Headquarters U.S. Air Force

Integrity - Service - Excellence

DoD Enterprise DevSecOps Initiative (Software Factory)

Mr. Nicolas Chaillan
Chief Software Officer, U.S. Air Force
Co-Lead, DoD Enterprise DevSecOps Initiative

v4.7 – UNCLASSIFIED

U.S. AIR FORCE

U.S. AIR FORCE

CSO Website – Continuously Updated!

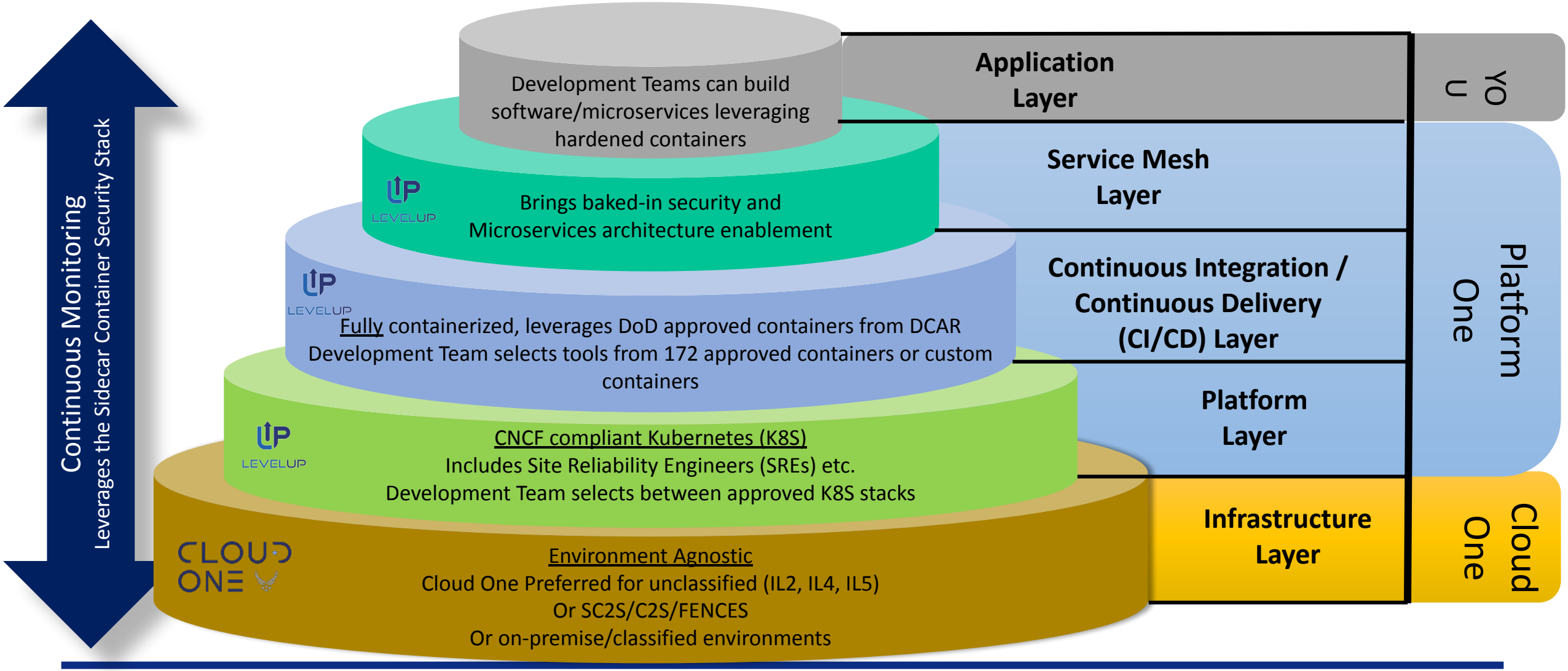
- Want to find information about the DevSecOps initiative and the CSO?
  - https://software.af.mil/
  - **Our latest documents/videos:** https://software.af.mil/dsop/documents/
  - **Our latest training videos from DAU available at:** https://software.af.mil/training/
  - More information about
    - Cloud One
    - Platform One
    - DevSecOps
    - Training including videos selection
    - Software Factories
    - Our Events/News!

What is the DoD Enterprise DevSecOps Initiative?

- Joint Program with OUSD(A&S), DoD CIO, U.S. Air Force, DISA and the Military Services.
- Technology:
  - **Avoid vendor lock-in** at the Infrastructure and Platform Layer by leveraging FOSS with Kubernetes and OCI containers,
  - Creating the DoD Centralized Artifacts Repository (DCAR) of hardened and centrally accredited containers: selecting, certifying, and securing best of breed development tools and software capabilities (over 170+ containers) - https://dccscr.dsop.io/dsop/ and https://dcar.dsop.io
  - **Baked-in Zero Trust Security** with our Sidecar Container Security Stack (SCSS) leveraging behavior detection, zero trust down to the container/function level.
  - Leveraging a Scalable Microservices Architecture with Service Mesh/API Gateway and baked-in security (Istio)
  - Leveraging KNative to avoid lock-in to Cloud provider Serverless stacks
- Bringing **Enterprise IT Capabilities with Cloud One and Platform One** – Cloud and DevSecOps as Managed Services capabilities, on-boarding and support!
- Standardizing metrics and define acceptable thresholds for **DoD-wide continuous Authority to Operate**
- Massive **Scale Training with Self Learning Capabilities** (train over 100K people within a year) and bring state of the art DevSecOps curriculum
- Creating new Agile contracting language to enable and incentivize the use of DevSecOps

U.S. AIR FORCE

Understanding the DevSecOps Layers



U.S. AIR FORCE

LEVELUP CODE WORKS

"Platform One by LevelUP"
The Air Force Software Factory Team

- Merged top talent across U.S. Air Force from various Factories (Kessel Run, SpaceCAMP and UP).
- Helps instantiate DevSecOps CI/CD pipelines / Software Factories within days at various classification levels.
- Manages Software Factories for Development teams so they can focus on building mission applications.
- Provides Basic Ordering Agreement (BOA) DoD-wide DevSecOps contracts for Cloud Service, Talent and Licenses. Enables awards every 15/30 days with bulk discounts.
- Decouples Development Teams from Factory teams with DevSecOps and Site Reliability Engineer (SRE) expertise.
- Partners with Cloud One to provide IL2, 4, 5 and 6 access but also uses C2S/SC2S and various on-premise environments!
- Self-learning and training capabilities to enable teams move to Scrum/Kanban/eXtreme Programming (XP) Agile practices.
- Leverages the DoD hardened containers while avoiding one-size-fits-all architectures.
- Fully compliant with the DoD Enterprise DevSecOps Initiative (DSOP) with DoD-wide reciprocity and an ATO. Leverages Zero Trust model.
- Hardens the 172 DoD enterprise containers (databases, development tools, CI/CD tools, cybersecurity tools etc.).
- Provides Software Enterprise Services with Collaboration tools, Cybersecurity tools, Source code repositories, Artifact repositories, Development tools, DevSecOps as a Service, Chats etc. These services will be MANAGED services on Cloud One.

U.S. AIR FORCE

LEVELUP CODE WORKS

"Platform One by LevelUP" Managed Services "A La Carte"

- Hardened Containers Options
  - Delivery of hardened enterprise containers with accreditation reciprocity (existing containers only).
  - Delivery of custom hardened containers as needed.
- Continuous Integration / Continuous Delivery (CI/CD) Options
  - Delivery of existing hardened Kubernetes/OpenShift/PKS playbooks (full Infrastructure as Code).
  - Delivery of a **turnkey CI/CD pipeline** (Software Factory) with complete « Infrastructure as Code » to instantiate on any environment (development teams picks the tools from the approved hardened containers) on various classified/unclassified environment.
- Training/On-Boarding Options
  - 1-day training Session: introduction to DevSecOps. Overview and understanding of the vision and activities.
  - A 3 day introduction to LevelUP DevSecOps tech stack. Hands on code and User-Centered Design (UCD) to deploy your first demo app to production.
  - A several week full on-boarding, that concludes with an MVP ready for production.
  - A several month full on-boarding, that concludes with your platform team being able to support your own DevSecOps applications for development and production.
  - Customized training options (both at our locations or on your premises).
- Contracting Support Options
  - Ability to leverage the DevSecOps BOAs (Cloud Services, Talent and Licenses).
  - Enable access to DevSecOps engineers/SREs Full-Time-Equivalent (FTEs) (Medics/Counselors) to assist Programs.

U.S. AIR FORCE

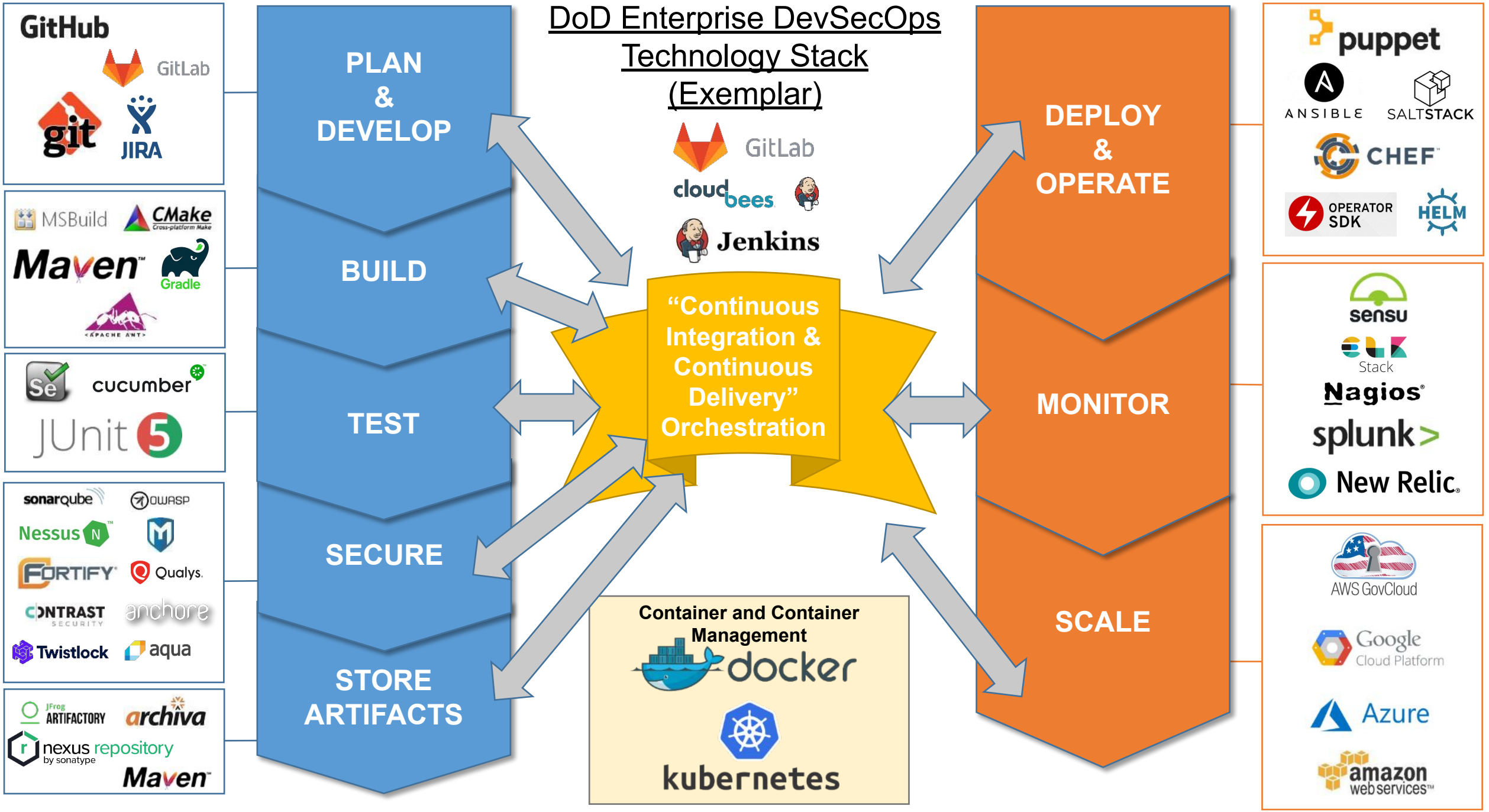
CLOUD ONE

Cloud One

- Air Force Cloud Office with turnkey access to AWS GovCloud and Azure Government at IL2, 4 and 5. IL6 available by December 2019.
- Simple “Pay per use” model with ability to instantiate your own Development and Production VPCs at various Impact Levels within days with full compliance/security and a baked-in ATO.
- Enterprise Solution: we provide the guardrails to the cloud in a standard manner so you can focus on your mission
- Fully Automated: All environmental stand-up is managed by Infrastructure as Code, drastically speeding up deployment, reducing manual work, and human error
- Centralized Identities and Single-Sign-On (SSO): one login across the Cloud stack
- Internet facing Cloud based VPN to connect to IL5 enclaves with a Virtual Internet Access Point (coming within January 2020).
- DevSecOps Focused: secure, mission driven deployments are built into the framework to ensure self-service and seamless deployments. Leverages Zero Trust model.
- Proactive Scaling and System Monitoring: Mission Owners can see all operational metrics and provide rules and alerts to manage each mission their way
- Accreditation Inheritance has been identified in the AF-Cloud One eMASS accounts (AWS & Azure) to include inheritance from the CSP, USAF, DoD and CSSP. All that’s left for the mission is the controls that are unique to them.

DoD Enterprise DevSecOps Architecture

DoD Enterprise DevSecOps Technology Stack (Exemplar)



Key "DevSecOps" Ingredients

- **Abstracted**: to avoid drifts, be agnostic to environment (Cloud/on-premise/classified/disconnected…) and prevent lock-ins with Cloud or Platform layers, we leverage CNCF compliant Kubernetes and OCI compliant containers - open source stacks with U.S eyes on code and continuous scanning,
- **GitOps / Infrastructure as Code (IaC):** no drift, everything is code (including configuration, networking etc.) Instantiate entire stack automatically,
- **Continuous Integration/Continuous Delivery pipeline (CI/CD)**: fully containerized and using Infrastructure as Code (IaC),
- **Hardened Containers**: hardened "Lego blocks" to bring options to development teams (one size fits all lead to shadow IT)
- **Software Testing**: mandated high test coverage,
- **Baked-in Security**: mandated static/dynamic code analysis, container security, bill of material (supply chain risk) etc.
- **Continuous Monitoring:**
  - **Centralized logging and telemetry**,
  - Automated alerting,
  - **Zero trust**, leveraging Service Mesh as Sidecar (part of SCSS), down to the container level,
  - **Behavior detection** (automated prevention),
  - CVE scanning,
- **Chaos engineering**: Dynamically kills/restarts container with moving target defense.

What is GitOps?

- Based on Infrastructure as Code concepts, makes Git the single source of truth of the desired state of your Infrastructure, Platform and Applications.
- Benefits:
  - Everything is code: infrastructure, networking, configuration, sealed secrets etc.
  - Auditability & Compliance
  - Consistent deployments and rollback (no drifts between environment)
  - Configuration Management enforcement
  - Disaster Recovery
  - Baked-in security: Kubernetes clusters **pulls** from Git. CI/CD won't have access to production clusters. Removing human from production environments
  - Declarative manifests and playbooks
- Options:
  - Argo CD, Flux as FOSS. Projects are merging into a single FOSS and be part of CNCF.

"Infrastructure as Code" Benefits

The "Infrastructure as Code" concept is a critical DevSecOps ingredient to ensure that production environments do not drift from development/testing environments. No human should make changes in production environments. Changes should only be made in source code and redeployed by the CI/CD pipeline.

- No drift between environments, whether classified/disconnected/Cloud/on-premise,
- Immutable,
- Replicable,
- Automated,
- No human in production environments: reduces attack surface (disable SSH etc.), insider threat and configuration drifts,
- Everything is code: including playbooks, networking, tests, configuration etc.

U.S. AIR FORCE

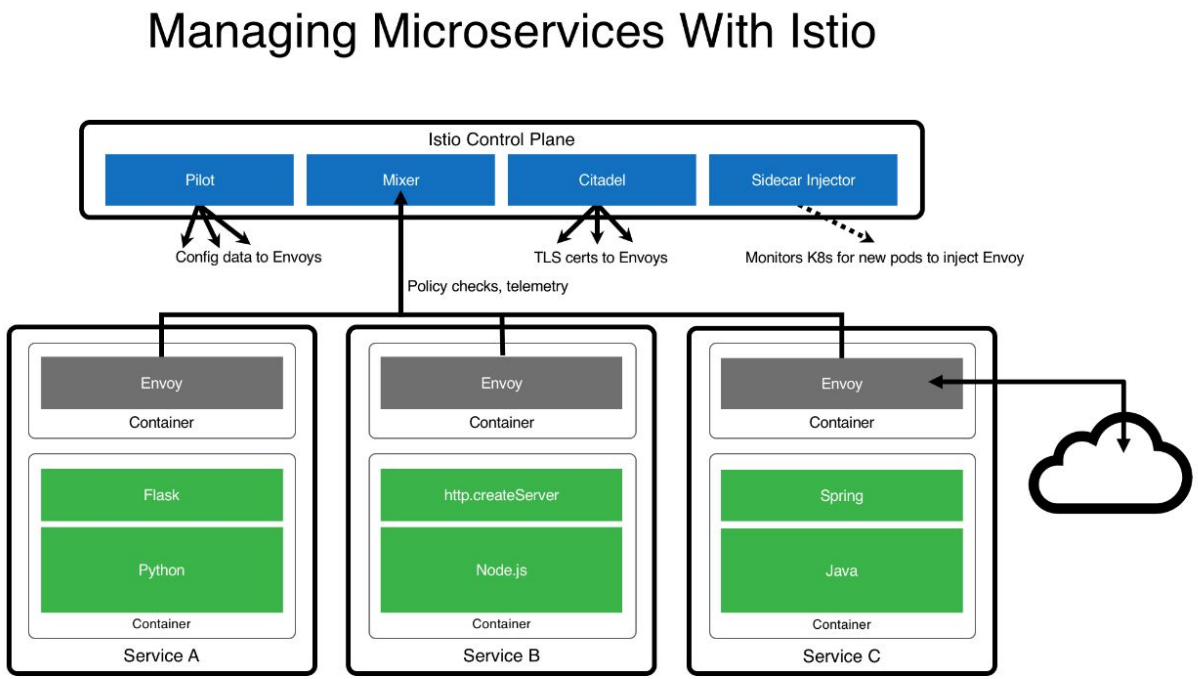
Key "Continuous Security" Ingredients

- **Kubernetes hardening.**
  - Automated injection of Sidecar Container Security Stack (SCSS) into all containers/pods running without manual action.
  - RBAC/SSO/SELinux enabled
  - Compliant with CIS Kubernetes Benchmark, mapped to NIST 800-53
  - Nodes, master, etcd are hardened.
  - Automated backups of cluster and persistent storage!
- **Sidecar Container Security Stack (SCSS):**
  - Automated centralized logging and telemetry with Elasticsearch, Fluentd, Kibana (EFK),
  - Service Mesh (Istio):
    - Baked-in **zero trust model** down to the container level!
      - Strong identities automatically generated using certificates.
      - mTLS tunnel injected across all container communication
    - Whitelist enforcement, Layer 7 load balancer etc
  - Container security: Continuous Scanning, Alerting, CVE scanning, **Behavior detection** both in development and production (Build, Registry, Runtime) with Twistlock (looking into StackRox and Sysdig)
  - Container security and insider threat (custom policies detecting unapproved changes to Dockerfiles) with Anchore
  - Automated STIG compliance with OpenSCAP.

U.S. AIR FORCE

Microservices Architecture (ISTIO)

- Turnkey Service Mesh (ISTIO) architecture
- ISTIO side car proxy, baked-in security, with visibility across containers, by default, without any developer interaction or code change
- Benefits:
  - API Management, service discovery, authentication…
  - Dynamic request routing for A/B testing, gradual rollouts, canary releases, resilience, observability, retries, circuit breakers and fault injection
  - Layer 7 Load balancing
  - Zero Trust model: East/West Traffic Whitelisting, ACL, RBAC…
  - TLS encryption by default, Key management, signing…



U.S. AIR FORCE

DevSecOps Platform Stack (continuously evolving)

U.S. AIR FORCE

DevSecOps Product Stack (1)

Source Repository	API Gateways	Programming Languages	Databases
GitHub Government	Kong	C/C++	SQL Server
GitLab	Azure API	C#/.NET	MySQL
	AWS API	.NET Core	PostgreSQL
Container Management technologies:	Axway	Java	MongoDB
	3Scale	PHP	SQLite
Kubernetes	Apigee	Python	Redis
Openshift	ISTIO (service mesh)	Groovy	Elasticsearch
VMWare Tanzu		Ruby	Oracle
PKS	**Artifacts**	R	etcd
OKD	Artifactory	Rust	Hadoop/HDInsight
Rancher (K8S only)	Nexus	Scala	Cloudera
D2IQ (K8S only)	Maven	Perl	Oracle Big Data
Docker EE (K8S only)	Archiva	Go	Solr
	S3 bucket	Node.JS	Neo4J
Container Packagers:		Swift	Memcached
Helm			Cassandra
Kubernetes Operators			MariaDB
			CouchDB
			InfluxDB (time)

DevSecOps Product Stack (2)

Message bus/Streams
Kafka
Flink
Nats
RabbitMQ
ActiveMQ

Proxy
Oauth2 proxy
nginx ldap auth proxy
openldap
HA Proxy

Visualization
Tableau
Kibana

Logs
Logstash
Splunk Forwarder
Fluentd
Syslogd
Filebeat
rsyslog

Webservers
Apache2
Nginx
IIS
Lighttpd
Tomcat

Docker base images OS:
~~Alpine~~
~~Busybox~~
~~Ubuntu~~
~~Centos~~
~~Debian~~
~~Fedora~~
Universal Base Image

Serverless
Knative

U.S. AIR FORCE

DevSecOps Product Stack (3)

Build
MSBuild
CMake
Maven
Gradle
Apache Ant

Tests suite
Cucumber
J-Unit
Selenium
TestingWhiz
Watir
Sahi
Zephyr
Vagrant
AppVerify
nosetests
SoapUI
LeanFT

Test coverage
JaCoCo
Emma
Cobertura
codecov

CI/CD Orchestration
Jenkins (open source)
CloudBees Jenkins
GitLab

Jenkins plugins
Dozens (Need to verify security).

Configuration Management / Delivery
Puppet
Chef
Ansible
Saltstack

Security
Tenable / Nessus Agents
Fortify
Twistlock
Aqua
SonarQBE
Qualys
StackRox
Aporeto
Snort
OWASP ZAP
Contrast Security
OpenVAS
Metasploit
ThreadFix
pylint
JFrog Xray
OpenSCAP (can check against DISA STIG)
OpenControl for compliance documentation

Security (2)
Snyk
Code Climate
AJAX Spider
Tanaguru (508 compliance)
InSpec
OWASP Dependency-Check
Burp
HBSS
Anchore
Checkmarx
SD Elements
Clair
Docker Bench Security
Notary
Sysdig
Layered Insight
BlackDuck
Nexus IQ/Lifecycle/Firewall

U.S. AIR FORCE

DevSecOps Product Stack (4)

Monitoring
Sensu
EFK (Elasticsearch, Fluentd, Kibana)
Splunk
Nagios
New Relic
Sentry
Promotheus
Grafana
Kiali

Collaboration
Rocket.Chat
Matter.Most
PagerDuty

Plan
Jira
Confluence
Rally
Redmine
Pivotal Tracker

Secrets
Kubernetes Secrets
Vault
Credentials (Jenkins)
CryptoMove

SSO
Keycloak

Documentation
Javadoc
RDoc
Sphinx
Doxygen
Cucumber
phpDocumentator
Pydoc

Performance
Apache AB
Jmeter
LoadRunner

Legacy to DevSecOps => Strangler Pattern

- Martin Fowler describes the Strangler Application:
  - *One of the natural wonders of this area are the huge strangler vines. They seed in the upper branches of a fig tree and gradually work their way down the tree until they root in the soil. Over many years they grow into fantastic and beautiful shapes, meanwhile strangling and killing the tree that was their host.*
- To get there, the following steps were followed:
  - First, add a proxy, which sits between the legacy application and the user. Initially, this proxy doesn't do anything but pass all traffic, unmodified, to the application.
  - Then, add new service (with its own database(s) and other supporting infrastructure) and link it to the proxy. Implement the first new page in this service. Then allow the proxy to serve traffic to that page (see below).
  - Add more pages, more functionality and potentially more services. Open up the proxy to the new pages and services. Repeat until all required functionality is handled by the new stack.
  - The monolith no longer serves traffic and can be switched off.
- Learn more: https://www.ibm.com/developerworks/cloud/library/cl-strangler-application-pattern-microservices-apps-trs/index.html and https://www.michielrook.nl/2016/11/strangler-pattern-practice/

U.S. AIR FORCE

Training Options

- **Our latest training videos from DAU available at: https://software.af.mil/training/**
- Check out our curated YouTube videos on Kafka, Kubernetes, Service Mesh, Microservices, Cloud etc. at https://software.af.mil/training/
- **NEW:** Federal employees/Military personnel (limited number of seats, free of charge): reach out to us at usaf.cso@mail.mil if you want to pilot the access to the **O'Reilly Online Learning Platform** (all O'Reilly content + virtualized K8S env)!
- Platform One Training/On-Boarding Options:
  - 1-day training Session: introduction to DevSecOps. Overview and understanding of the vision and activities
  - A 3-day introduction to LevelUP DevSecOps tech stack. Hands on code and User-Centered Design (UCD) to deploy your first demo app to production
  - A several week full on-boarding, that concludes with an MVP ready for production
  - A several month full on-boarding, that concludes with your platform team being able to support your own DevSecOps applications for development and production
  - Customized training options (both at our locations or on your premises)
- Follow the CNCF channel: https://www.youtube.com/channel/UCvqbFHwN-nwalWPjPUKpvTA

U.S. AIR FORCE

Thank You!

Nicolas Chaillan
Chief Software Officer, U.S. Air Force
usaf.cso@mail.mil

Backup Slides

U.S. AIR FORCE

Nicolas M. Chaillan

Chief Software Officer

- Nicolas M. Chaillan is the Chief Software Officer at the U.S. Air Force and the Co-Lead for the DoD Enterprise DevSecOps Initiative.
- He is the former Special Advisor for Cloud Security and DevSecOps at OSD, A&S.
- He was the Special Advisor for Cybersecurity at the Department of Homeland Security and the Chief Architect for Cyber.gov, the new robust, innovative and holistic .Gov cyber security architecture for all .gov agencies.
- Chaillan is a technology entrepreneur, software developer, cyber expert and inventor. He is recognized as one of France's youngest entrepreneurs after founding his first company at 15 years of age.
- With 19 years of international tech, entrepreneurial and management experience, Chaillan is the founder of more than 12 companies, including AFTER-MOUSE.COM, Prevent-Breach, anyGuest.com, and more.
- Over the last eight years alone, he has created and sold over 180 innovative software products to 40 Fortune 500 companies.
- Chaillan is recognized as a pioneer of the computer language PHP.

2018
OFFICIAL MEMBER
Forbes
Technology Council

U.S. AIR FORCE

Questions about DCCSCR/DCAR?

- Containers accredited in the DCCSCR/DCAR repository have DoD-wide reciprocity across classifications.
  - Source code repo: DCCSCR: https://dccscr.dsop.io/dsop
  - Source code repo: DCCSCR Infrastructure as Code (IaC): https://dccscr.dsop.io/levelup-automation/aws-infrastructure
- DCAR (Container binaries): https://dcar.dsop.io
- Programs can contribute containers that have enterprise benefits to DCCSCR/DCAR and our team will accredit them DoD-wide and maintain them.
- If you need to accredit/harden custom containers Platform One can do this as a “managed service”, Pay per use model.
- We are building a container which automatically download container updates into your K8S cluster, checks signatures and pushes them to your local registry (agnostic to your artifact repo)
- Questions? Email usaf.cso@mail.mil

“Cloud One” vs “Platform One”

- Cloud One:
  - Centralized team to provide Cloud Infrastructure with baked-in security to DoD programs. Think of it as the Infrastructure team with baked-in security, CSSP and Authority to Operate (ATO).
  - Only contact Cloud One if you **only** need Cloud compute/storage, if you need a DevSecOps stack (on Cloud One or not), contact « Platform One by LevelUP »
  - Point of Contact: DOLCE, JOSEPH G Maj USAF - joseph.dolce@us.af.mil; Watson, Todd M Lt Col USAF todd.watson@us.af.mil
- Platform One:
  - Centralized team to provide DevSecOps/Software Factory with baked-in security to DoD Programs. Think of it as the Platform Team with the ability to deploy a DevSecOps (Kubernetes compliant) Platform and CI/CD pipeline with a Continuous ATO (c-ATO). You select from accredited tools to accelerate your ability to focus on delivering mission capabilities.
  - **Contact Platform One if you need both Cloud and DevSecOps capabilities!**
  - Point of Contact: Slaughter, Rob Maj USAF - rob.slaughter@afwerx.af.mil; Bryan, Austen R Capt USAF - austen.bryan.1@us.af.mil; Lopez De Uralde, Richard A Lt Col USAF - richard.lopezdeuralde@us.af.mil.

Questions about the DevSecOps Basic Ordering Agreements (BOAs)?

- Covers Cloud Services, Talent and Licenses so DoD programs can get everything they need for a DevSecOps environment, completely turnkey.
- Pay per use models.
- Not selected yet? We will have quarterly on-boarding events for new vendors/award opportunities!
- Aims to award each order within 15 days (!!!)
- Available to the entire Department of Defense
- Point of Contact:
  - Paul, Christopher C 2d LT USAF AFLCMC (USA) christopher.paul.3@us.af.mil
  - Slaughter, Robert C Maj USAF (USA) robert.slaughter.3@us.af.mil
  - Wyler, Victoria R Capt USAF SAF-AQ (USA) victoria.r.wyler.mil@mail.mil

U.S. AIR FORCE

Contribute your containers or get your COTS/FOSS containers accredited!

- **Containers are the easiest way to get accredited DoD-wide across multiple classifications today.**
- Containers accredited in the DCCSCR/DCAR repository have **DoD-wide reciprocity across classifications**.
- **Check out the vendor on-boarding guide at:** https://dccscr.dsop.io/dsop/dccscr/tree/master/contributor-onboarding
- By being compliant with the DoD Enterprise DevSecOps Container Hardening guide (last version at https://software.af.mil/dsop/documents/), you can have your containers (FOSS/COTS/GOTS) accredited for DoD use.
- Recommend using the hardened STIG UBI7/8 images (Universal Base Image which is lightweight RHEL but doesn't need a license) from the DCAR repo as your base image so you don't have to STIG your container base OS: https://dcar.dsop.io. **Use the binary signed version on DCAR, do not rebuild it.**
- Key aspects:
  - Your container must be able to build offline. If you have dependencies, provide them with an automated script that will download new updates of those dependencies. ALL DEPENDENCIES must be included
  - Container must be able to be built offline, no downloads in Dockerfile!
  - Dockerfiles must be provided and be able to be rebuilt. If you have a different base OS (not UBI), it must be STIGed.

U.S. AIR FORCE

Why Kubernetes / Containers?

- One of the most critical aspect of the DevSecOps initiative is to ensure we **avoid any vendor lock-in** so the DoD mandated:
  - **Open Container Initiative (OCI) containers** (no lock-in to containers/container runtimes/builders)
  - **Cloud Native Computing Foundation (CNCF) Kubernetes compliant cluster** for container orchestration, no lock-in to orchestration options/networking/storage APIs.
- SaaS vs COTS/FOSS containers:
  - SaaS requires FedRAMP certification and will limit you to unclassified environments (mostly IL5 for FedRAMP high) which doesn't satisfy most needs for DoD programs. Often takes up to 1 year.
  - COTS/FOSS as containers: can be sold as a managed service deployed in DoD cloud environments (including classified clouds) on Kubernetes and can be accredited at multiple classification levels, within weeks, by following the container hardening guide and vendor on-boarding process!
- Containers are **immutable** and will allow the DoD to centrally accredit and harden containers (FOSS, COTS, GOTS) (think of a true gold disk concept but that actually scale and works).
- Continuous Monitoring is a critical piece of our Continuous ATO model and the Sidecar Container Security Stack (SCSS) brings those capabilities with Behavior, Zero Trust and CVE scanning.
- Kubernetes will provide:
  - **Resiliency**: Self-healing so containers that crash can automatically be restarted,
  - **Baked-in security**: thanks to **automatic injection** of our Sidecar Container Security Stack (SCSS) to any K8S cluster with Zero Trust,
  - **Adaptability**: containers are "Lego" blocks and can be swapped with no downtime thanks to load balancing and modern routing (A/B testing, canary release etc.),
  - **Automation**: thanks to our Infrastructure as Code (IaC) and GitOps model,
  - **Auto-scaling**: if load requires more of the same container, K8S will automatically scale based on compute/memory needs,
  - **Abstraction layer**: ensure we don't get locked-in to Cloud APIs or to a specific platform as K8S is managed by CNCF and dozens of products are compliant with its requirements.

U.S. AIR FORCE

Problem Statement

- **What is DevSecOps?**
  - The software automated tools, services, and standards that enable programs to develop, secure, deploy, and operate applications in a secure, flexible and interoperable fashion.
- **Why should I care?**
  - Software and cybersecurity pervades all aspects of DoD's mission (from business systems to weapons systems to Artificial Intelligence to cybersecurity to space) - establishing DevSecOps capabilities will:
    - Deliver applications rapidly and in a secure manner, increasing the warfighters competitive advantage
    - Bake-in and enforce cybersecurity functions and policy from inception through operations
    - Enhance enterprise visibility of development activities and reduce accreditation timelines
    - Ensure seamless application portability across enterprise, Cloud and disconnected, intermittent and classified environments
    - Drive DoD transformation to Agile and Lean Software Development and Delivery
  - Leveraging industry acquisition best practices combined with centralized contract vehicle for DevSecOps tools and services will enable rapid prototyping, real-time deployments and scalability.
  - We cannot be left behind: China, Russia and North Korea are already massively implementing DevOps.

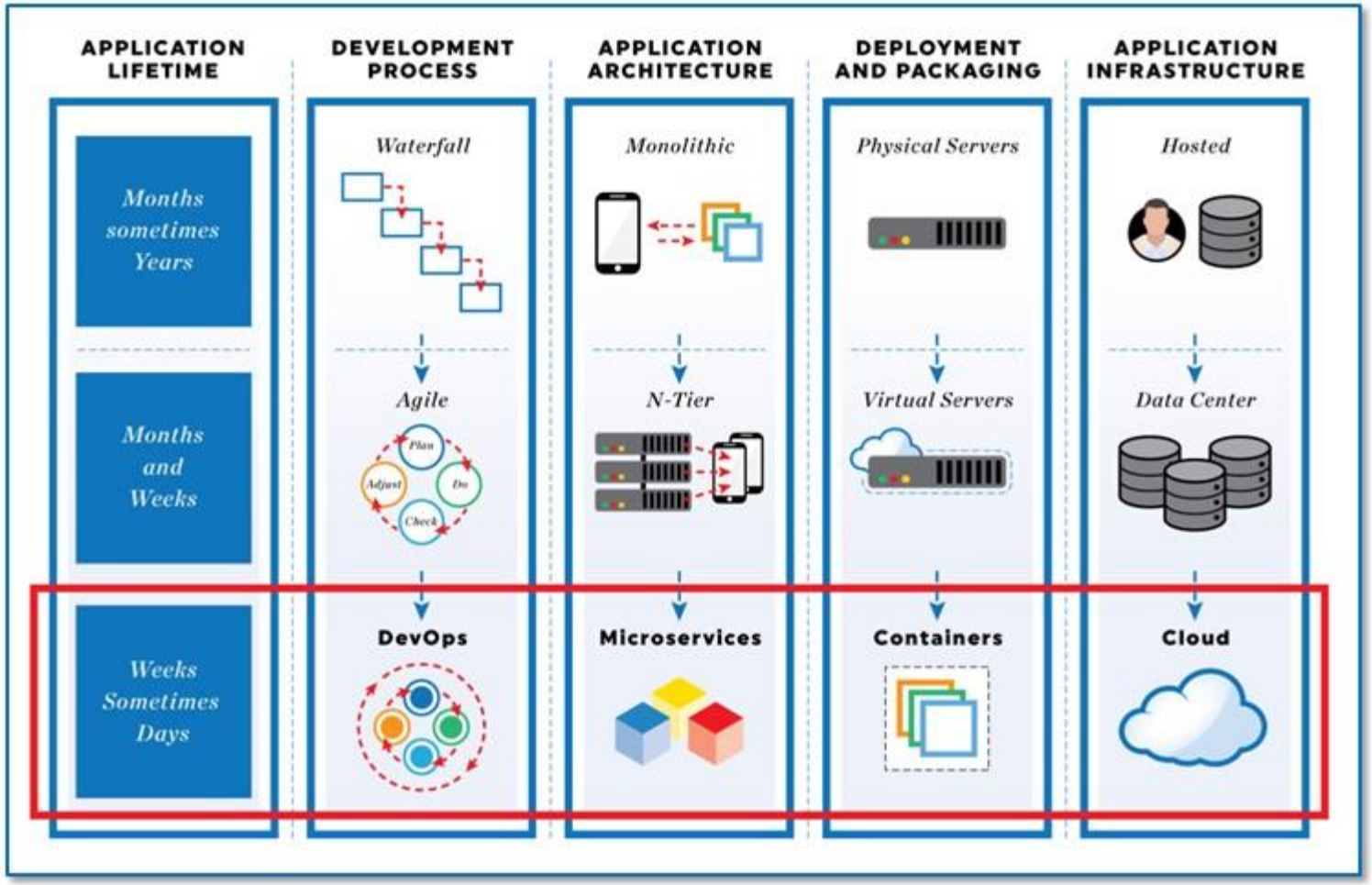
U.S. AIR FORCE

Value for DoD Programs

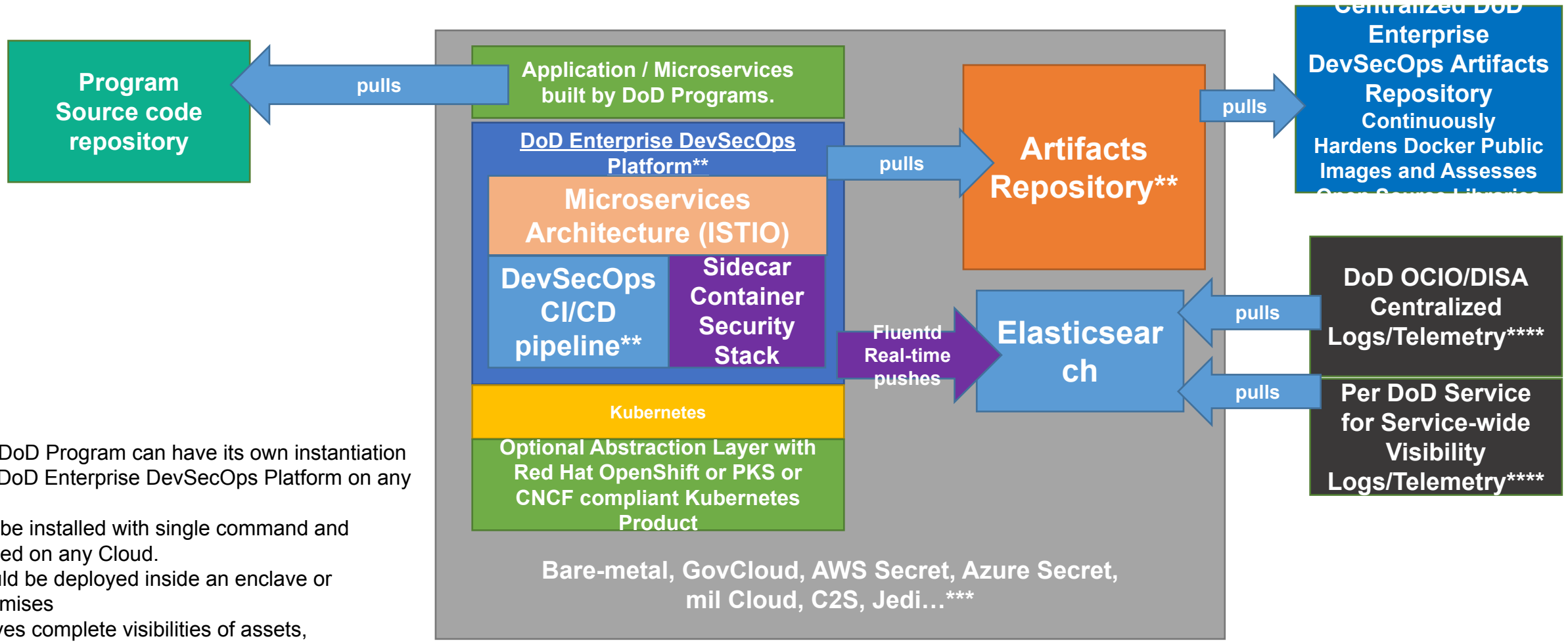
- Enables any DoD Program across DoD Services deploy a DoD hardened Software Factory, on their existing or new environments (including classified, disconnected and Clouds), within days instead of a year. Tremendous cost and time savings.
- Multiple DevSecOps pipelines are available with various options (no one-size-fits-all)
- Enables rapid prototyping (in days and not months or years) for any Business, C4ISR and Weapons system. Deployment in PRODUCTION!
- Enables learning and continuous feedback from actual end-users (warfighters).
- Enables **bug and security fixes in minutes** instead of weeks/months.
- Enables automated testing and security.
- Enables **continuous Authorization to Operate (c-ATO)** process. Authorize ONCE, use MANY times!
- Brings a holistic and baked-in cybersecurity stack, gaining complete visibility of all assets, software security state and infrastructure as code.

U.S. AIR FORCE

From Waterfall to DevSecOps



DoD Enterprise DevSecOps Architecture*



*each DoD Program can have its own instantiation of the DoD Enterprise DevSecOps Platform on any Cloud.
** can be installed with single command and deployed on any Cloud.
*** could be deployed inside an enclave or on-premises
**** gives complete visibilities of assets, security/vulnerability state etc. can be integrated to existing cybersecurity shared services.

Self-Learning (1)

- Recommended Videos (Part 1)
  - Watch our playlists, available at different expertise levels and continuously augmented!
  - Kafka / KSQL (message bus, pub/sub, event driven):
    - Beginners: https://www.youtube.com/playlist?list=PLSIv_F9TtLIzz0zt03Ludtid7icrXBesg
    - Intermediate: https://www.youtube.com/playlist?list=PLSIv_F9TtLIxxXX0oCzt7IaO6mD61UIQw
    - Advanced: N/A
  - Kubernetes
    - Beginners: https://www.youtube.com/playlist?list=PLSIv_F9TtLIydFzQzkYYDdQK7k5cEKubQ
    - Intermediate: https://www.youtube.com/playlist?list=PLSIv_F9TtLIx8dSFH_jFLK40Tt7KUXTN_
    - Advanced: https://www.youtube.com/playlist?list=PLSIv_F9TtLIytdAJiVqbHucWOvn5LrTNW

Self-Learning (2)

- Recommended Videos (Part 2)
  - Watch our playlists, available at different expertise levels and continuously augmented!
  - Service Mesh
    - Beginners: https://www.youtube.com/playlist?list=PLSIv_F9TtLIxtC4rDIMQ8QiG5UBCjz7VH
    - Intermediate: https://www.youtube.com/playlist?list=PLSIv_F9TtLIwWK_Y_Cas8Nyw-DsdbH6vl
    - Advanced: https://www.youtube.com/playlist?list=PLSIv_F9TtLIx8VW2MFONMRwS_-2rSJwdn
  - Microservices
    - Beginners: https://www.youtube.com/playlist?list=PLSIv_F9TtLIz_U2_RaONTGYLkz0lh-A_L
    - Intermediate: https://www.youtube.com/playlist?list=PLSIv_F9TtLIxqjuAXxoRMjvspaEE8L2cB
    - Advanced: https://www.youtube.com/playlist?list=PLSIv_F9TtLIw4CF4F4t3gVV3j0512CMsu

U.S. AIR FORCE

Self-Learning (3)

- Recommended Books
  - A Seat at the Table – by Mark Schwartz (former CIO of USCIS, leader in Agile)

  This book is highly recommended for ALL leadership as it is not technical but focused on the challenges around business, procurement and how leadership can enable DevOps across the organization and remove impediments.
  - The Phoenix Project – by the founders of DevOps
  - The DevOps Handbook – by Gene Kim, Patrick Debois.

For those who drive to work like me (for hours), please note that these books are available as Audiobooks.

U.S. AIR FORCE

Current Challenges

U.S. AIR FORCE

Key Procurement Recommendations for Successful SW Acquisition

- Mandate the use of Agile methodologies including for the creation of RFPs by using user stories instead of pre-defined unverified technology claims or expectation. Clearly established success metrics should be defined based on the targeted end-goals. It should allow for rapid change of scope based on learnings as long as it meets those end goals.
- Incentivize the use of Microservices/Smaller code base instead of building giant monolith that cannot be supported or understood. The use of the 2 pizza rule by Amazon is a great example to size teams. https://www.theguardian.com/technology/2018/apr/24/the-two-pizza-rule-and-the-secret-of-amazons-success
- Incentivize the use of DevSecOps with pre-defined metrics for:
  - Test coverage %
  - Security scans (fuzzing, pen-test, static/dynamic code analysis)
  - Documentation
- Incentivize the use of Containers whenever possible to allow for scale and rapid deployment to any Cloud or on premise servers.
- Incentivize the use of Container Management solution (Kubernetes) whenever possible to allow for rapid deployment, side-car container security and rolling updates.

U.S. AIR FORCE

DevOps Challenges for Leadership

- Leadership can enable the adoption of DevSecOps by bringing its ENTIRE stack as a platform and by leveraging DevSecOps solutions.
- It is imperative not to “select” a limited option of tools. The key of microservices and containers is to be able to use the best solution to achieve the outcome desired. We should not limit options to the extend possible.
- Need to establish baseline requirements / thresholds for cybersecurity, test coverage, test results, documentation. This shouldn’t be reinvented per office but global to DoD as a standard practice to facilitate adoption.
- Once those baseline requirements are set, OCIO can provide CI/CD Platform’ stack with embedded security as a side-car container and provide pre-ATOs for systems using the Platform stack and automatically integrating the OCIO security baseline requirements.
- Bringing the entire stack as a Platform as a Service is key to avoid that each office reinvents the wheel and builds their own baseline requirements.
- Understand that Failing is a GOOD thing! It is part of learning. It allows us to understand what works and what doesn’t - AS LONG AS we leverage rapid prototyping, which allows for QUICK failure and mostly painless. It also allows us to reprioritize rapidly and leverage learnings.

U.S. AIR FORCE

DevOps Challenges for Acquisition Office

- DevOps is a complete disruption of the traditional Acquisition model. We need to leverage Sec 873/874 of the 2018 NDAA (check out https://cdnapisec.kaltura.com/index.php/extwidget/preview/partner_id/2203981/uiconf_id/38241181/entry_id/1_gib6brbc/embed/dynamic).
- No more complex RFPs with long planning phases with deliverables and milestones and fixed budgets. Keep in mind that when we write RFPs, we assume we know what we need and know exactly what the solution is. This doesn't allow for us to learn and adopt new ideas along the way. This is the main cause of current failures. The world/technology evolves, what we know evolves… We must be able to adapt continuously to guarantee success.
- This is NOT scope creep but proper agile scope management.
- RFPs should NOT define precise requirements with pre-defined technologies but focus on establishing mission outcomes and precise metrics to prove success of those end-goals.
- There is no "beginning" or "end" of a project, the project will continuously evolve based on mission needs. Yes, a project might be terminated but the idea is continuous evolution and development.
- The traditional accounting methods - where you have the R&D, development and deployment in production followed by maintenance/support phases - don't apply anymore. We CONTINUOUSLY develop, deploy and learn. There are multiple releases per day.
- New procurement tools must enable continuous development and incentivize the use of containers, microservices and Agile methods.
- Most of the DevOps tools are opensource and the only costs are Cloud hosting/computing/storage. Some members are worried about costs of licenses and think about consolidation for cost saving - this is a non-issue. We pay for what we use as a IaaS/PaaS/SaaS model.

U.S. AIR FORCE

But…How Do We Trust/Monitor?

- Most people treat IT as a contractor/vendor relationship that provides IT services and most don't trust that IT developers can/want to deliver on-time and efficiently. As long as we treat IT as a vendor, this relationship is bound to fail.
- Thanks to DevOps and Agile, we can continuously monitor everything. There is no need for phases and milestones, we can continuously check the status of the Product Backlog and the continuously delivered product in testing/staging environment or even in production.
- We can immediately verify that our assumptions are correct and learn faster.
- This provides BETTER and more TANGIBLE and qualified monitoring as we can leverage metrics and real-life feedback.
- Thanks to Agile and DevSecOps principles, we collect real-time telemetry to gain continuous visibility on our deployments.

U.S. AIR FORCE

But…How to Secure at Scale?

- Mandating the use of containers and Kubernetes whenever possible will enable us to gain complete visibility of the container's stack, thanks to the use of side-car containers.
- A side-car container runs along side of all container pods, ensuring that no matter the size of the clusters, a security container will be present to get visibility, enforce security best practices and compliance requirements.
- By leveraging Agile methodologies, integrating new DevSecOps pipeline solutions and defining a security baseline per product to ensure proper compliance and security.
- Each new product in the stack will be containerized in a Hardened Docker base image to enable any development team to reuse it without having to reinventing the wheel (similar to the gold images concept except this is using modern container technology).

Waterfall

- The waterfall model is a relatively linear sequential design approach for certain areas of engineering design. In software development, it tends to be among the less iterative and flexible approaches, as progress flows in largely one direction ("downwards" like a waterfall) through the phases of conception, initiation, analysis, design, construction, testing, deployment and maintenance.
- The waterfall development model originated in the manufacturing and construction industries; where the highly structured physical environments meant that design changes became prohibitively expensive much sooner in the development process. When first adopted for software development, there were no recognized alternatives for knowledge-based creative work.
- Waterfall basically is a sequential model where software development is segregated into a sequence of pre -defined phases – including feasibility, planning, design, build, test, production, and support. On the other hand, Agile development methodology follows a linear sequential approach while providing flexibility for changing project requirements, as they occur.

U.S. AIR FORCE

Agile Terminologies

Agile

- Agile software development describes an approach to software development under which requirements and solutions evolve through the collaborative effort of self-organizing and cross-functional teams and their customer(s)/end user(s). It advocates adaptive planning, evolutionary development, early delivery, and continual improvement, and it encourages rapid and flexible response to change.
- The term *agile* (sometimes written *Agile*) was popularized, in this context, by the *Manifesto for Agile Software Development*. The values and principles espoused in this manifesto were derived from and underpin a broad range of software development frameworks, including Scrum and Kanban.
- There is significant evidence that adopting agile practices and values improves the agility of software professionals, teams and organizations;
- Agile enables you to learn directly for end users and implement changes while doing so.
- Extreme programming (XP) is a software development methodology which improves software quality and responsiveness to changing customer requirements. It advocates frequent "releases" in short development cycles, which is intended to improve productivity and introduce checkpoints at which new customer requirements can be adopted. It includes: programming in pairs or doing extensive code review, unit testing of all code, avoiding programming of features until they are needed, flat management structure, code simplicity and clarity, expecting changes in the requirements as time passes and the problem is better understood, and frequent communication with the customer and among programmers.

U.S. AIR FORCE

Agile Scrum Sprints

- Scrum Sprint is a repeatable fixed time-box during which a "Done" product of the highest possible value is created. It also has a maximum duration. Usually, a Sprint lasts for one month or less.
- Usually, daily meetings are held to discuss the progress of the project undertaken and any difficulty faced by any team member of the team while implementing the project. The outcome of the sprint is a deliverable, albeit with some increments. The scrum is used for projects like Web Technology or development of a product for the new market, i.e. the product with lots of requirement or fast-changing requirement.

Agile User Story

User stories are one of the primary development artifacts for Scrum and Extreme Programming (XP) project teams. A user story is a very high-level definition of a requirement, containing just enough information so that the developers can produce a reasonable estimate of the effort to implement it.

As a [type of user], I want [some goal, function] so that [some reason].

As a HR Manager, I need to view a candidate's status so that I can manage their application process throughout the recruiting phases.

Learn more: http://www.agilemodeling.com/artifacts/userStory.htm

Agile Product Owner

Product Owner is responsible for product's current state of development and for maximizing the product's value. Product Owner can be one person, even if he represents a committee. His jobs includes:

- Maintaining items in Product Backlog.
- Assigning order to items in Backlog.
- Ensuring that items in Product Backlog are clear to the Development Team.

It is important to ensure that the product owners include end users that understand the end goals for the product.

U.S. AIR FORCE

Agile Product Backlog

The product backlog comprises an ordered list of product requirements that a scrum team maintains for a product. The format of product backlog items varies, common formats include user stories, use cases, or any other requirements format the team finds useful. These will define features, bug fixes, non-functional requirements, etc.—whatever must be done to successfully deliver a viable product. The product owner prioritizes product backlog items (PBIs) based on considerations such as risk, business value, dependencies, size, and date needed.

https://en.wikipedia.org/wiki/Scrum_(software_development)#Product_backlog

More Agile Terminologies (1)

- **Development Team**

The development team is responsible for the implementation of the articles in Sprint Backlog . Although several members of the development team may specialize in different areas, the development team as a whole is responsible for the development of functionality.

- **Sprint Backlog**

Sprint Backlog refers to a subset of Product Backlog that is selected for a Sprint along with its delivery plan. Based on the items in the Sprint Backlog, Development Team decides how they will create a "Done" product.

U.S. AIR FORCE

More Agile Terminologies (2)

- Daily Scrum
  - Daily Scrum is a fixed time, fixed place event that allows Development Team to synchronize and plan work for the next 24 hours based on the amount of work done since the last Daily Scrum. During Daily Scrum, Development Team members explain:
    - What did I do yesterday that helped towards Sprint Goal?
    - What am I going to do today towards my Sprint Goal?
    - What Impediments I see towards accomplishing my Sprint Goal?
  - Daily Scrum usually lasts for 15 minutes, but can be followed by other meetings for detailed discussions.
- Sprint Review & Retrospective
  - Sprint Review is scheduled after the sprint ends to inspect the amount of work done and adapt the Product Backlog if necessary. Similarly, Retrospective is used to analyze what went right in the Sprint and what could be improved upon. This Retrospective feedback helps improve the process in Sprints to follow.

Agile vs Waterfall (1)

Top 10 differences between Agile and Waterfall Methodology:

1. **The software development process is divided into different phases in the Waterfall model while Agile methodology segregates the project development lifecycle into sprints**
2. **Waterfall is a structured software development methodology, and often times can be quite rigid, whereas the Agile methodology is known for its flexibility**
3. **According to the Waterfall model, software development is to be completed as one single project, which is then divided into different phases, with each phase appearing only once during the SDLC. However, the Agile methodology can be considered as a collection of many different projects, which are nothing but the iterations of the different phases focusing on improving the overall software quality with feedbacks from users or the QA team**
4. **If you want to use the Waterfall model for software development, then you have to be clear with all the development requirements beforehand as there is no scope of changing the requirements once the project development starts. The Agile methodology, on the other hand, is quite flexible, and allows for changes to be made in the project development requirements even after the initial planning has been completed**

Agile vs Waterfall (2)

5. All the project development phases such as designing, development, testing, etc. are completed once in the Waterfall model while as part of the Agile methodology, they follow an iterative development approach. As a result, planning, development, prototyping and other software development phases can appear more than once during the entire SDLC
6. One of the major differences between Agile and Waterfall development methodology is their individual approach towards quality and testing. In the Waterfall model, the “Testing” phase comes after the “Build” phase, but, in the Agile methodology, testing is typically performed concurrently with programming or at least in the same iteration as programming
7. While Waterfall methodology is an internal process and does not require the participation of customers, the Agile software development approach focuses on customer satisfaction and thus, involves the participation of customers throughout the development phase.

Agile vs Waterfall (3)

8. The Waterfall model can be regarded as a stringently sequential process, however, the Agile methodology is a highly collaborative software development process, thereby leading to better team input and faster problem solving
9. The Waterfall model is best suited for projects which have clearly defined requirements and in which change is not expected at all, while Agile development supports a process in which the requirements are expected to change and evolve. Thus, if you are planning to develop a software that would require frequent overhauls and has to keep up with the technology landscape and customer requirements, Agile is the best approach to follow
10. The Waterfall model exhibits a project mindset and lays its focus strictly on the completion of project development, while Agile introduces a product mindset that focuses on ensuring that the developed product satisfies its end customers, and changes itself as the requisites of customers change

DevOps Terminologies

DevOps

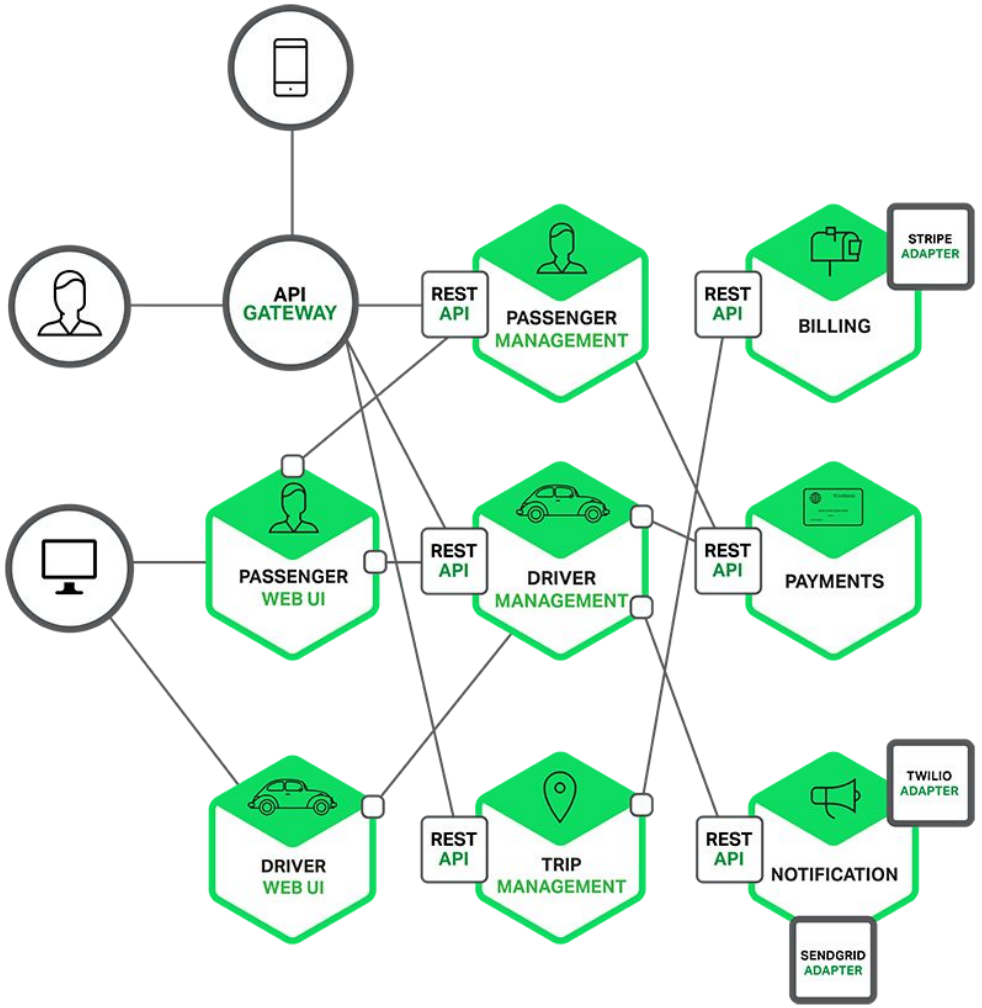
- DevOps is a software engineering culture and practice that aims at unifying software development (Dev) and software operation (Ops). The main characteristic of the DevOps movement is to strongly advocate automation and monitoring at all steps of software construction, from integration, testing, releasing to deployment and infrastructure management. DevOps aims at shorter development cycles, increased deployment frequency, and more dependable releases, in close alignment with business objectives.
- DevOps is NOT ENOUGH! DevSecOps is what must be implemented with the cybersecurity stack built-in into the DevOps pipeline.

Microservices

- A 'microservice' is a software development technique—a variant of the service-oriented architecture (SOA) architectural style that structures an application as a collection of loosely coupled services.
- In a microservices architecture, services are fine-grained and the protocols are lightweight. The benefit of decomposing an application into different smaller services is that it improves modularity and makes the application easier to understand, develop, test, and more resilient to architecture erosion. It parallelizes development by enabling small autonomous teams to develop, deploy and scale their respective services independently.
- It also allows the architecture of an individual service to emerge through continuous refactoring. Microservices-based architectures enable continuous delivery and deployment.

U.S. AIR FORCE

Example of Microservices Architecture



Containers (Docker…)

No we are not talking about shipping containers!

- **Platform independence: Build it once, run it anywhere**
- **Resource efficiency and density**
- **Effective isolation and resource sharing**
- **Speed: Start, create, replicate or destroy containers in seconds**
- **Immense and smooth scaling**
- **Operational simplicity**
- **Improved developer productivity and development pipeline (thanks to DevOps)**

Learn more about containers:
https://medium.freecodecamp.org/a-beginner-friendly-introduction-to-containers-vms-and-docker-79a9e3e119b

CI/CD

Continuous Integration and Continuous Delivery are key parts of the DevOps pipeline.

- **Continuous integration (CI) is the practice of consolidating all new source code into a shared version control server such as GitHub, several times a day.**
- **Continuous delivery (CD) is used to deliver (release) software in short cycles, ensuring that the software can be reliably released at any time. It aims at building, testing, and releasing software with complete automation. The approach helps reduce the cost, time, and risk of delivering changes by allowing for more incremental updates to applications in production. A straightforward and repeatable deployment process is important for continuous delivery.**
- **Extreme programming (XP) adopted the concept of CI and did advocate integrating more than once per day – perhaps as many as tens of times per day.**

Agile vs DevSecOps

DevSecOps is the next evolution of agile and builds on the agile principles by adding the following:

- Leverages Containers and Microservices concepts.
- Leverages Cloud deployment for scalability and prototyping.
- Continuous Integration/Continuous Delivery to rapidly prototype, test and deploy.
- Leverage A/B testing and canary deployment for rapid feedback loops.
- Embed security in the pipeline instead of an after-thought.

Automated Testing

Automated testing is a key part of DevSecOps. It is enabled by multiple tools that measure both test code coverage and test results. They are fully automated and do not require human action.

It also enables new concepts like pair programming and peer code review.

Agile brings several new models for creating the right tests:

Test-driven development (TDD) is a software development process that relies on very short development cycles: requirements are turned into very specific test cases first, then the software is built to pass the tests.

Acceptance test–driven development (ATDD) is a development methodology based on communication between the business customers, the developers, and the testers. ATDD encompasses many of the same practices as specification by example, behavior-driven development (BDD), example-driven development (EDD), and support-driven development also called story test–driven development (SDD). All these processes aid developers and testers in understanding the customer's needs prior to implementation and allow customers to be able to converse in their own domain language.

U.S. AIR FORCE

Extreme DevOps Risk Management Example – Chaos Monkey (Netflix)

Chaos Monkey is a tool invented in 2011 by Netflix to test the resilience of its IT infrastructure. It works by intentionally disabling computers in Netflix's production network to test how remaining systems respond to the outage. Chaos Monkey is now part of a larger suite of tools called the Simian Army designed to simulate and test responses to various system failures and edge cases.

As part of the DevOps movement, special attention is paid to the safe operation of computer systems, thus providing a sufficient level of confidence despite frequent releases. By contributing to the DevOps Tool Chain, Chaos Monkey meets the need for continuous testing.

They are part of the pattern "Design for failure", "designed to support failure": a computer application must be able to support the failure of any underlying software or hardware component.

https://en.wikipedia.org/wiki/Chaos_Monkey

Kubernetes (Container Orchestration)

- Kubernetes is a portable, extensible open-source platform for managing containerized workloads and services, that facilitates both declarative configuration and automation. It has a large, rapidly growing ecosystem. Kubernetes services, support, and tools are widely available.
- Google open-sourced the Kubernetes project in 2014. Kubernetes builds upon a decade and a half of experience that Google has with running production workloads at scale, combined with best-of-breed ideas and practices from the community.
- Kubernetes provides a container-centric management environment. It orchestrates computing, networking, and storage infrastructure on behalf of user workloads. This provides much of the simplicity of Platform as a Service (PaaS) with the flexibility of Infrastructure as a Service (IaaS), and enables portability across infrastructure providers.

Kubernetes Benefits

- Allows for the deployment of complex applications (ie. Multiple microservices) with a single command.
- Allows for rolling updates and rollbacks with no downtime.
- Deploys security stack as a side-car container to gain complete visibility, including security posture, MFA enforcement, identity management, logging etc.
- Auto-scaling – allows to provision resources dynamically based on use and maximums, saving significant Cloud computing costs
- Restarts containers seamlessly and automatically in case of crash or other issue.

Helm

Helm helps you manage Kubernetes applications — Helm Charts helps you define, install, and upgrade even the most complex Kubernetes application.

Charts are easy to create, version, share, and publish — so start using Helm and stop the copy-and-paste madness.

The latest version of Helm is maintained by the CNCF - in collaboration with Microsoft, Google, Bitnami and the Helm contributor community.

- **Enable the creation of packages for easy deployment of complex container/Kubernetes based solutions.**
- **Allows for versioning, rolling-update and rollbacks.**

U.S. AIR FORCE

DevOps Benefits (1)

- Implementing DevOps allows organizations to get more done. DevOps promotes teamwork by eliminating silos and encouraging collaboration. Teams that adopt the DevOps model are able to increase lead time, create new features at a faster pace, all while driving innovation and increasing employee engagement and communication. In turn, they are making applications more secure and stable.
- When leveraging DevOps and implementing continuous integration and continuous delivery (CI/CD), organizations can see a tremendous improvement in deployment frequency, lead time, detection of cybersecurity vulnerabilities and flaws, mean time to repair and mean time to recovery.

Learn more :
https://www.forbes.com/sites/forbestechcouncil/2018/07/19/the-two-biggest-disruptions-to-cybersecurity-since-the-invention-of-the-firewall/

U.S. AIR FORCE

DevOps Benefits (2)

- Allows for rapid experimentation and uncertainty while focusing on mission end goals.
- Enable rapid prototyping and A/B testing or canary releases.
- New services and innovations are made available
- Frequency of deployments is increased
- Collaboration between various department is increased
- Time spent maintaining applications and fixing is greatly reduced
- Number of end users who are using organizations' services is increased
- Performance and quality of applications is improved
- Time-to-market is reduced
- Time needed for testing, development and operations is reduced.

U.S. AIR FORCE

DevOps Benefits (3)

- Avoids shadow IT by enabling all directorates/divisions/services to reuse the DevSecOps pipeline instead of reinvent the wheel and building their own pipeline without supervision
- Avoids technical debt by continuously fixing bugs and security issues thanks to automated tests and real-time scans.

DevOps Metrics

- Mean-time to recovery: shows how long it would take applications in the production stage to recover from failure
- Mean-time to production: show how long it takes when new code is committed into code repository
- Average lead-time: shows how long it would take for a new requirement to be delivered and deployed
- Deployment speed: shows how fast you can deploy a new version of the application between staging, test and production
- Deployment frequency: shows how often you can deploy a new release into production environment and testing / QA.
- Production failure rate: shows how often software fails during production

U.S. AIR FORCE

DevSecOps Metrics

- Ability to detect and prevent security flaws and injections.
- Ability to perform fuzzing and static/dynamic source code analysis.
- Ability to monitor container security including container base images and libraries.

DevOps Challenges (1)

1. Culture

Solution: Organizations should focus on building a collaborative culture with shared goals. This also includes finding employees who are DevOps champions in the organization.

2. Test automation

Solution: Many organizations neglect test automation while focusing on CI/CD deployments. Continuous testing is key for DevOps success, and security must be considered from the outset.

3. Legacy systems

Solution: Include modeling for legacy infrastructure and applications in your DevOps plans. Installing new hardware or software to coexist with older systems is always difficult.

4. App complexity

Solution: Consider application architecture changes based on on-premises, cloud, and containers early on in the process.

5. No DevOps plan

Solution: Create a clear plan that includes milestones, project owners, and well-defined deliverables.

DevOps Challenges (2)

6. Managing environments

Solution: Your organization can standardize and automate complex DevOps environments with cloud sandboxes and other tools.

7. Skillset

Solution: Teams need training on DevOps. Organizations should standardized processes and establish common operational procedures.

8. Budget

Solution: Remember that opensource does not mean free, and factor in integration and operational complexity in your costs.

9. Tools

Solution: Avoid fragmented toolset adoption, which can add to your costs.

10. Leadership support

Solution: Educate leadership about the benefits of DevOps, in order to gain resource and budget support.

Recommended Books

- **Recommended Books**
  - **A Seat at the Table – by Mark Schwartz (former CIO of USCIS, leader in Agile)**

  This book is highly recommended for ALL leadership as it is not technical but focused on the challenges around business, procurement and how leadership can enable DevOps across the organization and remove impediments.
  - **The Phoenix Project – by the founders of DevOps**
  - **The DevOps Handbook – by Gene Kim, Patrick Debois.**

For those who drive to work like me (for hours), please note that these books are available as Audiobooks.

U.S. AIR FORCE

Thank You!

Nicolas Chaillan
Chief Software Officer, U.S. Air Force
usaf.cso@mail.mil