Introduction to DevOps
Christian Townsend

- Manager / Solutions Architect, Government Cyber at KPMG
- Former DevOps/Operations Engineer at Nuance Communications
- Passionate about:
  - Cloud Architecture
  - Infrastructure Management
  - Automation
  - Delivering high quality to customers
What is DevOps?

- **Combination of:**
  - Practices
  - Tools:
    - Planning
    - Building
    - Testing
    - Deploying
    - Monitoring
  - Culture shift – move from siloed teams to shared responsibility
  - Heavy focus on automation and feedback
  - DevOps adoption is a journey, not a step
  - Primary benefit is to frequently deploy high quality software releases
What problems is DevOps trying to solve?

- Value deliverance
- Reduced cycle time
- Faster deployments
- Reduced deployment failures
- Faster recovery
- Time to market
The Three Ways - the principles underpinning DevOps

- **The First Way:** Flows/Systems Thinking
- **The Second Way:** Amplify Feedback Loops
- **The Third Way:** Culture of Continual Experimentation and Learning
The First Way - Flows/System Thinking

- Emphasises the performance of the entire system
- Limit work in progress
- Make work visible
- Do not pass defects downstream
The Second Way - Amplify Feedback Loops

- Create high quality information flow throughout value stream
- Feedback and Feedforward loops
- Complex systems typically have a high degree of interconnectedness
- Create feedback mechanisms where they are required
The Third Way - Continual Learning and Experimentation

• Create a culture that fosters continual experimentation, and understanding that repetition is the prerequisite to mastery
• Reserve time specifically for conducting such experiments
• Transform local discoveries into global improvements
• Promote injection of resilience into daily work
Supporting principles of DevOps

• Collaboration
• Automation
• Continuous Integration
• Continuous Delivery/Deployment
• Continuous Monitoring
• Continuous Improvement
Collaboration

- Teams traditionally worked in isolation (minimal collaboration)
- Shared responsibility
- Give teams greater autonomy
- Separate failure from blame
Automation

- Automate everything
- Repeatable Processes (simple and complex)
- Consistent
- Safer and more reliable
- Allows time to be spent on other activities
Continuous Integration

- Commit to main branch
- Robust testing regimes
- Trunk based development
- Reduce Developer coordination and associated overheads
Continuous Delivery and Deployment

- Deploy to production frequently (daily)
- Develop and Deploy in small batches
- Rollbacks are less impactful
- Easier to troubleshoot issues with deployment/code
- Customer receives improvements more regularly
- Developers see their work deployed more regularly
Continuous Monitoring

- Analyse performance of software:
  - What is normal behaviour? What is abnormal behaviour?
- It is critical to configure key metrics:
  - Hardware usage, User usage, Log telemetry
- Proactively respond to potential issues
- Enables customer to operate with minimal downtime
Continuous Improvement

- Plan
- Do
- Check
- Act

Key Benefits:
- Better quality
- Improved productivity
- Boost employee morale
- Customer satisfaction

Challenges:
- Bottlenecks
- Suppressed innovation
The challenges

- Initial effort
- Dealing with legacy systems
- Unique skill set required
- Culture shift as well as technical shift
Story Time
The Previous Architecture
The New Architecture
Infrastructure as Code (IaC)

• Describe and implement our cloud infrastructure using code
• Creates a layer of abstraction between declaration and implementation (human readable)
• Consistent
• Repeatable
• Can be automated
• Version controlled
Terraform

- IaC tool
- Created by HashiCorp
- Open source
- Declarative
- “Cloud agnostic”
IaC Demonstration
Recommendations

• **Reading:**

• **Learning Pathway:**
  - DevOps Roadmap: [https://roadmap.sh/devops](https://roadmap.sh/devops)

Good luck!
Thank you