PUTTING THE SEC IN DEVSECOPS

ANDREW HARDIE

THAT WAS THE SEC THAT WAS

• Hard shell, soft centre
• Perimeter around the corral, Wild West inside
• Focus on people security, not process security
• Manual procedures
• Hard boundary between Dev and Ops
• “Throw it over the wall” mentality
DEVOPS CHANGES EVERYTHING

• What’s DevOps anyway…?

• DevOps comprises the tools, techniques and practices for the rapid, reliable and repeatable delivery and deployment of infrastructure and application artefacts, with:
  • automated tests for validation
  • security procedures for assurance
  • logging for observability
  • metrics for improvement

DEVOPS IS NOW MUCH MORE THAN…

• Automated application code compile, package and test
• Deploy to an artefact repository
• That’s CI (Continuous Integration)
DEVOPS IS NOW MUCH MORE THAN...

• Automated application code compile, package and test
• Deploy to an artefact repository
• That’s CI (Continuous Integration)

• On-demand deployment of those tested ready-to-run artefacts into target environments
• That’s CD (Continuous Deployment)

DEVOPS IS NOW MUCH MORE THAN...

• Automated application code compile, package and test
• Deploy to an artefact repository
• That’s CI (Continuous Integration)

• On-demand deployment of those tested ready-to-run artefacts into target environments
• That’s CD (Continuous Deployment)

• Rolling the two together, automated all the way through from code commit
• That’s (the other) CD – Continuous Deployment
SO, WHAT’S CHANGED?

• In a word: Infrastructure
  • In the beginning, your own iron/tin – hand crafted, limited automation
  • Then came the cloud (hardware substitution) – 10 years ago
  • Then came automating cloud instance creation (the start of infrastructure as code) – 8 years ago (CloudFormation)
  • Then came Docker and containers – March 2013
  • Then came Kubernetes – June 2014
  • And everything changed…

WHICH MEANS WHAT?

• DevOps now covers the entire SDLC
• DevOps now covers all the traditional system & environment setup
• DevOps now covers all the traditional application provisioning, configuration, maintenance and deletion
• DevOps is thus now (at least) 90% of your IT strategy
• DevOps is now, de facto, your new IT strategy
• The old ways of working have to be transformed (exterminated)
WHICH MEANS WHAT FOR SECURITY?

• “Ops” staff (whether in-house or at service providers) as the gatekeepers to infrastructure and root access is no longer a sustainable practice.
• “Take a ticket and wait” for infrastructure resources creation is no longer a sustainable practice.
• Waiting for Ops to install and configure applications on that infrastructure is no longer a sustainable practice. (The cost of even large disposable infrastructure is almost always less than the cost of holding up a development team.)
• Root access to create infrastructure and then access that infrastructure to configure it must be eliminated. Indeed, “root” becomes a non-event in terms of access control for staff. There is nothing to access!

WHICH MEANS WHAT FOR SECURITY?

• Only systems will have root-level access to other systems, if at all.
• No more shared root password for the Ops oppos!
• People configure what that root access will do via code and config files, to be run automatically.
• Trust thus shifts from who has root system access to who has access to the code that will run as root. Think about that!
BUT, WAIT, THERE’S MORE – MUCH MORE…

- Source code repo access – covered.
- But that’s only for the code you write – what about the rest? (>50%)
- Security at the other end? Contractors, maybe. Open Source no!
- Signed commits? Commit history. Checksums/digests?
- Import procedures
- Secrets scanning
- With everything as code, your source code repo is your crown jewels!

AND YET MORE…

- Artefact repository security – who/what can commit?
- Who/what can access?
- Third party binaries – “sheepdip” import, verification, etc
- Third party repositories – compromise risk
SOURCE CODE AND ARTEFACT HANDLING

• Supply chain integrity
• Multi-level audit trail – repo, company, item, persons
• If any one of these is found to have been compromised, need to know FAST if and where that code or artefact is running in my estate

SDLC SECURITY

• Bearing in mind the SDLC now runs from code commit to production deploy...
• Pipeline runner security:
  • Source code access (esp for IoC code)
  • Artefact repo commit access
  • Infrastructure create/configure access
  • Artefact deploy access
  • Network & service mesh config
THE SURROUNDING LANDSCAPE

- Logging integrity – immutable logs ideally
- Metrics integrity – diversionary tactics by intruders
- Tracing integrity – will it detect MITM net activity?
- Policy control – what can do what and with whom
- Access control – certificates, keys, etc
- Zero trust networks – nothing to see here, move along…

SDLC PROCESS SECURITY

- Accelerate (you read the book, right?) but let’s be careful out there… 😊
- GitOps – env promo as code, automated; but if/when is human approval required?
- Regulatory or “risk theatre”?
- How many tests are enough? Balance between automation and humans…
- If it’s immutable it must be killable – how is that controlled?
- If it’s scalable, how is scale controlled?
- There are no more CMDBs (Ops confession books) – the code describes it!
INCIDENT RESPONSE

• Automated incident response/remediation – don’t rely on it too much…
• If it’s immutable, no hot fixes! So, how fast can you replace that cow?
• What checks are safe to skip for that? NFT? What else?

THREAT SUMMARY

• The Dev Zone – repo access security, not endpoint access security
• Dev account integrity – identity hijack – attacker commits
• The dependency chain – can run far and wide
• Artefact and image repository integrity and access control
• The connectivity matrix risks – APIs as threat points
• Keep it dark – zero trust networks, segmentation, etc
TAKEAWAYS

• This is new
• This is different
• This is hard
• Remember, most successful compromises are either:
  • Via social engineering – e.g. phishing emails
  • Via zero-day exploits – e.g. Apache Struts

TAKEAWAYS

• It’s now mostly not about edge/boundary attacks (aka firewalls)
• It’s the penetrate to the inside, then exploit, e.g. via:
  • Compromised user accounts
  • Compromised imported code or binaries or container images
  • Supply chain integrity – perhaps the hardest problem right now…
  • The “sleeper” compromise – only activated when something interesting happens
  • The compromise you discover only later: exfiltration, reputation destruction, extinction
THE GOAL

• Shift left!
• You shift testing left, so why not security?
• Turn DevSecOps into SecDevOps…

QUESTIONS?

ANDREW HARDIE
BCS@DEVOPERATIVE.COM