CONTROLLED CHAOS
The Inevitable Marriage of DevOps & Security

Kelly Shortridge (@swagitda_)  Velocity Berlin 2019
Hi, I’m Kelly
“Chaos isn’t a pit. Chaos is a ladder.”
— Petyr Baelish, *Game of Thrones*
Infosec has a choice: marry DevOps or be rendered impotent & irrelevant
Infosec won’t survive in a silo. It must be embedded in software delivery.
DevOps can learn to carve its own path to secure software delivery
How can controlling chaos create a marriage of infosec and DevOps?
1. Chaos Theory
2. Time to D.I.E.
3. A Phoenix Rises
Chaos Theory
Chaos engineering = continual experimentation to test resilience
“Things will fail” naturally extends into “things will be pwned”
Security failure is when security controls don’t operate as intended.
What are the principles of chaotic security engineering?
1. Expect that security controls will fail & prepare accordingly
2. Don’t try to avoid incidents – hone your ability to respond to them
Game days: like planned firedrills
Prioritize security game days based on potential business impacts
Decision trees: start at target asset, work back to easiest attacker paths
Determine the attacker’s least-cost path (hint: it doesn’t involve 0day)
Your goal is to raise the cost of attack, ideally beginning at design
Time to D.I.E.
We need a model promoting qualities that make systems more secure
Enter the D.I.E. model by Sounil Yu: Distributed, Immutable, Ephemeral
Distributed: multiple systems supporting the same overarching goal
Distributed infrastructure reduces risk of DoS attacks by design
A service mesh is like an on-demand VPN at the application level
Attackers are forced to escalate privileges to access the iptables layer
Immutable: infrastructure that doesn’t change after it’s deployed
Immutable infra is more secure by design – ban shell access entirely
Patching is no longer a nightmare with version-controlled images
Ephemeral: infrastructure with a very short lifespan (dies after a task)
Ephemerality creates uncertainty for attackers (persistence = nightmare)
Installing a rootkit on a resource that dies in minutes is a waste of effort
Optimizing for D.I.E. reduces risk by design & supports resilience
A Phoenix Rises
Begin with “dumb” testing before moving to “fancy” testing
D.I.E.ing is an art, like everything else
Controlling Chaos: Distributed
Distributed is mostly covered by the existing repertoire of chaos eng tools
Repurpose these tools, but make attackers the source of failure
Multi-region services present a fun opportunity to mess with attackers.
Shuffle IP blocks regularly to change attackers’ lateral movement game
Test: inject failure into your service mesh to test authentication controls
Controlling Chaos: Immutable
Immutable infra is like a phoenix – it disappears & comes back a lot
Volatile environments with continually moving parts raise the cost of attack
Create rules like, “If there’s ever a write to disk, crash the node”
Attackers must stay in-memory, which hopefully makes them cry.
Bonus: disallowing all local IO improves service reliability
Metasploit Meterpreter + webshell: Touch passwords.txt & kaboom
Build your Docker images with a garbage-filled “bamboozle layer”
Mark garbage files as “unreadable” to craft enticing bait for attackers
A potential goal: architect immutability turtles all the way down
Test: inject attempts at writing to disk to ensure detection & reversion
Treat changes to disk by adversaries similarly to failing disks: mercy kill
Controlling Chaos: Ephemeral
Most infosec bugs are stated-related – get rid of state, get rid of bugs
Reverse uptime: longer host uptime adds greater security risk
Test: change API tokens & test if services still accept old tokens
Test: retrograde libraries, containers, other resources in CI/CD pipelines
Test: inject hashes of old pieces of data to ensure no data persistence
Leverage lessons from toll fraud – cloud billing becomes security signal
Test: exfil TBs or run a cryptominer to inform billing spike detection
Chaos/resilience are natural homes for infosec & represent its future.
The future of infosec involves unified responsibility & accountability.
Security can be innovative and fuel the engine of business as well.
“You must have chaos within you to give birth to a dancing star.”

— Friedrich Nietzsche