipeline of data processing. Together with Jenkins and Docker, we have been able to run a full pipeline across all layers on a single Mac. It would be wrong to say that there aren’t any complications with Docker, as even small containers can take a significant amount of time to build. However, we want to ensure that our developers are as satisfied as possible and enable them to work rapidly. With all of the management involved in storage, security, networking — and everything surrounding containers — this can be a challenge.

We see Docker progressing and look forward to welcoming the company’s new management and orchestration solutions. For those who might be having issues with Docker, we’ve also compiled a list of challenges and solutions when migrating to Docker.

7. Ansible

ansibleAgain, simplicity is key. Ansible is a configuration management tool that is similar to Puppet and Chef. Personally, we found those two to have more overhead and complexity to our use case– so we decided to go with Ansible instead. We know that Puppet and Chef probably have a richer feature set, but simplicity was our desired KPI here. We see some tradeoffs between configuration management using Ansible and the option to simply kill and spin new application instances using a Docker container. With Docker, we almost never upgrade machines but opt to spin new machines instead, which reduces the need to upgrade our EC2 cloud instances. Ansible is used mostly for deployment configuration mostly. We use it to push changes and re-configure newly-deployed machines. In addition, its ecosystem is great, with an easy option to write custom applications.

8. Collectd/Collectl

Collectd/l are nifty little tools that gather and store statistics about the system on which they run and are much more flexible than other tools. They allow users to measure the values of multiple system metrics and unlike other log collection tools that are designed to measure specific system parameters, Collectd/l can monitor different parameters in parallel. We use these two tools to measure customer performance parameters and ship them to our ELK-as-a-Service platform. We’ve specifically wrapped a Collectl agent in a Docker container and push it with Ansible to all of our servers. It collects information every couple of seconds and then ships it to ELK to allow us to run reports and send alerts. If you’d like to see a specific example of how we do this process in our environment and how others can do the same, we’ve created a guide for everyone.

9. Git (GitHub)

githubGit was created 10 years ago following the Linux community’s need for SCM (Source Control Management) software that could support distributed systems. Git is probably the most common source management tool available today. After running Git internally for a short period of time, we realized that we were better suited with GitHub. In addition to its great forking and pull request features, GitHub also has plugins that can connect with Jenkins to facilitate integration and deployment. I assume that mentioning Git to modern IT teams is not breaking news, but I decided to add to it to the list due to its great value to us.