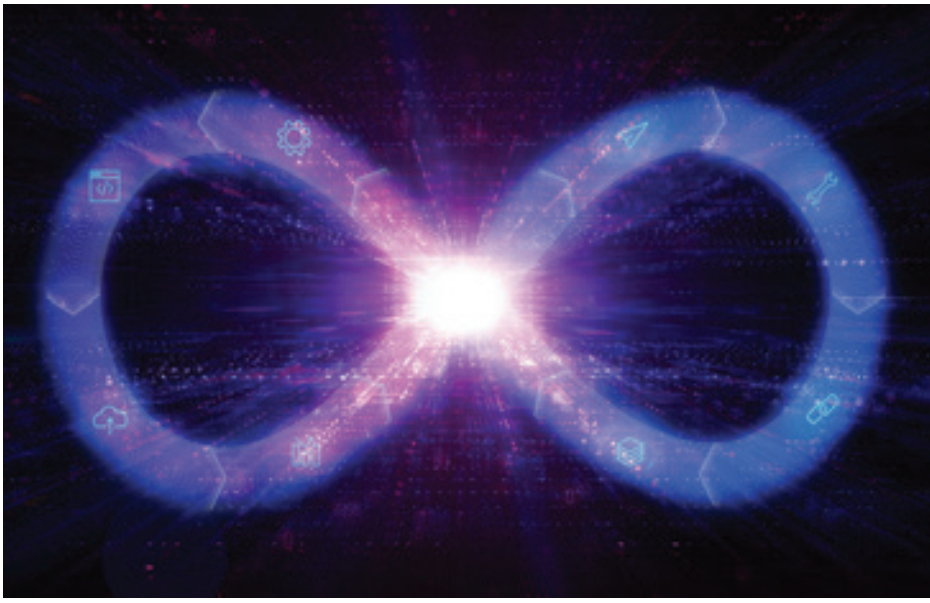
Advancing Sustainability Goals With DevOps

When it comes to sustainability of the technology ecosystem in an organization, there are a number of different things that term can potentially mean. Often, organizations include diversity, environmental footprints, social impact, green computing initiatives, charitable giving and numerous other expressions of their values under the sustainability umbrella.

For those of us on the technical side of the house, the temptation can be to (wrongly) assume that these sustainability goals do not truly impact us—or at least the technical controls in our organization and the broader technology landscape. This thinking is a bit of a trap, though. Why? Because, in fact, there are multiple things that we can do to support—either directly or indirectly—sustainability goals in and through the controls, procedures, processes and countermeasures that we work with every day. In fact, with a bit of creativity, there is a near-infinite number of ways this can be true.

One area where opportunities abound is application development—specifically, in enterprises that have embraced DevOps (and, even better, DevSecOps). The reasons this is true are twofold. First, applications represent one area that is often under-considered from a digital trust standpoint. Application development (and by extension application security) represents a specialized and highly technical subdiscipline of both technology and digital trust. Moreover, this specialization is one that many organizations have not historically invested in building strength. This, in turn, means that there is often much room to improve generally (both in areas that directly intersect sustainability and others). The second reason this is true is automation. The touchstone of DevOps/DevSecOps is the ability to automate. This means that it represents fertile ground for implementing automated controls that can support specific sustainability use cases.

To illustrate this, there are several example areas that showcase how to automate mechanisms that support sustainability in the manner described. Note that the items listed here are just food for thought. They are not intended as an exhaustive list of every possible control that you might decide to implement (or even every control that can have an impact on sustainability). Instead, your particular organization's needs, technology footprint, culture, industry, geographic region and numerous other organization-specific factors determine which might be most appropriate for you.



ED MOYLE | CISSP

Is currently director of Software and Systems Security for Drake Software. In his 20 years in information security, Moyle has held numerous positions including director of thought leadership and research for ISACA®, application security principal for Adaptive Biotechnologies, senior security strategist with Savvis, senior manager with CTG, and vice president and information security officer for Merrill Lynch Investment Managers. Moyle is coauthor of *Cryptographic Libraries for Developers* and *Practical Cybersecurity Architecture*, and he is a frequent contributor to the information security industry as an author, public speaker and analyst.

SBOMs can help stakeholders understand what software is being incorporated into the applications their organization writes, and threat modeling can do something similar for service providers.

The reason the specific areas highlighted were chosen is twofold:

1. They clearly illustrate the connection between sustainability and digital trust (via the channel of software development).
2. They are useful and valuable mechanisms for any organization, so they are valuable regardless of what an organization's plans are relative to sustainability goals. Even if your organization decides to deprioritize ESG, green computing and other sustainability initiatives, you're still likely to get value from these areas.

Example #1: Software Bill of Materials

For initiatives such as environmental, social and governance (ESG) programs, many of the factors organizations often consider first consist of items over which they have direct control. For example, consider core ESG principles such as workforce diversity and sustainability of resources used in facilities (e.g., power, water). Organizations can directly control these to a large degree. They can shift resources to help bolster green (renewable) energy use and make investments in facilities, suppliers or people to help achieve ESG goals.

But the items that organizations have direct control over are not the only areas to consider. Supply chains can play a significant role. It is not just the values associated with what we do, but also the values of the partners and vendors that support us that can be at issue. These include not only the traditional value and supply chains, but also the software supply chains. For example, an organization might ask cloud providers the percentage of power they consume from renewable or green sources; or the enterprise may want to eschew software vendors that support oppressive regimes. The first step to being able to do so is to understand what is being used in the first place: that is, what components are in play and where and how they are used.

There are numerous strategies to investigate this, but one mechanism is a software bill of materials (SBOM). And, in fact, creation of one or more SBOMs is a task that lends itself well to automation as part of the DevOps tool chain and release pipeline. Automation is already implicit in the process due to n^{th}-order dependencies (e.g., software that depends on other software which, in turn, depends on other software, and so on).

There will, of course, be additional effort required beyond merely building an understanding of what software is being used. After all, the binary yes/no of whether you use a given piece of software or not tells you absolutely nothing about the ethics and values of the organization supplying that software. It does however, give you a framework to build upon for future efforts to unpack and analyze the values of those providers (since one needs to know they exist in the first place to be able to work toward understanding their values).

Example #2: Threat Modeling

The second area to highlight is threat modeling. Infrastructure-as-Code (IaC) artifacts can be used for assurance, privacy, security and governance activities. They can also be used both to vet released software and as a way to query the running state of the production environment.

There are ways to automate certain elements of review. For example, when working with a declarative IaC technology (e.g. Terraform, Ansible, SaltStack), the creation of certain artifacts, such as data flow diagrams instrumental to the process of application threat modeling, can be automated—at least partially. For those who are not familiar with the application security space, threat modeling is a process of deconstructing an application into its component parts to allow examination of threat scenarios associated with application entry points. It is a go-to tool in the application security tool kit and can be directly hooked into the release pipeline (assuming IaC is being utilized, of course).

How does this help with sustainability? The most obvious way is that it helps stakeholders understand the flow of the application throughout the entirety of a given use case. This, in turn, allows a much higher-resolution view of what, where and who are involved in the delivery of a given service. SBOMs can help

LOOKING FOR MORE?

- Explore the *COBIT® for DevOps Audit Program*. *www.isaca.org/cobit-for-devops-audit-program*
- Learn more about, discuss and collaborate on emerging technology in ISACA's Online Forums. *https://engage.isaca.org/onlineforums*

stakeholders understand what software is being incorporated into the applications their organization writes, and threat modeling can do something similar for service providers (e.g., Platform as a Service [PaaS] and Infrastructure as a Service [IaaS] dependencies). This is particularly helpful with large, complicated applications that span multiple service providers.

Technologies such as OS virtualization help both in understanding usage and in making workloads more portable to different environments should the need arise.

Example #3: Reallocating Footprint

As anybody who has looked at green computing knows, there is much to unpack in ensuring that an organization's power usage footprint is what and where stakeholders expect. Moreover, once an organization understands what it has and where the power that fuels it comes from, it might very well decide that it wants to shift the balance. For example, an organization might want to shift from environments that are largely powered with fossil fuels to those that are fueled by renewable energy. Doing that, though, is not always easy.

One excellent strategy to help address both problems involves artifacts that are key to DevOps models. Technologies such as operating system (OS) virtualization help both in understanding usage and in making workloads more portable to different environments should the need arise (e.g., to relocate them from one environment to another). Another technology that maximizes portability is application containerization (e.g., Docker). An organization not only can (and should) employ orchestration tools to monitor usage—which is easily automated to correlate to power consumption—but also use these technologies to rapidly move containers between environments.

In fact, microservice architectures using technologies such as application gateways (e.g., KrakenD) or service mesh (e.g., Istio/Envoy) allow teams to relocate these elements even more easily. Teams can also choose to automate elements in keeping with ESG goals. For example, one could choose to run a given workload in cloud provider A (e.g., one that meets cost criteria) until such time as a certain threshold is reached (e.g., carbon footprint, power usage threshold) and then "burst" (relocate) that workload to a different provider with different properties. This allows teams to strategically take advantage of favorable costs while still meeting sustainability targets.

Conclusion

The point of highlighting these examples is to illustrate how digital trust controls can be leveraged to help directly support ESG and broader sustainability goals. Again, there are numerous ways to approach this and the few listed here are not the only options. However, they are highlighted here because, historically, applications have been overlooked. Hence, it is always valuable not only to spend time bolstering them, but also to illustrate how to make progress on sustainability while improving security.