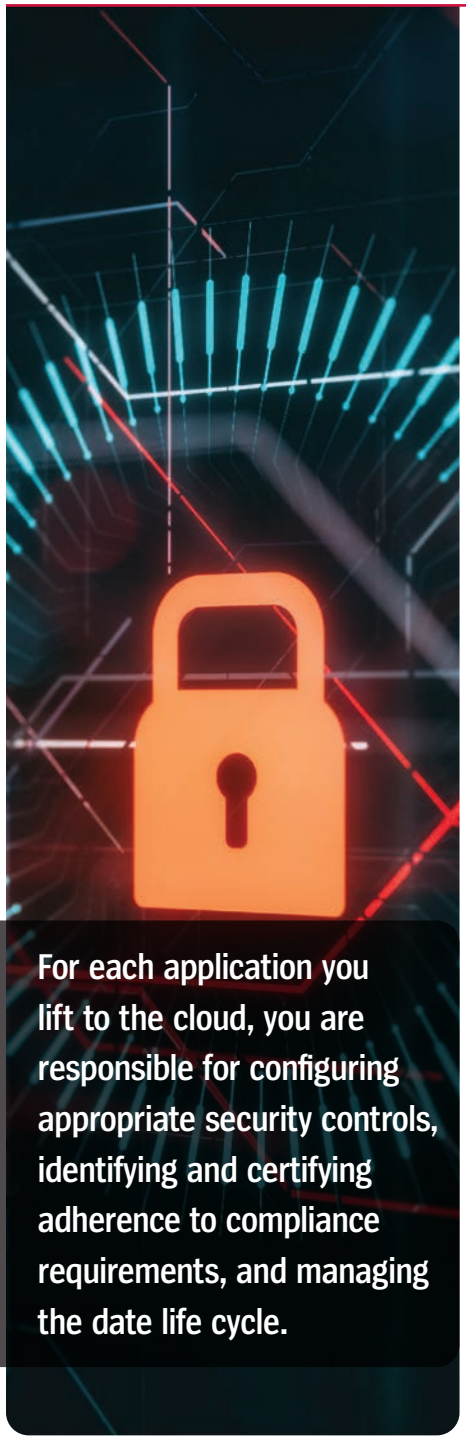
UNISYS | Securing Your Tomorrow®

How to Manage Cloud-Based Security Risks and Governance

By Chandra Gundlapalli



For each application you lift to the cloud, you are responsible for configuring appropriate security controls, identifying and certifying adherence to compliance requirements, and managing the date life cycle.

While offering business efficiencies, cost benefits, and competitive advantages, the shift to cloud-based services (i.e., Azure, AWS and Google Drive) has its own implications — not least of which is security. Many assume that switching to cloud-based services and using their security tools will fix pre-existing security vulnerabilities or deficiencies — but this is simply not so.

Public clouds *do* provide perimeter security for the data stored within their data centers, which is an important function. But that is only *one* consideration among many in ensuring security.

CIOs often talk about the need to build secure containers to accelerate speed-to-market. While there is a recent focus in the cloud space on Kubernetes and containerization, let's not lose the forest for the trees. Security needs are far-reaching, and Kubernetes is just a small part of that equation.

Other critical components include compliance considerations, DevOps automation and creating a robust testing platform that includes static (versus dynamic) vulnerability scanning.

Compliance Considerations

Various industry verticals (e.g., health care) have their own complex, rigid compliance standards that are known to be costly and burdensome to maintain. And updating applications and infrastructure to meet compliance standards often leaves you prone to various security and compliance vulnerabilities.

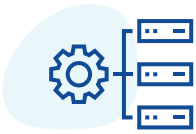
But, as I've written about before, you can combat vulnerabilities by investing in automation and security scanning for code development and deployment. Automating these processes means faster access to information that can identify security vulnerabilities so you can respond to them more efficiently.

Once you identify your security vulnerabilities, it's only natural to wonder which security functions your cloud-based services might be able to fulfill and which you will still need to maintain responsibility for. As a Unisys colleague explained, the reality is that cloud security is a shared responsibility.

As I noted, cloud-based service providers typically handle perimeter security for data you store within their data centers, as well as limited compliance controls for infrastructure. However, for each application you lift to the cloud, you are responsible for configuring appropriate security controls, identifying and certifying adherence to compliance requirements, and managing the data life cycle.

If that seems like a lot to contend with, that's because it is.

So how can you make it easier? By using automation.



By automating risk reporting and monitoring, you'll ensure your network and applications are constantly being monitored.

DevSecOps Automation Culture

Assessing your network and applications on an ongoing, continuous basis and reporting out on any noncompliance or potential risk you identify is an incredibly tedious and traditionally manual process. Imagine pouring over tens — if not hundreds — of compliance standards, making sure each of them is met, and knowing the ramifications if those standards are not met.

One of the challenges to embedding security is changing the organization's mindset. With the DevSecOps culture, developers are not only "security-aware" but can also act as the first line of defense. Understand that a security-aware culture is necessary for the members of all functional teams to report potential anomalies. Translate applicable controls (policy-as-code) into appropriate software components as part of the evolving SDLC process.

However, this is a recipe for mistakes, which could entail a hefty fine.

Instead of exposing yourself to unnecessary liability risk, build a DevSecOps automation culture. By automating risk reporting and monitoring, you'll ensure your network and applications are constantly being monitored.

Testing as a Service

You may be thinking that automation is great, but constant risk monitoring is expensive and requires you to build it from scratch. That's why you should consider testing as a service.

Testing as a service allows you to hire a central organization to handle the testing apparatus on behalf of your company. Such testing is not specific to security. It can also be applied to integration, regression, performance and more. It is typically more cost-effective than creating the infrastructure and organization around such a robust testing platform in house.

There is no need to start from scratch when you can buy out-of-the-box testing capabilities from an outside vendor — or, even better, ask the vendor to create a testing program specifically for your company.

For example, ask them to standardize the end-to-end testing-as-a-service (test planning, execution and reporting) templates with reusability, maintainability and cost-reduction as the core goals. This will give you the flexibility to expand capabilities without any vendor lock-in and make sure you challenge the public cloud providers to do the heavy lifting for you. They might create a framework like the following for you:

- **Framework**: Maven Page Object Model Framework
- **Programming Language**: Java
- **Platforms**: Web app (Selenium), mobile (Appium), API testing (REST Assured), database (PostgreSQL)
- **Testing**: Performance (Jmeter), security through static and dynamic vulnerability scanning (Veracode)
- **Reporting**: Extent Report, Log4J
- **CI (Continuous Integration) and CD (Continuous Deployment)**: Azure DevOps

Forward-looking security stakeholders understand the best way to avoid future liability risk is to implement a robust security program from the start.

Chandra Gundlapalli
Chandra is Vice President Application Services, is a hands-on senior technology executive with 20+ years of proven digital transformation expertise. He is a firm believer in agile team building, one of his key missions is to create a "TEAM of TEAMS" to help articulate a company-wide, client experience transformation vision for Application Services.

He can be reached at
Chandrasekhar.Gundlapalli@unisys.com

When you're creating your strategy around testing for potential vulnerabilities, you will also need to weigh the pros and cons of dynamic or static scanning.

Static scanning is performed with the application off and involves the examination of source code or application binaries to identify security vulnerabilities. If you're examining millions of lines of code, this process can quickly become cumbersome and highly inefficient.

Dynamic scanning occurs while the application is running and entails manipulating the application — how it would be used in the real world — to discover areas of risk. This tends to be more comprehensive, but it is also an inefficient way to scan, as I've found that 10 million lines of code can take upwards of a month.

The takeaway is not to decide between static and dynamic scanning. It's that you should understand the need to challenge testing vendors to develop a better alternative. This alternative could be delivered via technological innovation or improvements in their dynamic scanning. Companies also play a role in optimizing security scanning, including by refactoring and modernizing their code.

But if there is just one lesson you get from this article, it is that you should take charge of your continuous cloud application security. Avoid repeating what you have done with legacy, on-premises deployments. Forward-looking security stakeholders understand the best way to avoid future liability risk is to implement a robust security program from the start.

For more information,
visit www.unisys.com/closingthecloudgap

06/21 21-0262