CIS™ Center for Internet Security®

Security and chaos engineering

Sean Atkinson

Chief Information Security Officer

Chaos engineering

The principles of chaos engineering

“Chaos Engineering is the discipline of experimenting on a distributed system
in order to build confidence in the system’s capability
to withstand turbulent conditions in production.”

Reference: http://principlesofchaos.org/

Practice

1. Gauge what is normal or the steady state
2. Define the control group and experimental group
3. Introduce the variables or “chaos”
4. Compare the control and the experimental

Once we have introduce the issues to the experimental group can we define the capability of our systems to identify, detect, protect , respond and recover from those situations.

The aim: the harder it is to deviate the steady state in the experimental group compared to that of the control group. It builds confidence in the overall resiliency of the underlying system.

Improvements to the process

As this experiment will either reveal:

- Multiple issues that have severely impacted the steady state of the system
- The steady state did not vary at all between the control and experimental

Solution:

Improve resiliency – utilize the findings to improve the gaps and try again, the measure should be falling and as it falls changes to the hypothesis should be introduced.

Experimental design – if the system is completely resistant to any variation, maybe its time to change the test or re-evaluate the design of the experiment.

Improved experimental design

1. What measure are being used in order to define the steady state?
2. Don’t run the same experiments, change as threats change to your environment
3. Move the tests to production, use an authentic system to provide reliance on the experiments outcome
4. Automation
5. Contain the blast radius – moving into production will require that you do not severely impact stakeholders, it’s a short term negative impact not destroying your business

Agile application

Expect failures. A component might crash or stop at any time. Dependent components might fail or stop at any time. There will be network failures. Disks will run out of space. Handle all failures gracefully.

Keep things simple. Complexity breeds problems. Simple things are easier to get right. Avoid unnecessary dependencies. Installation should be simple. Failures on one server should have no impact on the rest of the data center.

Automate everything. People make mistakes; people need sleep; people forget things. Automated processes are testable, fixable, and ultimately much more reliable. Automate wherever possible.

Reference: https://opensource.com/article/18/6/security-experimentation

Agile application

(Security) Chaos Experiments are foundationally rooted in the scientific method, in that they seek not to validate what is already known to be true or already known to be false, rather they are focused on deriving new insights about the current state.

Security Chaos Engineering is the discipline of instrumentation, identification, and remediation of failure within security controls through proactive experimentation to build confidence in the system's ability to defend against malicious conditions in production.

Reference: https://opensource.com/article/18/1/new-paradigm-cybersecurity

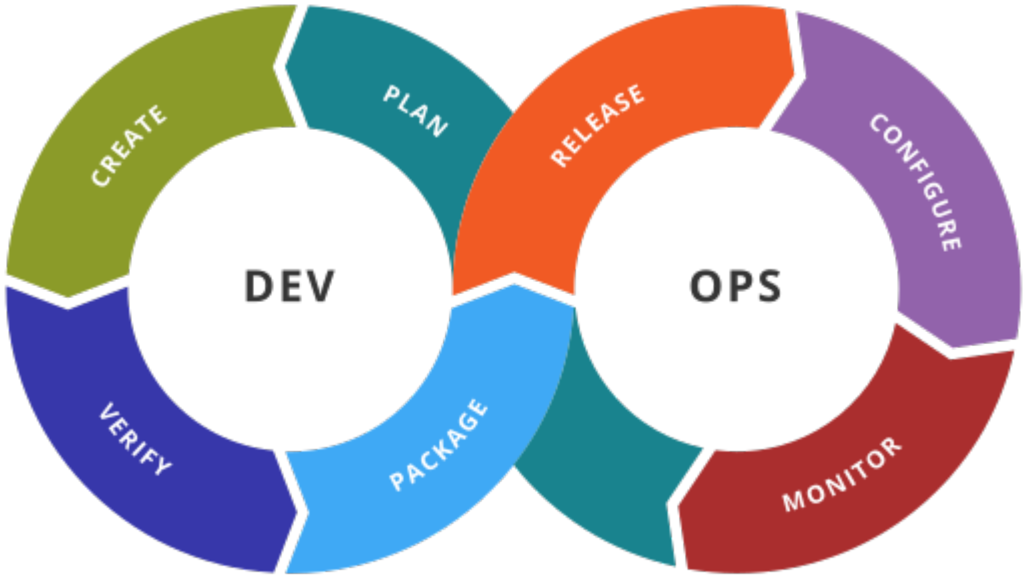
Agile application

DevOps

Where to integrate the testing?

What is the weakest link or what is it that I want to test?

As a monitored process and agile in its approach to system deployment and integration the systems should be set to identify deviations from the “norm”



Roles

Chaos Engineer: Managing the chaos to apply for a given test. This will be determined by:

- Plan the experiment or test to be executed
- Contain the blast radius – in most cases the experimental design is to learn or test the hypothesis not attack an entire infrastructure
- Scale the experiment to make sure that you have gained some understanding of resiliency.

The chaos engineer can and should monitor the outcome of the experiment, was anything gained from the test, if not then the test should be scaled in order to gain value from the test

Experimental Design

Defining the experiment is crucial to understand what and how a plan of chaos is to be administered.

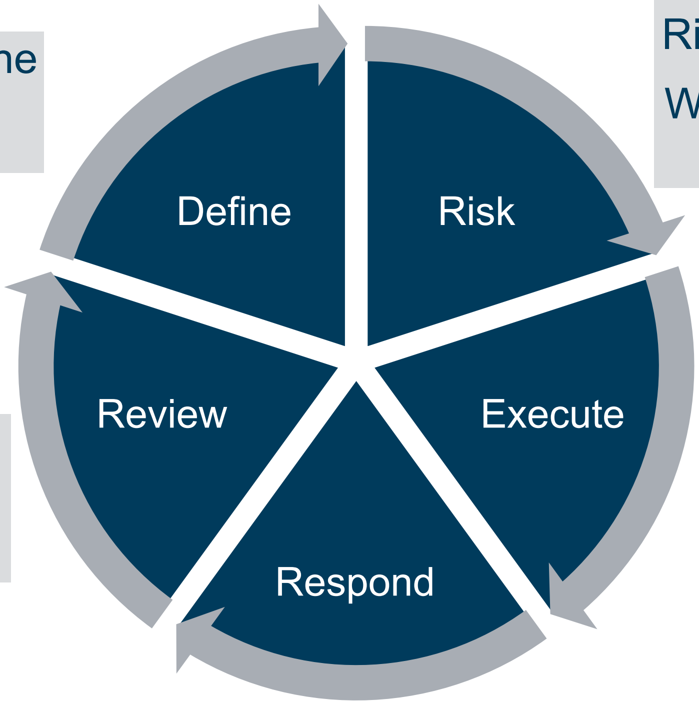
1. Hypothesis – what do we think will happen as a result of the test?

2. Create the conditions that will be scaled in order to test the hypothesis but have a blast radius of the smallest possible range to gain enough information to conclude if the hypothesis was correct or not.

3. Measure the impact, the consequences and if the conclusion of a successful test or a failure of the experiment, not the underlying system'

If the system fails and that was hypothesized the experiment is a success and now a new identified weakness, gap or vulnerability can be fixed.

Security Lifecycle Approach



Lessons learned, fix the gaps or build better tests

Risk based approach

What needs to prove resilience?

Apply the test to the environment

Metrics and review – detection

Resilience, time to detect, survival rate

Red/Blue/Purple vs security experimentation

Security experimentation is considered a differentiation of the same ideals as security testing. The goal is to proactively identify security failures (human factors, poor design, or lack of resiliency). The goal of Resilience Testing (RT) or Penetration Testing (PT) exercises is to exploit vulnerabilities.

Security experimentation is based on simple isolated and controlled tests rather than complex attack methods. The requirement is focus to look at specific controls and identify the weaknesses directly affected. Risk management and blast containment is crucial and using specific tests rather than a complex set of changes may create more noise than actionable results.

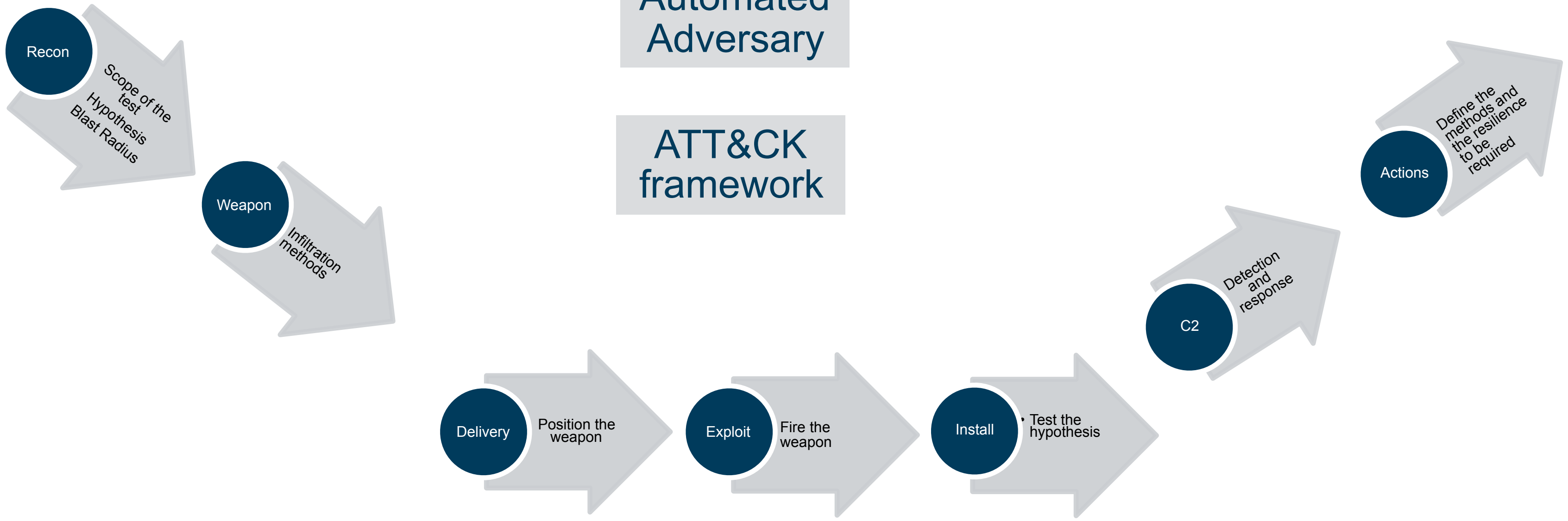
Red/Blue/Purple vs security experimentation

The intention of security experimentation is to create a continuous learning environment to expand the capability of resilience and reduce the affect for design and human failures. While risk management looks at starting with small, focused tests the end results is to expand the confidence in security experimentation to automate and run on production systems. (Integrated method)

RT and PT exercises predominantly focus on the malicious attacker or adversarial approach, these can be utilized as we build confidence to understand and apply such methods into our systems. This becomes a precursory event to hypothesis what RT and PT will find.

RT/PT and security experimentation are complementary, these add enforcement actions between each other with an objective approach to the process. Failures in security experimentation should translate to a greater resilience to RT and PT, weaknesses found in RT and PT will contribute to new areas of experimentation.

Chaos kill chain



Automated Adversary

ATT&CK framework

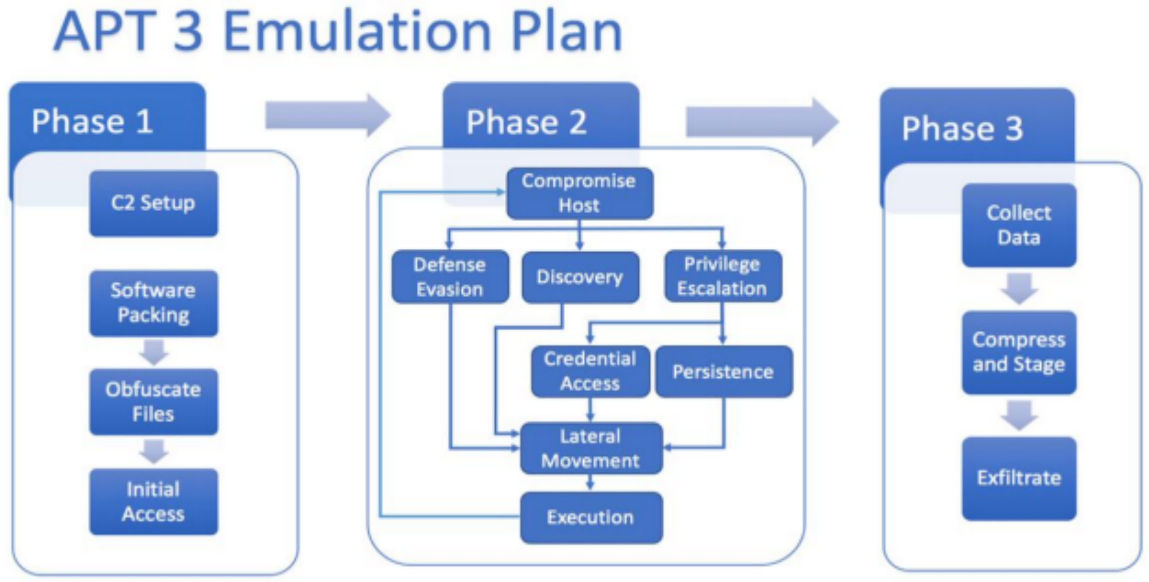
Review / threat management

Methods of control, secure delivery, secure exposure

Control, ownership, responsibility

Adversary Emulation Plans

Delivery	Exploit	Install	C2



Build the test and contain the blast radius.

Think about the threat and your specific conditions for most likely threat actor

https://attack.mitre.org/wiki/Adversary_Emulation_Plans

Why make waves?

The question becomes:

- Why would I want to subject myself to such testing?
- I know at some point I am going to fail, if I do this correctly!

Technically that is the point, knowing where you fail and providing the resilience in technology or process to overcome and mature the processes to deal with attacks.

Resilience engineering

Expectations for improvement

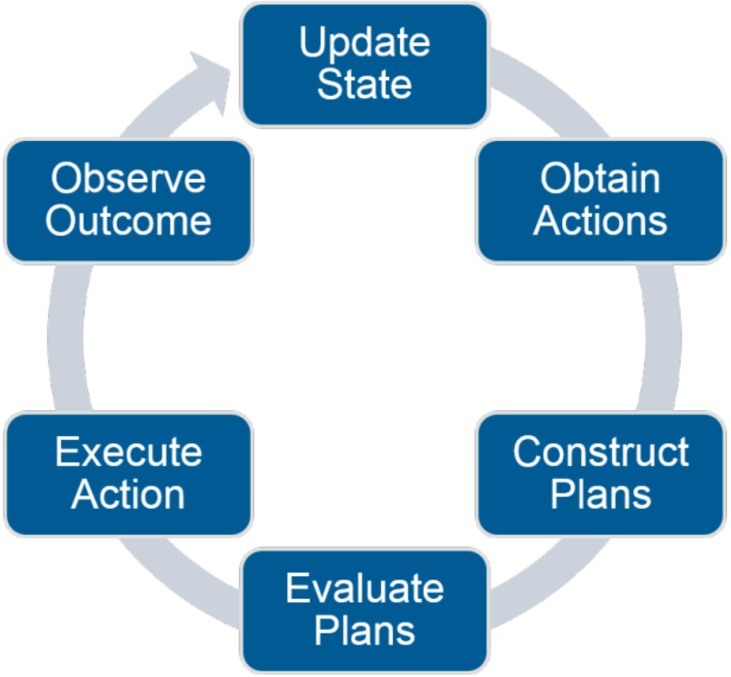
The whole point is to add a level of confidence in the systems, defensive posture and detection of malicious events both from the perspective of resilience testing and adversarial emulation.

The overall risk metric is improvement over time based on differing methods of attack. These attack scenarios should be based on the likely attack vectors used by those who are more likely to be your adversary.

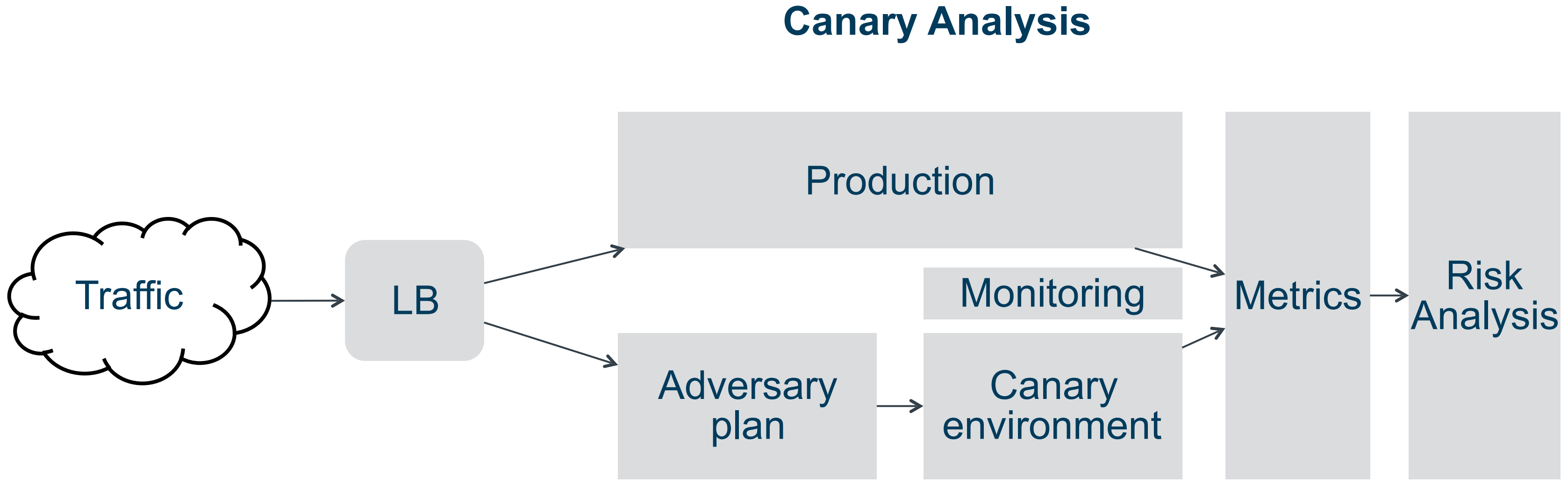
Given the number of Tactics, Techniques, Procedures (TTP) and vulnerabilities identified, the aim is to make this continuous, scheduled and an expectation to provide risk mitigation.

Automated Adversarial Emulation

When it comes to the stage to define the experiment based on the underlying risk hypothesis, we can start to look at what is the testing requirement to confirm the hypothesis, what are the TTP's needed to execute the test and provide feedback to the team



An integrated method



A process of risk management for either testing a new release to prod via the canary instance or testing current production but minimizing the blast radius to just the canary prod environment – real data via the load balancer but not a test across all of the production environment

Risk based approach

Where should I focus this chaos?

This is the risk based approach, do you look at the most critical infrastructure in the organization and start to break it?

This is a work in progress in terms of the overall viability of applying such techniques, start small and make sure the blast radius is enough to give you confidence to move forward.

This however is not a one time DR exercise, this should be baked into continuous monitoring, hence the need for automation, scale and expand as necessary but make sure the underlying program provides value.

Tools

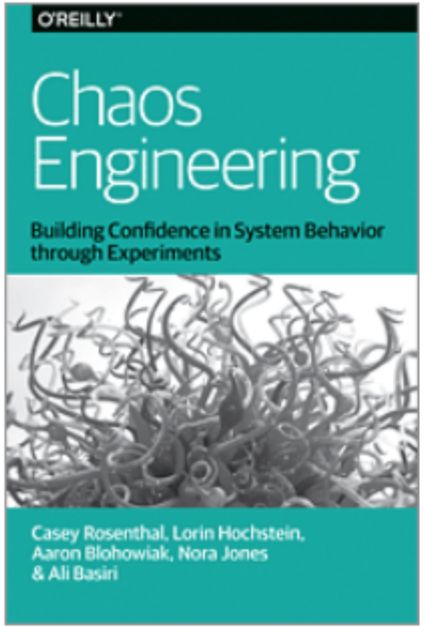
Chaos Tools

- ChaoSlingr - https://github.com/Optum/ChaoSlingr
- Chaos Monkey - https://github.com/netflix/chaosmonkey
- ChAP - Chaos Automation Platform

Automated adversary

- Atomic Red Team - https://github.com/redcanaryco/atomic-red-team
- Red Team Automation - https://github.com/endgameinc/RTA
- Metta - https://github.com/uber-common/metta
- Caldera - https://github.com/mitre/caldera
- APT2 - https://github.com/MooseDojo/apt2
- Invoke-Adversary - https://blogs.technet.microsoft.com/motiba/2018/04/09/invoke-adversary-simulating-adversary-operations/
- APTSimulator - https://github.com/NextronSystems/APTSimulator
- Infection Monkey - https://github.com/guardicore/monkey

References


O'REILLY
Chaos Engineering
Building Confidence in System Behavior through Experiments
Casey Rosenthal, Lorin Hochstein, Aaron Blohowiak, Nora Jones & Ali Basiri

https://www.oreilly.com/webops-perf/free/chaos-engineering.csp

Chaos Engineering diagram of tools and innovators

https://coggle.it/diagram/WiKceGDAwgABrmyv/t/chaos-engineering-companies%2C-people%2C-tools-practices/0a2d4968c94723e48e1256e67df51d0f4217027143924b23517832f53c536e62

In review

- From the perspective of applying such a process, is the test worth the result?
- Are users of your systems all the chaos you will ever need?
- As everything becomes continuous – integration, deployment, monitoring and now continuous chaos – does this provide a new level of confidence or just something added that makes more work?

About CIS

Center for Internet Security®

Thank you