TLA+ COMMUNITY EVENT 2023

TLA+ at AWS: Past, Present, and Future

Cezara Drăgoi
Principal Applied Scientist
AWS

Scarlet Schwiderski-Grosche
Principal Tech Program Manager
AWS

Agenda

Introduction

- Automated Reasoning at AWS
- Open Source at AWS

TLA+ at AWS

- Past, Present, and Future

Conclusions

Motivation for joining the TLA+ Foundation

- Formal methods

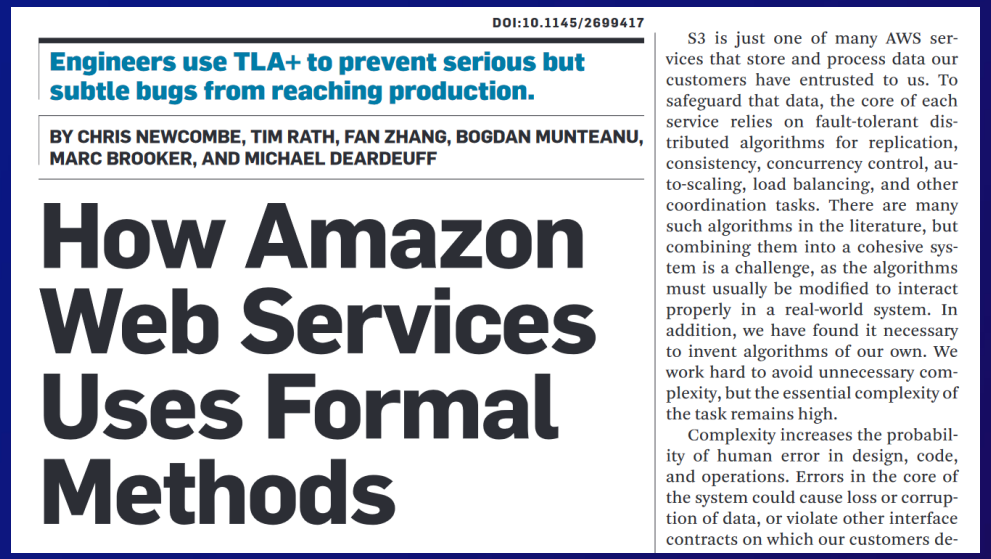
Motivation for joining the TLA+ Foundation

- Formal methods
- Open Source

TLA+ track record at AWS

- One of the main industry users
- Over more than 10 years
- Dozens of projects

AUTOMATED REASONING GROUP



DOI:10.1145/2699417

Engineers use TLA+ to prevent serious but subtle bugs from reaching production.

BY CHRIS NEWCOMBE, TIM RATH, FAN ZHANG, BOGDAN MUNTEANU, MARC BROOKER, AND MICHAEL DEARDEUFF

How Amazon Web Services Uses Formal Methods

S3 is just one of many AWS services that store and process data our customers have entrusted to us. To safeguard that data, the core of each service relies on fault-tolerant distributed algorithms for replication, consistency, concurrency control, auto-scaling, load balancing, and other coordination tasks. There are many such algorithms in the literature, but combining them into a cohesive system is a challenge, as the algorithms must usually be modified to interact properly in a real-world system. In addition, we have found it necessary to invent algorithms of our own. We work hard to avoid unnecessary complexity, but the essential complexity of the task remains high.

Complexity increases the probability of human error in design, code, and operations. Errors in the core of the system could cause loss or corruption of data, or violate other interface contracts on which our customers de-

Communications of the ACM | April 2015 | Vol. 58 | No. 4

amazonjobs | Search for jobs by title or keyword | Location | Your job application

Leadership Principles

We use our Leadership Principles every day, whether we're discussing ideas for new projects or deciding on the best way to solve a problem. It's just one of the things that makes Amazon peculiar.

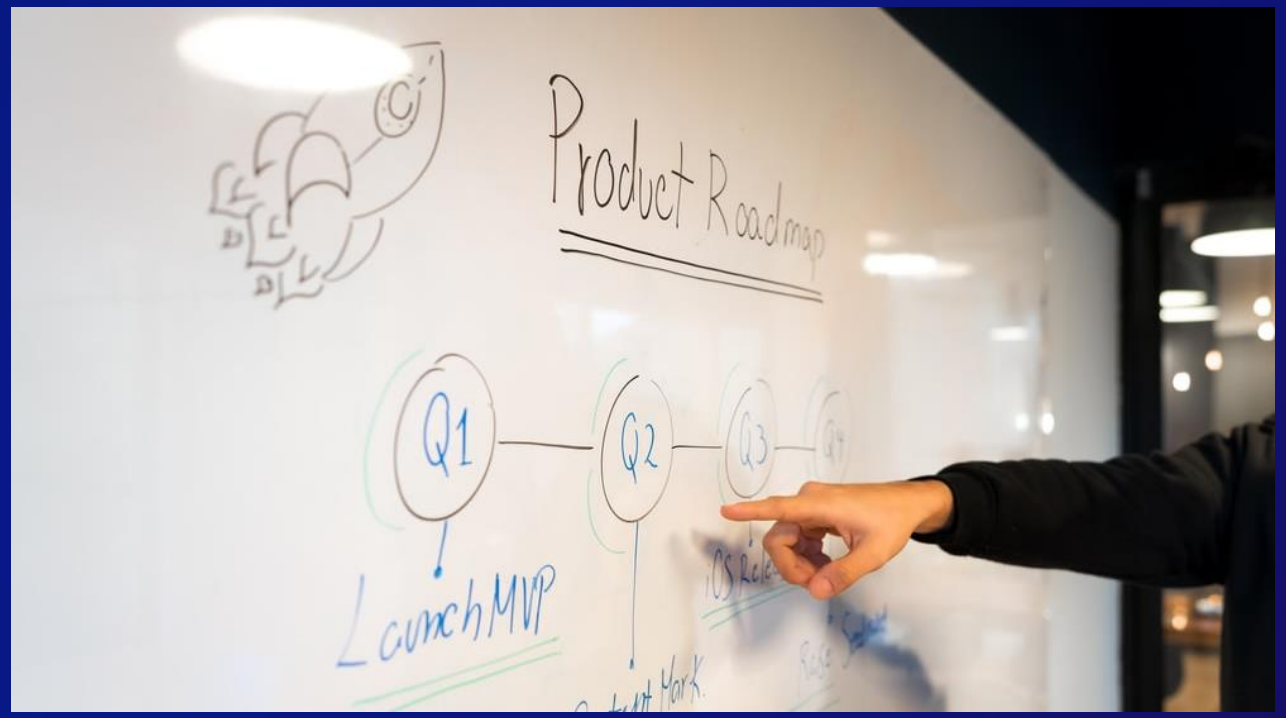
Customer Obsession

Leaders start with the customer and work backwards. They work vigorously to earn and keep customer trust. Although leaders pay attention to competitors, they obsess over customers.

Customer Obsession

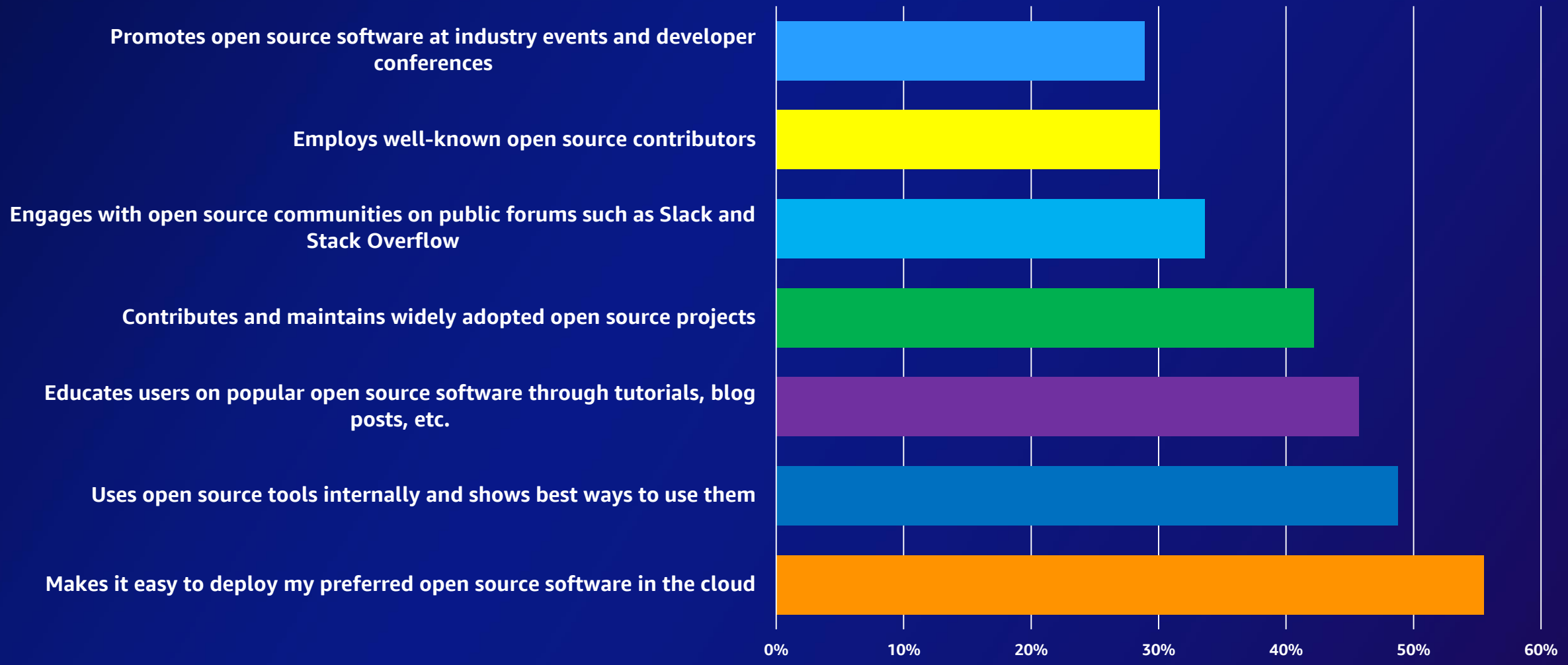
Leaders start with the customer and work backwards. They work vigorously to earn and keep customer trust. Although leaders pay attention to competitors,
they obsess over customers.

90% of the AWS roadmap is driven by direct customer feedback, and this is also true of open source.

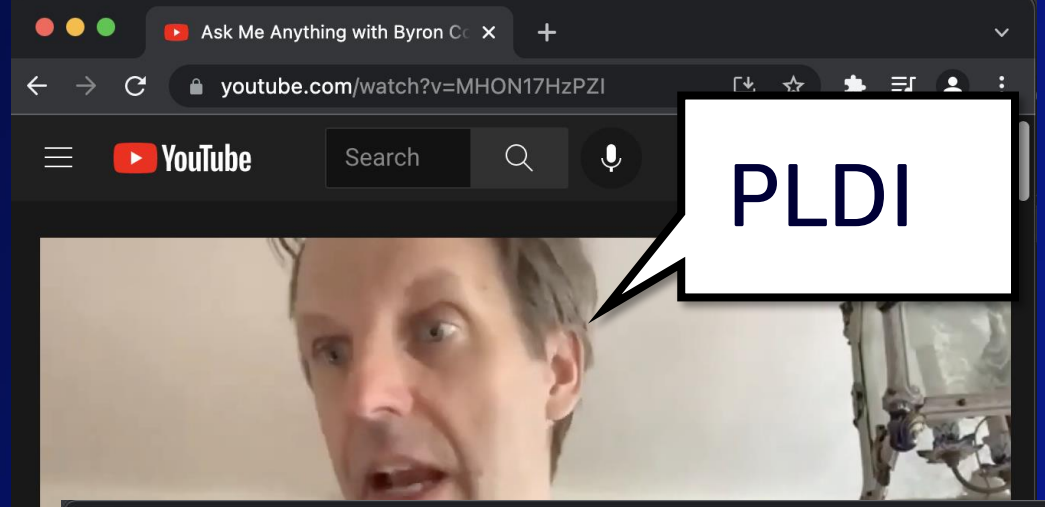


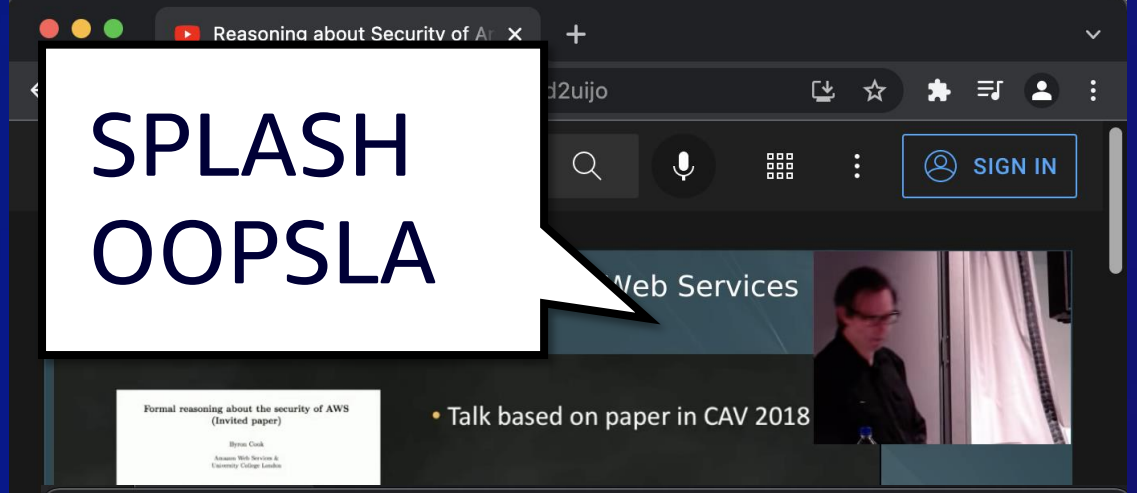
@slidebean

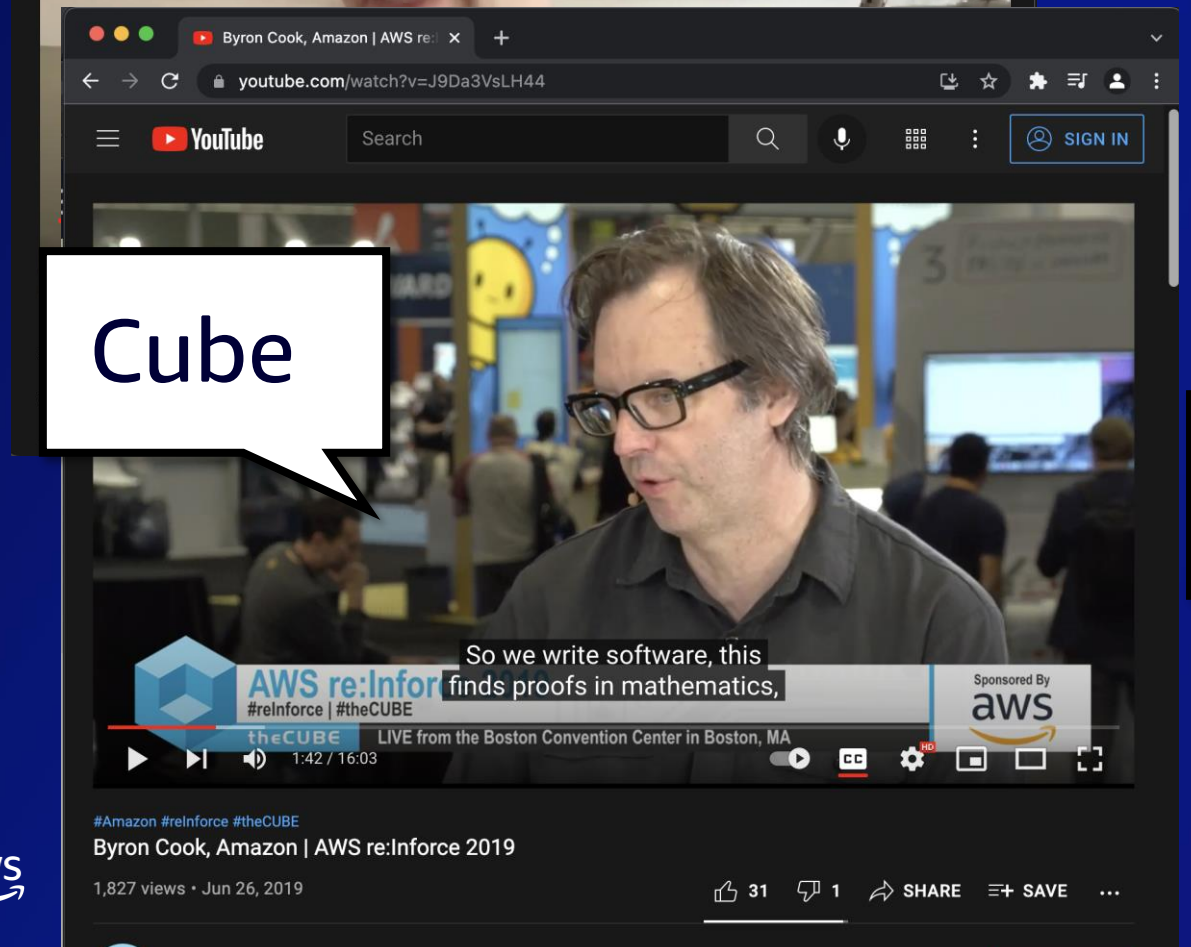
Open Source Leadership

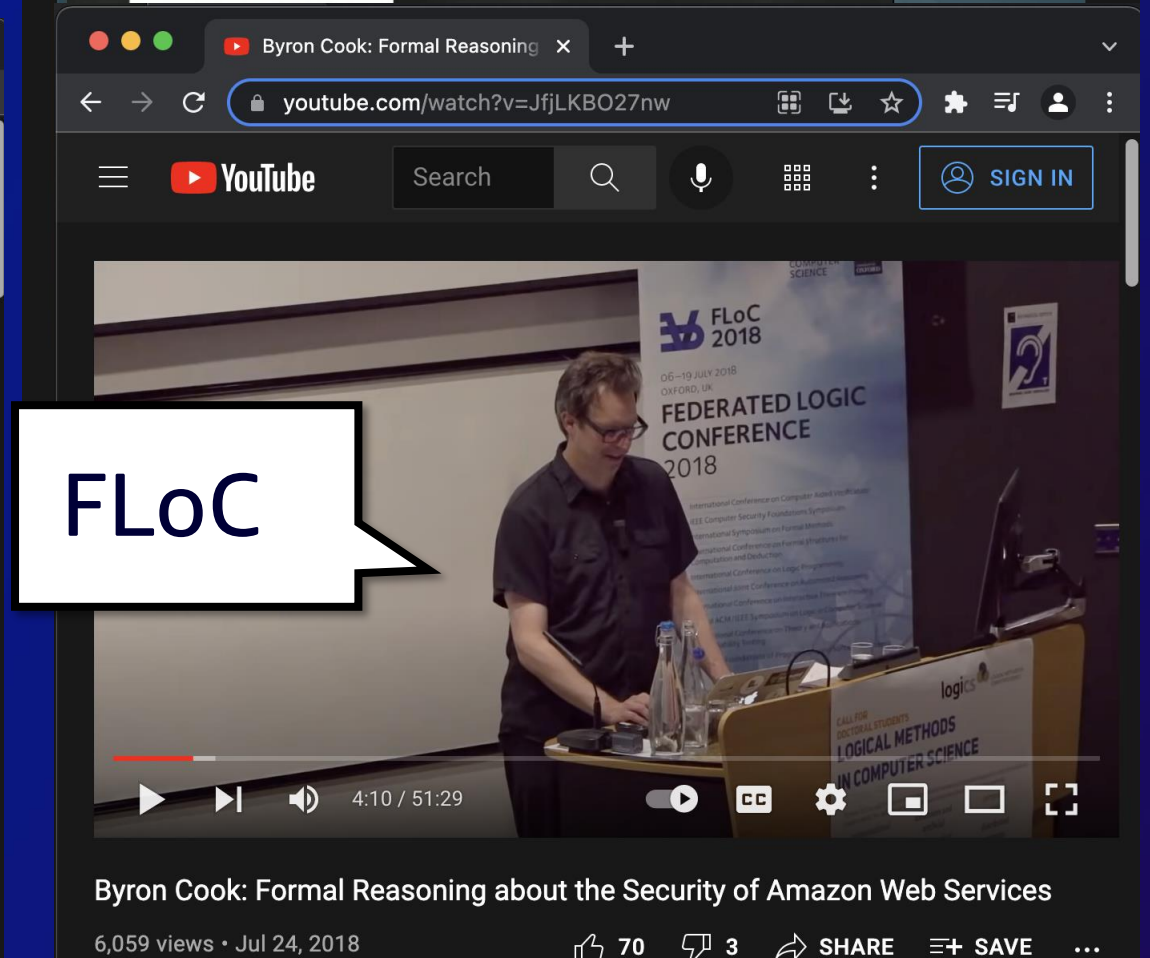


Source: AWS-commissioned survey, 2020


Ask Me Anything with Byron C
youtube.com/watch?v=MHON17HzPZI
YouTube
Search
PLDI


Reasoning about Security of Ar
SIGN IN
SPLASH
OOPSLA
Web Services
Formal reasoning about the security of AWS (Invited paper)
• Talk based on paper in CAV 2018


Byron Cook, Amazon | AWS re:
youtube.com/watch?v=J9Da3VsLH44
YouTube
Search
SIGN IN
Cube
So we write software, this
finds proofs in mathematics,
AWS re:Inforce
#reInforce | #theCUBE
Sponsored By
aws
theCUBE
LIVE from the Boston Convention Center in Boston, MA
1:42 / 16:03
#Amazon #reInforce #theCUBE
Byron Cook, Amazon | AWS re:Inforce 2019
1,827 views • Jun 26, 2019
31
1
SHARE
SAVE


Byron Cook: Formal Reasoning
youtube.com/watch?v=JfjLKBO27nw
YouTube
Search
SIGN IN
FLoC
FLoC 2018
FEDERATED LOGIC CONFERENCE 2018
4:10 / 51:29
Byron Cook: Formal Reasoning about the Security of Amazon Web Services
6,059 views • Jul 24, 2018
70
3
SHARE
SAVE

AWS re:Invent 2020 - Develope

youtube.com/watch?v=jt-gV1YwmnI

YouTube

Search

SIGN IN

#AWS

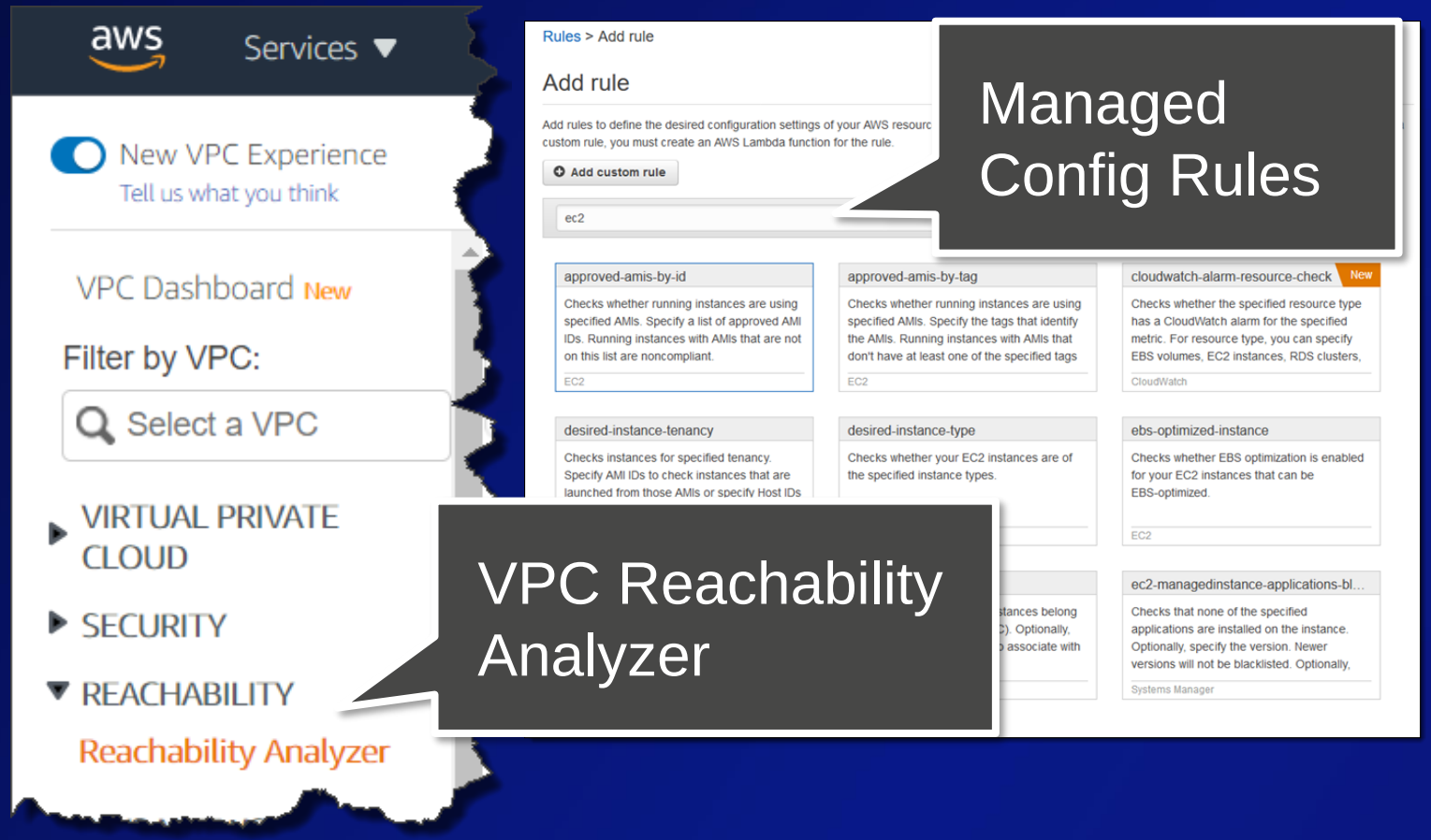
AWS re:Invent 2020 - Developer Keynote with Dr. Werner Vogels

417,708 views • Dec 18, 2020

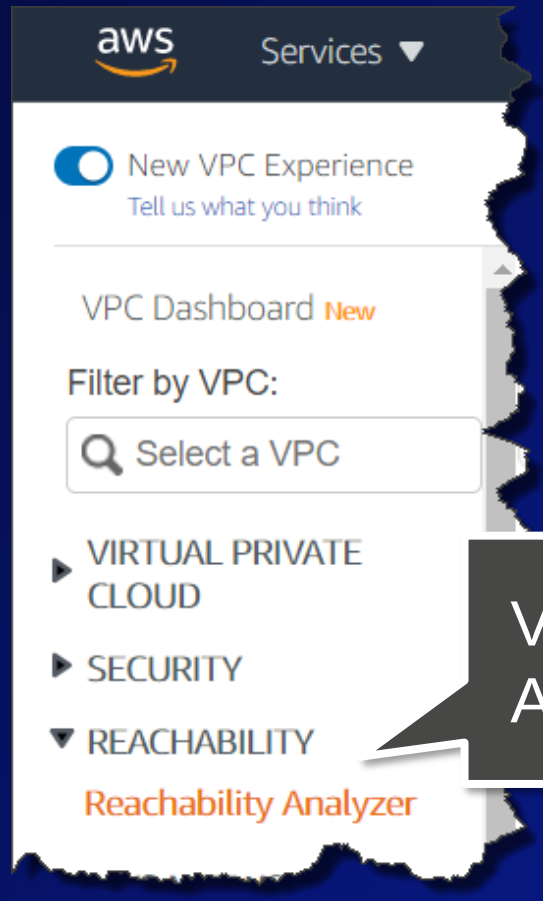
401 23 SHARE SAVE ...

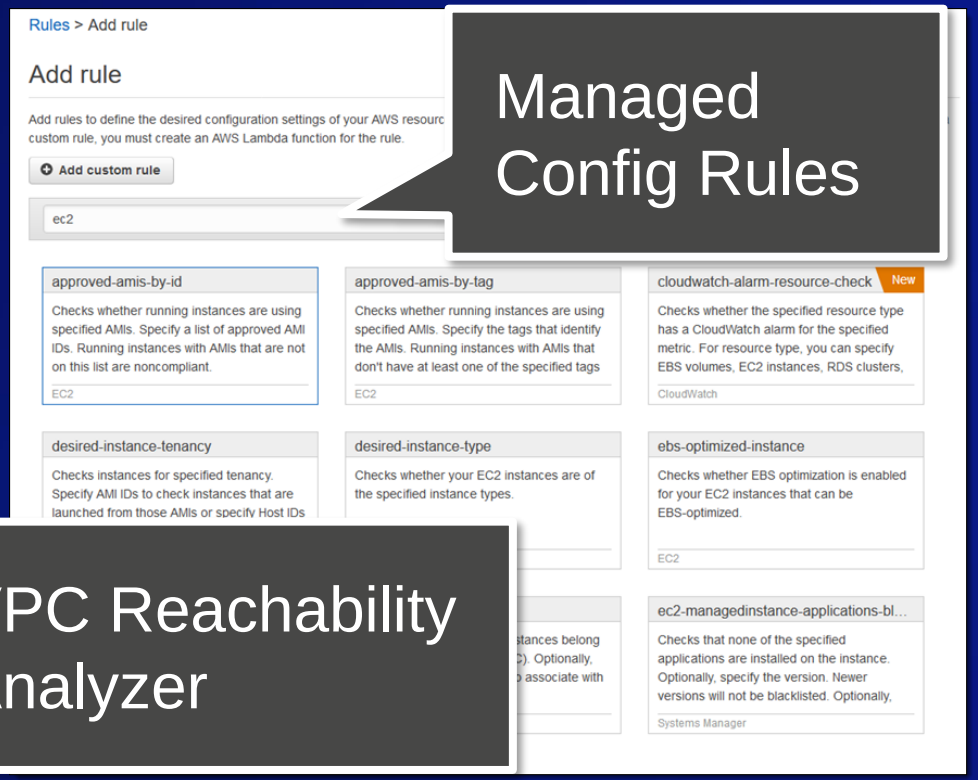
Examples of customer-facing features

Examples of customer-facing features



Examples of customer-facing features



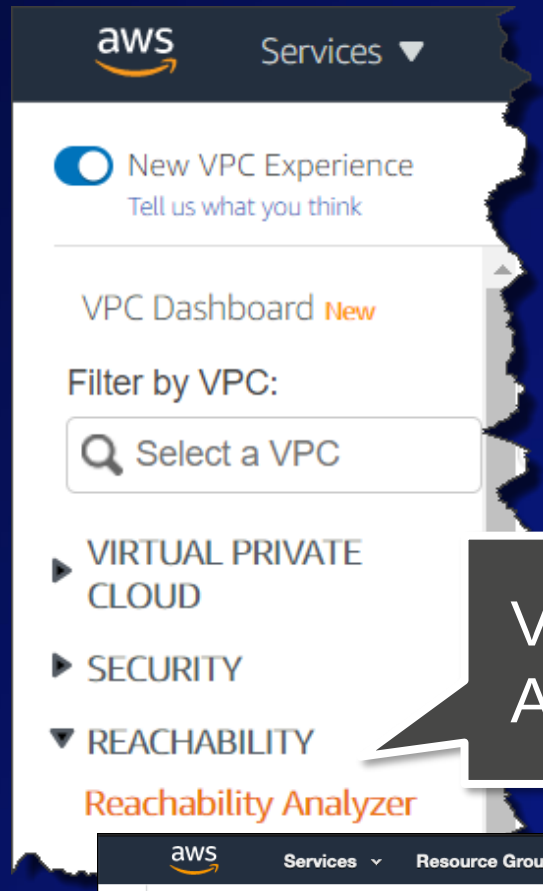


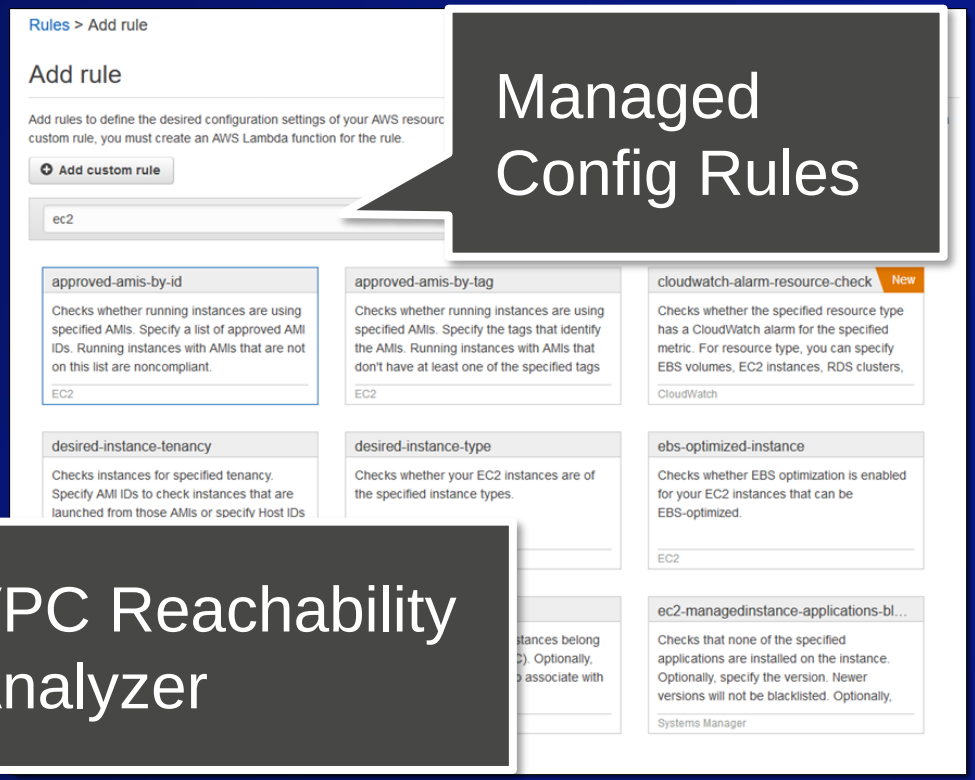
Managed Config Rules

VPC Reachability Analyzer

S3 Block Public Access

Examples of customer-facing features

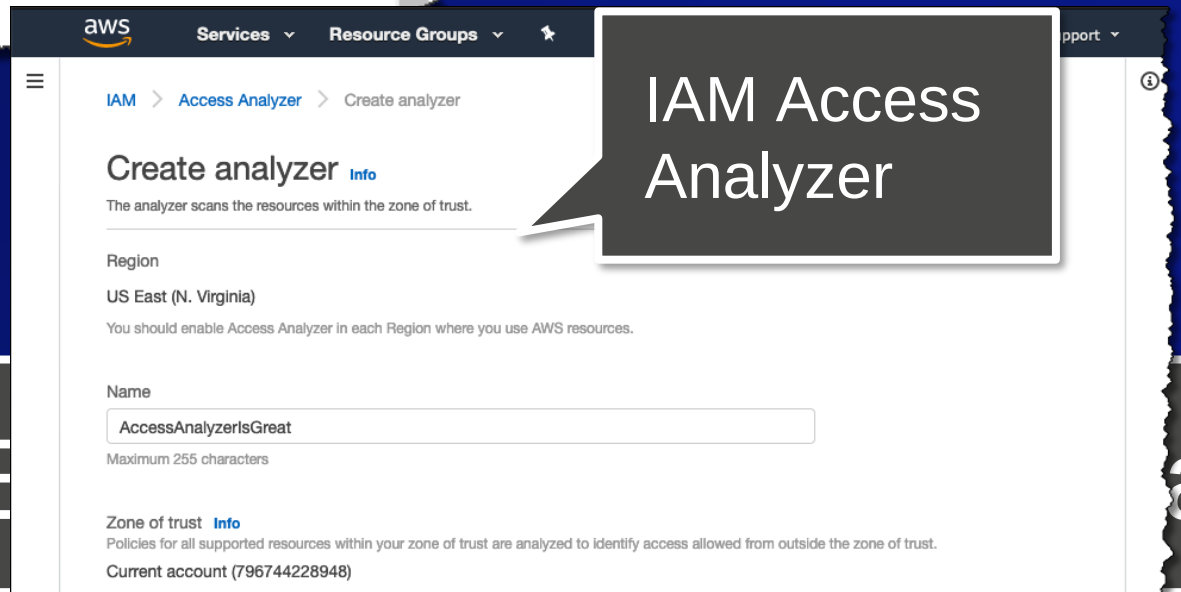




Managed Config Rules

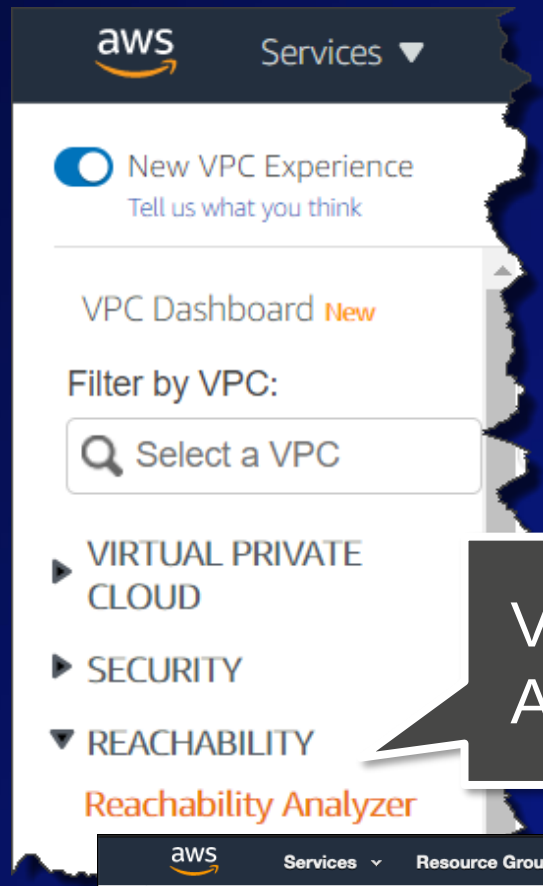
S3 Block Public Access

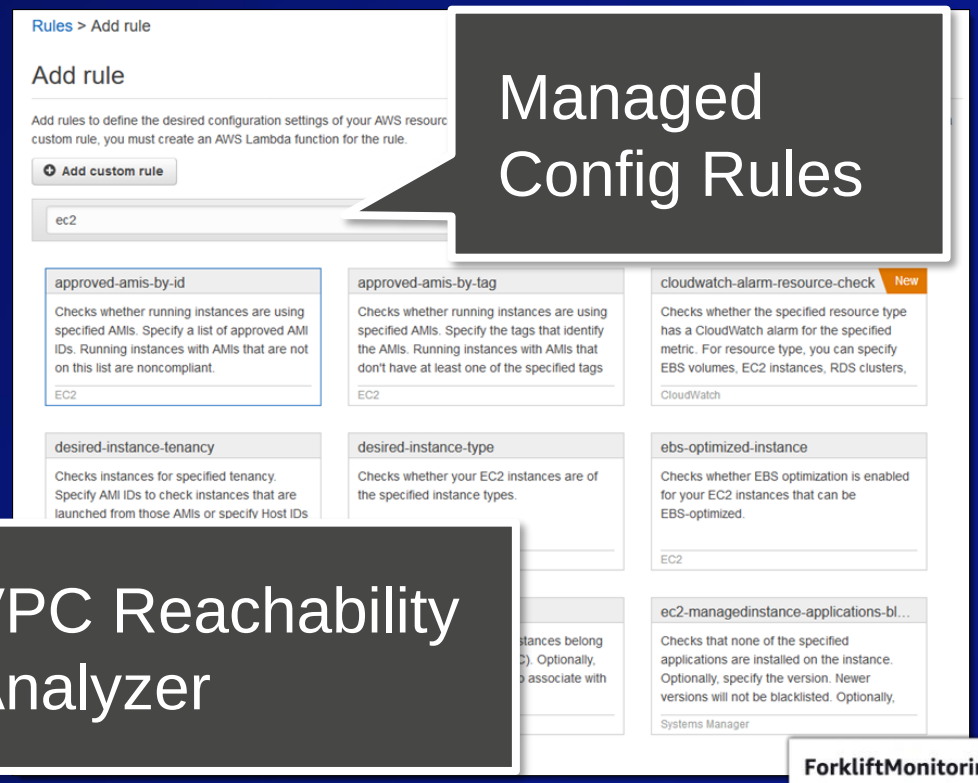
VPC Reachability Analyzer



IAM Access Analyzer

E acing features

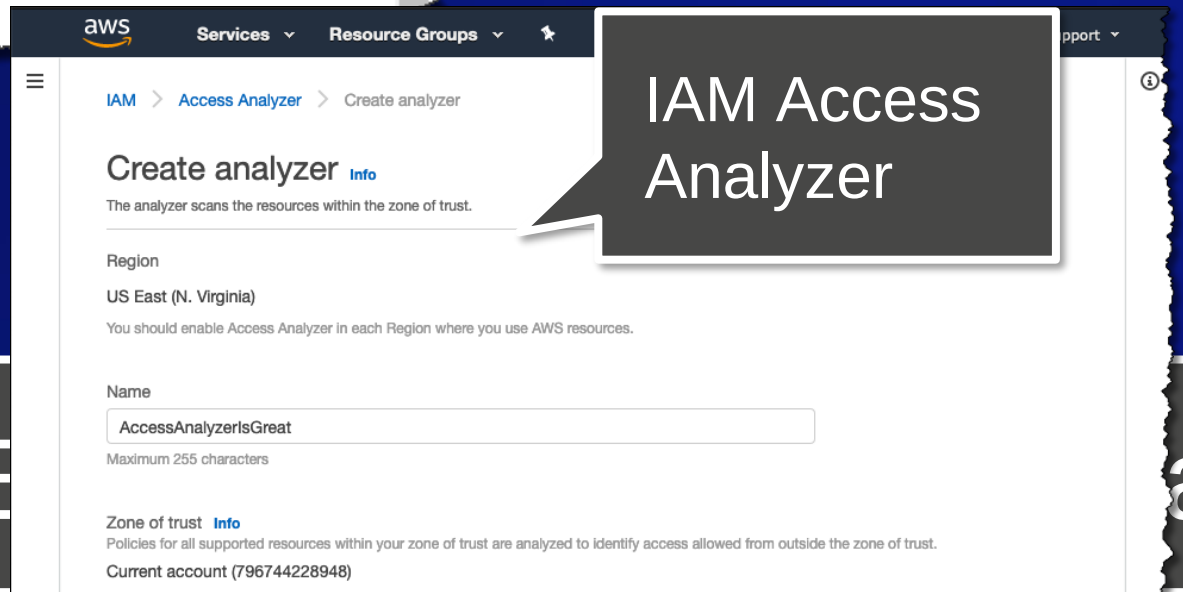




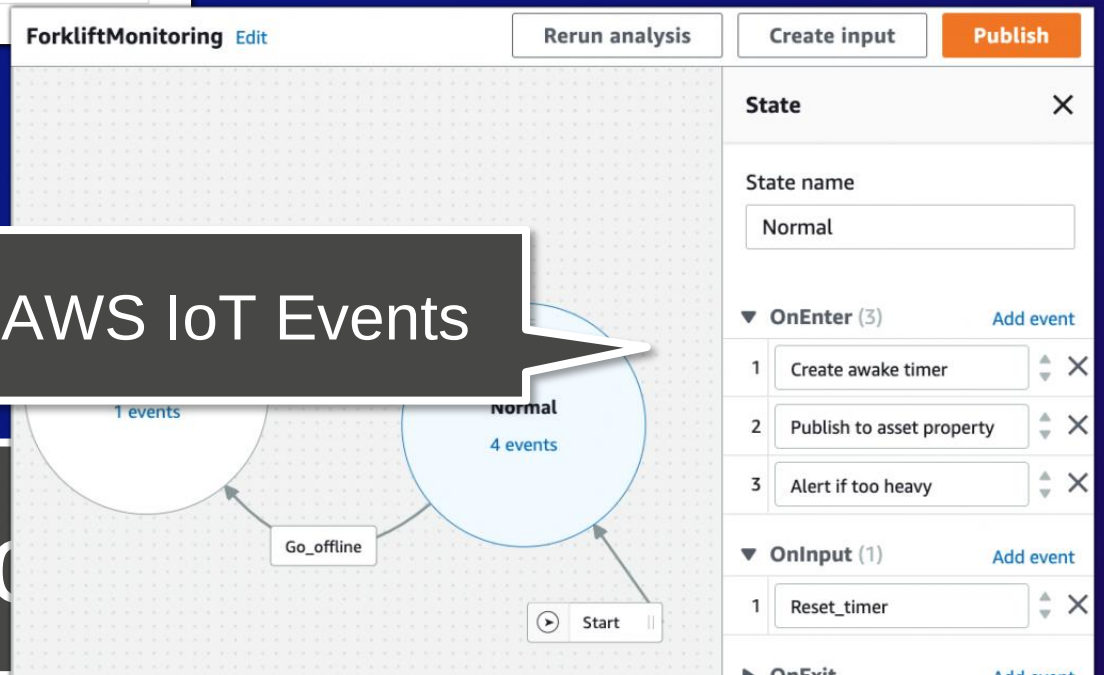
Managed Config Rules

S3 Block Public Access

VPC Reachability Analyzer



IAM Access Analyzer

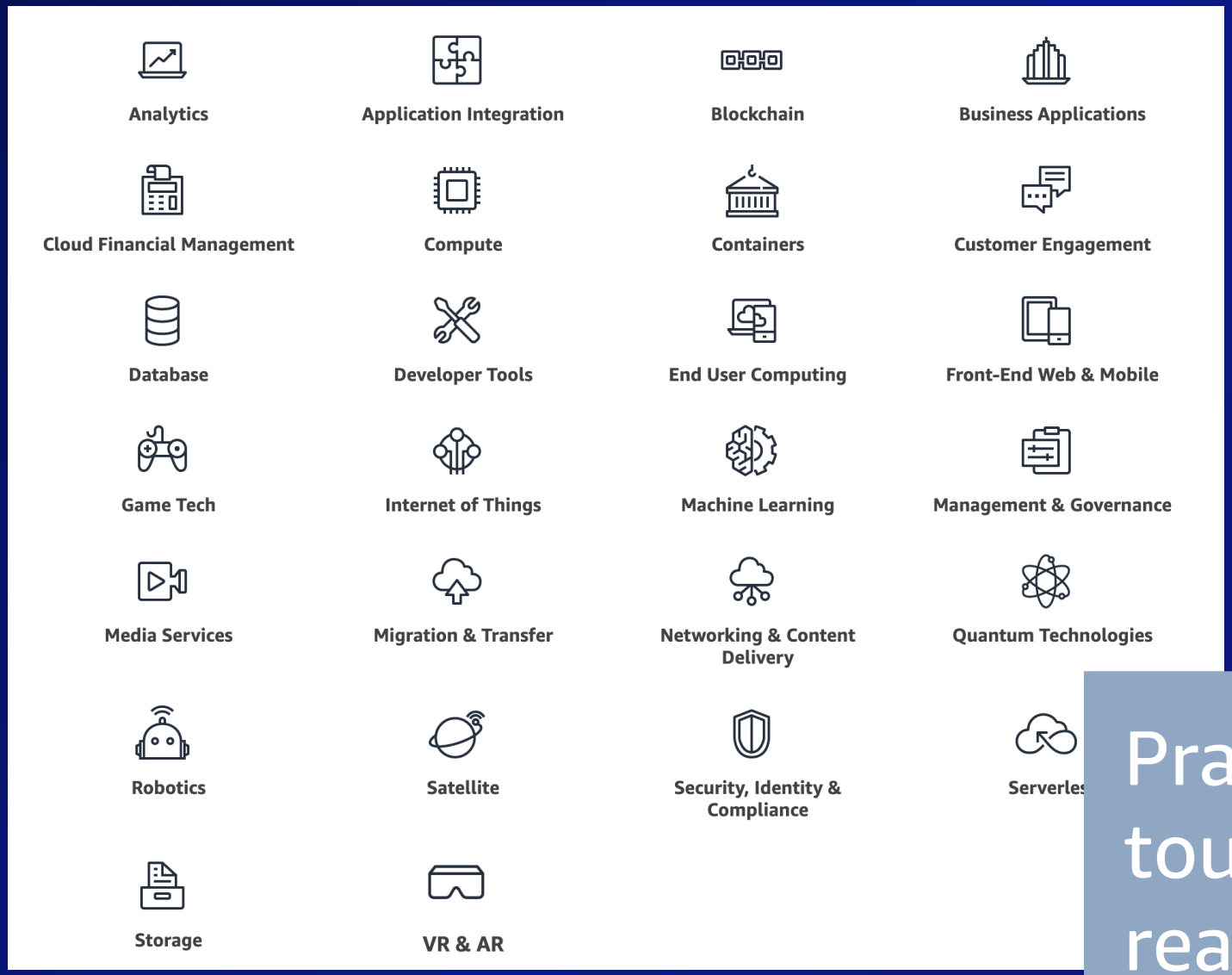


AWS IoT Events

AWS product categories

- Analytics
- Application Integration
- Blockchain
- Business Applications
- Cloud Financial Management
- Compute
- Containers
- Customer Engagement
- Database
- Developer Tools
- End User Computing
- Front-End Web & Mobile
- Game Tech
- Internet of Things
- Machine Learning
- Management & Governance
- Media Services
- Migration & Transfer
- Networking & Content Delivery
- Quantum Technologies
- Robotics
- Satellite
- Security, Identity & Compliance
- Serverless
- Storage
- VR & AR

AWS product categories



Practically every area touched by automated reasoning in some way

Scientists at *Principal* level or above


View Badge Photo
Rajeev Joshi


Gustavo Petri


Dominic Mulligan


Emina Torlak


Tancrède Lepoint


View Badge Photo
Jim Grundy


Muhammad Naveed


Cezara Drăgoi


Leo de Moura


Aws Albarghouthi


View Badge Photo
Jared Davis


John Harrison


Willem Visser


Zvonimir Rakamaric

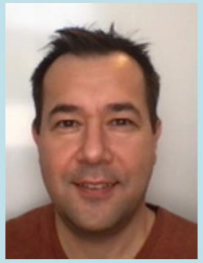

Cesar Munoz


Byron Cook


View Badge Photo
Jaco Geldenhuys


Mike Whalen


Andrew Gacek


Murat Demirbas


Ben Liblit


Mark Tuttle


Nathan Chong


View Custom Photo
Ernie Cohen

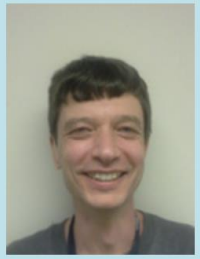

Mike Hicks


Sean McLaughlin


Remi Delmas


Rustan Leino


Serdar Tasiran


Bruno Dutertre


Temesghen Kahsai


Daniel Kroening


Dimitra Giannakopoulou


Lee Pike

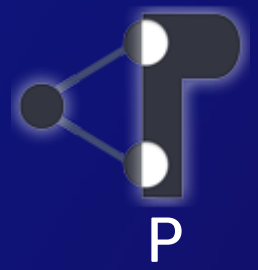

John Backes


Rod Chapman

AWS Zelkova

AMZN Dust

AWS Shuttle



AWS Tiros

CEDAR

Over to Cezara

"Use of formal methods at AWS" paper –2014

- "Since 2011, engineers at Amazon Web Services (AWS) have been using formal specification and model checking to help solve difficult design problems in critical systems."

- "We have found that testing the code is inadequate as a method to find subtle errors in design"

TLA+ modeling at Amazon

Code repo search shows more than 100 TLA+ models S3

- DynamoDB
- EBS
- Lambda
- Several internal services

Grassroots TLA+ use

Engineer has a hard distributed/concurrency problem

- hears about TLA+
- learns it on their own in less than a week
- writes a model, designs/mends the protocol
- moves on. . .

Takeaways

TLA+ did fill an important need

Use has been sporadic and on ad hoc basis

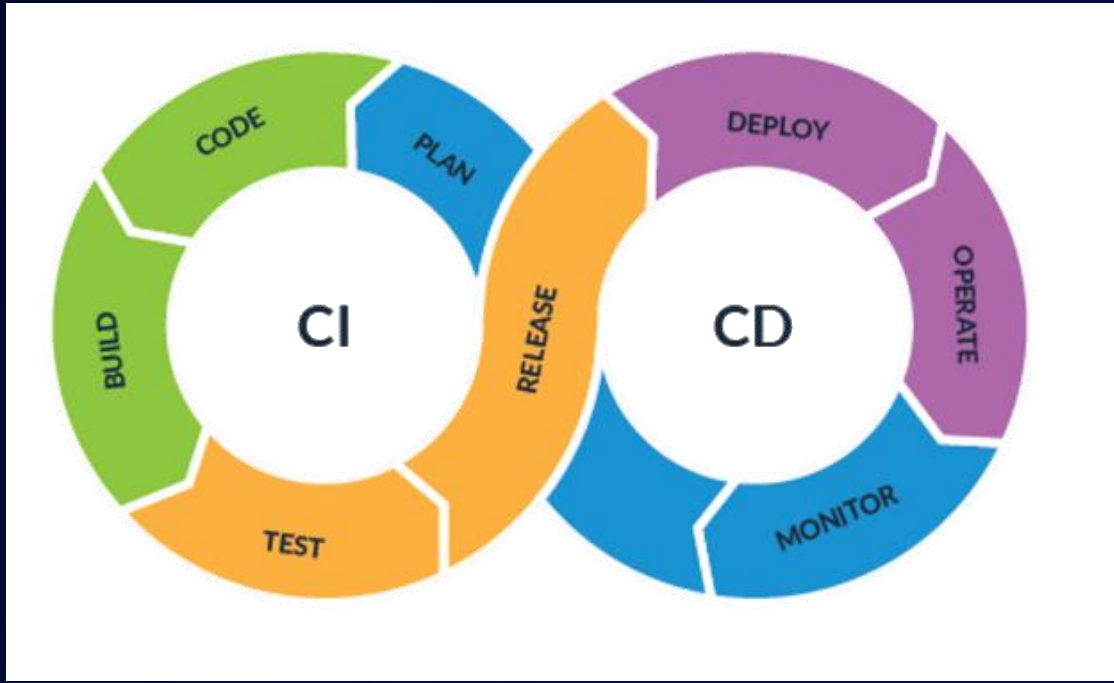
Motivated by an itch to scratch, quick ramp up was possible (survival bias?)

Present day software

- very large code bases
- composition of system of systems
- large teams
- continuous integration / continuous deployment (CI/CD)

We have found that testing the code is VERY DIFFICULT for software

CI/CD



- exploring protocols
- debugging concurrency
- code conformance
- adding features
- preventing code regression

TLA strengths and weaknesses

- TLA+ is declarative,
- succinct,
- high-level,
- is great for rapidly exploring distributed/concurrent protocol design space

But it is weak at

- Avoiding code drift
- Model-based testing
- Conformance checking
- Integration testing

Automated Reasoning at Amazon

- SMT solvers
- Code conformance checking support (Shuttle, P)
- High level modeling support (TLA+)
- Proofs and code generation from proved models (Dafny)

Dealing with scale

- random coverage rather than full coverage
- reference implementation testing
- compositional testing with abstract/mocks
- differential testing

TLA+

- TLA+ is still going strong because it is very strong at what it does: Debugging Designs!
- TLA+ is still grassroots and ad hoc

Future of TLA+: Reuse, Reduce, Recycle

- Without taking from the strengths of TLA+, can we bridge the gap between TLA+ modeling and large scale software deployments via CI/CD
- Reuse: How do we make TLA+ models more reusable?
- Reuse: How do we make TLA+ models more reusable to serve composable system of systems software?
- Recycle: Can we keep TLA+ sticky and check/prevent code drift from model?

Reducing friction

This was great!

- VSCode integration
- Adding statistical support peripherally
- Hillel’s work on popularizing TLA+

Can we do more?

Thank you!