DevOps & Security at Carbon Black

February 2017

Historically

Traditionally, software developers would release software a few times a year: new major versions with heavy marketing once every year or two, new minor versions with small feature updates a couple times a year and maintenance releases or patches every several weeks.

IT Operations staff, tasked with building, monitoring and managing systems, would take the software from the developers, install it in a lab, confirm compatibility with the other software in use, confirm compatibility with the business's procedures and validate the stability of the newly released software.

A separate team of Security Operations staff would validate the security of the newly developed software and monitor the systems post-deployment for security issues.

The separation of responsibilities works for each discipline, regardless the organization. It scales from small manufacturing businesses to large financial organizations with few changes. In every organization, at every size, in every market, the roles were clear: developers write code, IT manages systems and Security audits, monitors for and responds to breaches. This model also works when developers are in a different organization than IT and Security - the model enabled the on-premise Enterprise Software business as we know it today.

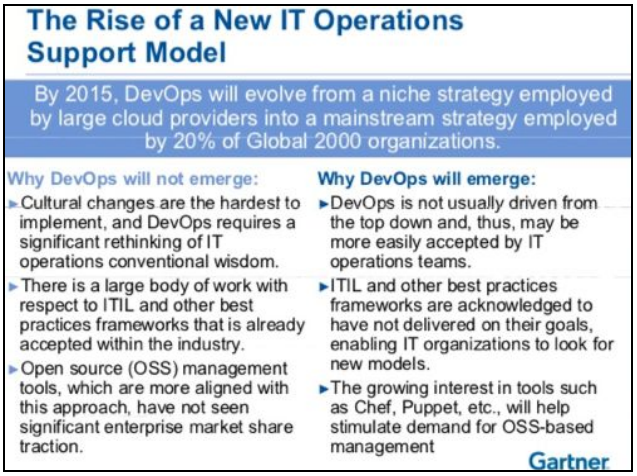
Over time, in regulated and high-risk industries, compliance and security policies trickled down from audit to security to IT and to developers. A wide array of IT regulations emerged that codified the best practices for development, IT and security, such as NERC, HIPAA, HITECH, PCI, GLBA, SOX and SSAE-16 SOC2. Alongside the regulations, a similar array of governance models emerged to help IT executives build their programs from organizations like NIST, COBIT, ITIL and ISO. Both the governance models as well as the regulations describe best practices developed in the context of the development, IT operations and security operations conventions of their time.

DevOps Emerges

The early internet companies built their operations around these conventions, because "that's how it was done." Over time however, the pressure to move quickly caused these companies to optimize the relationship between Developers and IT Operations.

Flickr® is attributed with the origin, based on a presentation by John Allspaw and Paul Hammond at the 2009 O'Reilly Velocity conference: 10 Deploys a Day, Cooperation between Dev and Ops at Flickr. In 2010, community momentum behind these principles grew and toolchains like Chef, Puppet and Vagrant spread. In 2011, just two years after the introductory Flickr presentation, Gartner made it official in their report *The Rise of a New IT Operations Support Model*. This report predicted "by 2015, DevOps will evolved from a niche strategy employed by large cloud providers into a mainstream strategy employed by 20% of Global 2000 organizations." "DevOps" was driven across The Chasm and into the mainstream.

"Servers in today's datacenters are like puppies - when they get sick, everything grinds to a halt while you nurse them back to health. DevOps teams manage servers like cattle. It takes a family of three to care for one puppy, but a few cowboys can drive tens of thousands of cows over great distances, all while drinking whiskey."
Bill Baker, Microsoft, 2014



The culture of technology companies allowed these teams to innovate, without the burden of compliance or existing convention. They manage IT Operations with development principles such as versioned source control, ruthless automation and repeatable processes. We now call this "infrastructure as code." Treating infrastructure as code allows teams to ship new software multiple times per day while still maintaining quality and consistency. We call this "continuous delivery."

These advances in the management of development and IT operations best practices have significantly increased productivity and quality, while maintaining a strong security posture. Puppet Lab's State of DevOps report concludes DevOps-driven IT Operations teams:

- Have 30x more frequent deployments
- Have 60x fewer failures
- Have 168x faster recovery from issues
- Have 50% higher market capitalization growth over three years
- Are 2x more likely to exceed profitability, market share and productivity goals

As enterprises migrate away from on-premises data centers to cloud-hosted applications, enterprise software providers can adopt DevOps principles and achieve the productivity and quality improvements previously limited to consumer cloud products. Carbon Black uses DevOps principles for all cloud-hosted applications.

DevOps & Security

The early consumer-focused internet companies are not strongly influenced by regulations or compliance. They have remained secure, but they do not allow regulatory or compliance requirements to dictate their procedures.

Industry verticals with more stringent regulatory and compliance burdens are only just starting to adopt DevOps principles. They are discovering the traditional security and audit expectations do not apply well to DevOps teams.

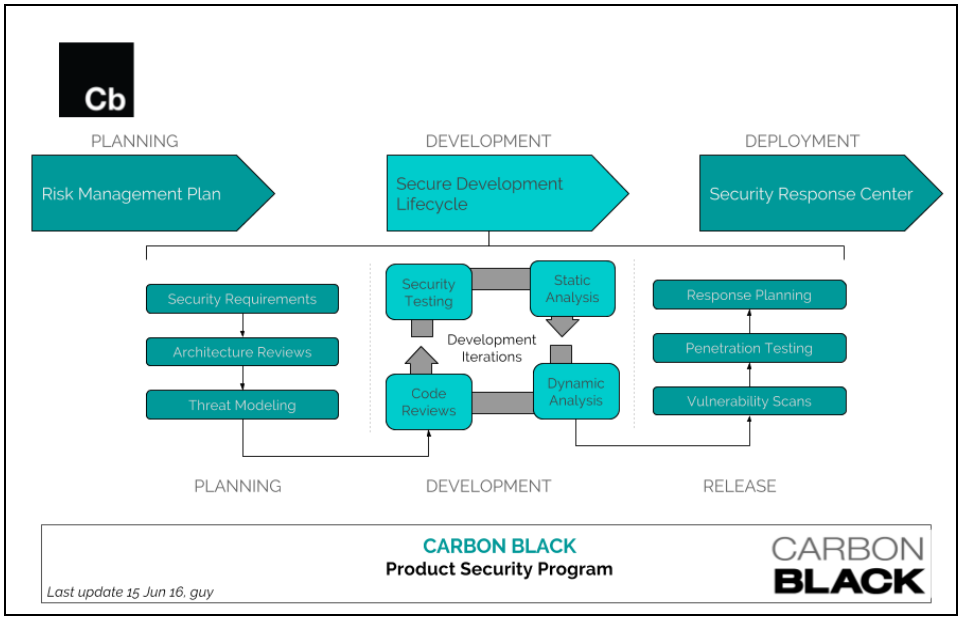
Existing regulations and governance guidelines assume traditional IT practices where infrastructure is static and procedures are manual. When servers are dynamic and procedures are automated, these guidelines break down. There are several initiatives to modernize the governance and regulatory guidance, but generally accepted best practices have not yet emerged.

Carbon Black's Approach to DevOps & Security

With normal software development, the product is the source code. In a DevOps culture, infrastructure is source code too. The provisioning and configuration of all infrastructure is in version controlled source code. Servers are built, deployed and tested continuously - just like the application. Any manual system configuration is a bug that gets ticketed and prioritized like other product bugs.

As a result, the product is not just the application source code, but the infrastructure source code. *Infrastructure is product too.* While this distinction can be muddy for legacy applications retrofitted to cloud deployments, it is mostly impossible to separate the application from the infrastructure in modern applications built exclusively for the cloud. This was the key driver for the original DevOps innovations: infrastructure is as responsible for end user experience as the application itself.

Since infrastructure is built and deployed following the same procedures as the application, we can apply the same Secure Development Lifecycle principles. This means both our application as well the supporting infrastructure are Secure By Design. Carbon Black's Product Security Program is documented in this whitepaper. We use these same security controls when developing and deploying infrastructure.



An Example: Configuration Control/Management

For example, a key element traditional IT Operations audits are concerned with is configuration control. Since server configurations are maintained via manual procedures and the systems themselves is the only source of ground truth, it can be very challenging to manage change in an organization of any size. As a result, configuration control best practices are also manual, with administrative procedures to maintain a central repository of all configuration items and in-depth coordination procedures to review and approve all changes before a system administrator is authorized to make the change in production systems.

When infrastructure is code, the infrastructure source code is checked into a version control system, eliminating the need for manually maintained documentation hosted in a central repository. Each change is subject to automated QA tests, eliminating the need for coordination to validate proposed changes will not affect interoperability with other enterprise systems. Each change is also subject to a code review, streamlining the coordination required to review security-sensitive changes. Finally, each change triggers security-specific tests as well as dedicated security testing tools such as source code analysis and vulnerability scanners, providing additional controls against human error.

Practically speaking, traditional software code change procedures provide richer and more effective configuration control than traditional IT configuration control procedures. They are simply different and require an understanding of modern software development to fully appreciate.

Summary of Security Procedures for Infrastructure Management

Any infrastructure change requires a ticket. It is included in the development backlog and cross-prioritized just like any other new feature or bug fix in the core product. If we need a new log source added, it gets a ticket. If an internal-only application support user interface needs more advanced RBAC (Role Based Access Control), it gets a ticket.

Any major change to the infrastructure that substantially changes attack surface requires threat modeling and architecture reviews. Just like major new product features get careful security consideration, so too do major infrastructure changes.

Since all configurations and all changes are checked into version control, there is a robust audit trail with complete revision history and attribution. The code review and merge request process provides change controls. Finally, integration with the development ticketing system gives rich context and traceability of each change.

Infrastructure configuration “bugs” (like too loose firewall rules) can introduce security issues as easily as application bugs (like improper authentication logic). We use the same mitigation process for both: code reviews, security-specific QA tests and source code analysis tools. These mitigation procedures are code themselves and iteratively improved and enriched.

Finally, infrastructure must have procedures for managing vulnerabilities post-ship, just like the application. Where the Carbon Black Security Response Center is mostly an internal administrative activity, security monitoring for infrastructure is an operational activity. Security monitors are managed alongside the health and availability monitors, paging on-call engineers when suspicious activity is observed 24x7x365. System and application logs are configured to feed into central systems for full-time security analysts to review other suspicious activity.

Why this is important to you

Many Carbon Black customers are deeply concerned about the security of cloud systems. We share your concern and are committed to maintaining a secure platform for your service. Our annual Risk Management Plan, described in the Product Security Program whitepaper, includes not just the security of our software, but also our cloud systems. These results are not reviewed by just the Engineering and Security personnel responsible for your systems, but both the

company's executive team as well as the entire board of directors. There is nothing more important to Carbon Black than providing a secure platform for your business.

Carbon Black's customers are primarily large enterprises with traditional IT operations and traditional information security programs. Many enterprise customers expect us to maintain security programs that follow *their* guidelines. Some require completion of detailed checklists and questionnaires, written with the assumption we follow traditional IT and security procedures typical in most enterprises. Since DevOps security procedures do not align well with traditional procedures, answering those questions honestly often leads to confusion and frustration, rooted in a basic misunderstanding of our security and operational philosophy.

We hope this description helps you better evaluate our practices and procedures. We care about your security. We are confident our operational procedures follow the best practices of modern cloud companies and that we are doing the best we possibly can to protect your data while still providing the service level you expect from cloud providers.

Additional Reading

- Carbon Black's Product Security Program. Our Secure Development Lifecycle, Nov 2015.
- Security and Privacy at Carbon Black. Our Operational Security Procedures, Feb 2017.
- An Unlikely Union: DevOps and Audit. DevOps Enterprise Forum, IT Revolution, Portland, OR, 2015
- Measuring Efficacy, Effectiveness and Culture to Optimize DevOps Transformations. DevOps Enterprise Forum, IT Revolution, Portland, OR, 2015.
- Mythbusting DevOps in the Enterprise, DevOps Enterprise Forum, IT Revolution, Portland, OR, 2015.
- Keeping the Auditor Away: DevOps Compliance Case Studies, Gene Kim and James Deluccia,