AWS re:Invent

DOP310-R

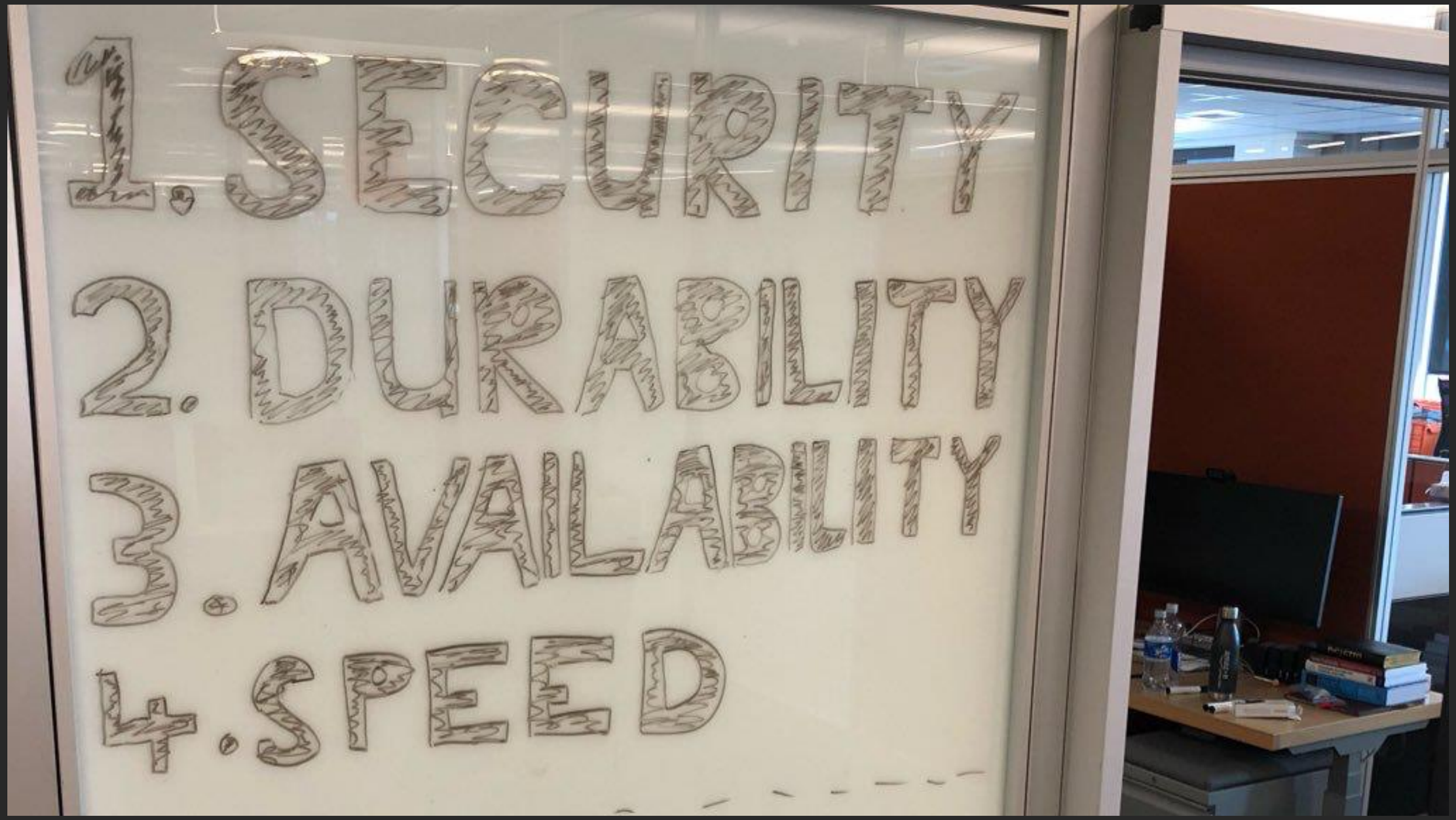
Amazon's approach to security during development

Colm MacCárthaigh
Senior Principal Engineer
Amazon Web Services

AWS re:Invent

© 2019, Amazon Web Services, Inc. or its affiliates. All rights reserved.

aws


1. SECURITY
2. DURABILITY
3. AVAILABILITY
4. SPEED

When am I getting that new feature?

Security as priority #1



Security as priority #1

- How do we promote and reward security in our development culture?
- How do we build security into our services?
- How do we react to new and emerging threats?
- How do we handle a security incident?

Promoting security in our culture

AWS re:Invent

© 2019, Amazon Web Services, Inc. or its affiliates. All rights reserved.

aws

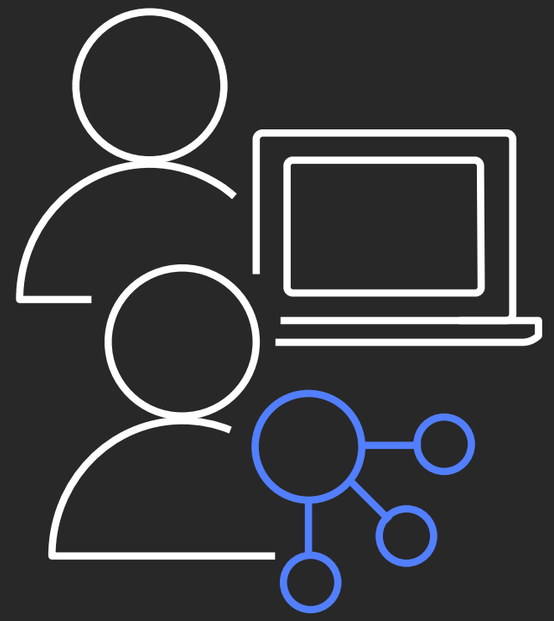
"Culture eats strategy for breakfast."

Peter Drucker

© 2019, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Security really, *really* matters to our leadership

If a team has a security issue, our CISO and CEO will be aware of it. We have well-oiled channels for talking to our security leadership team, and efficient escalation mechanisms, should we need them.

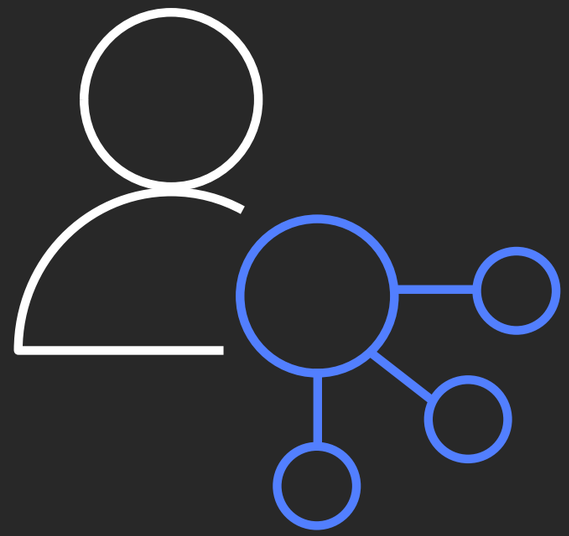


Principles and tenets



Ownership

Leaders are owners. They think long-term and don’t sacrifice long-term value for short-term results. They act on behalf of the entire company, beyond just their own team. They never say, “That’s not my job.”



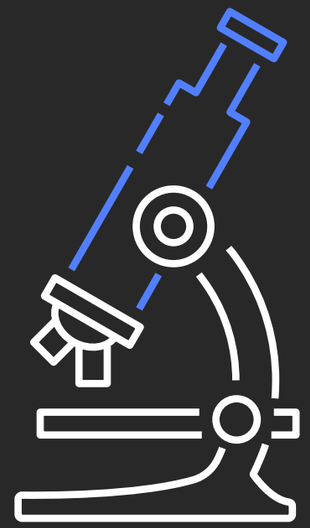
Insist on the highest standards

Leaders have relentlessly high standards—many people may think these standards are unreasonably high. Leaders are continually raising the bar and drive their teams to deliver high-quality products, services, and processes. Leaders ensure that defects do not get sent down the line and that problems are fixed so they stay fixed.



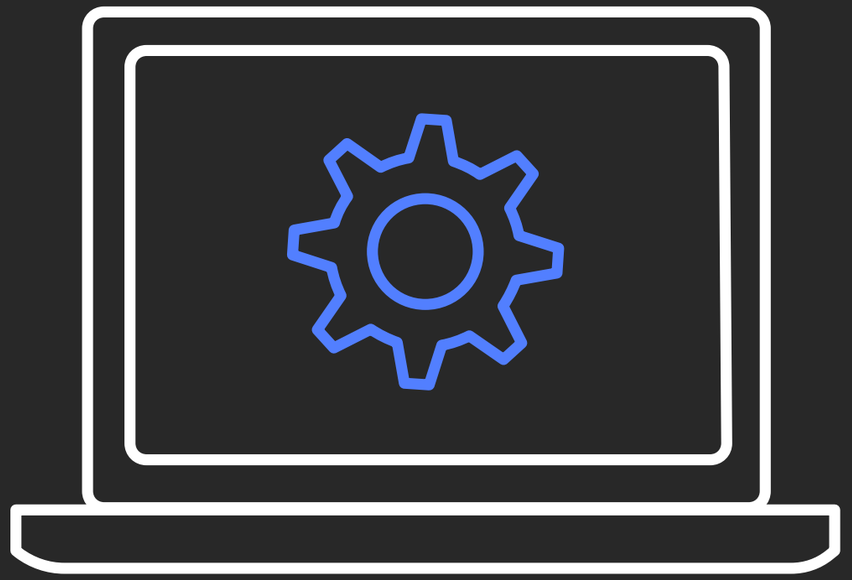
Dive deep

Leaders operate at all levels, stay connected to the details, audit frequently, and are skeptical when metrics and anecdote differ. No task is beneath them.



Stay simple

Write only as much code as is necessary, omit rarely used optional features, and support as few modes of operation as possible. We will also promote and encourage changes that reduce the size of our code base.



Write clear, readable code with a light cognitive load

Our code must be concise, easy to follow, and legible to a proficient programmer. Our code should be organized in a way that divides implementation into small units of work, with the entire necessary context at hand.



Defend in depth and systematically

Great care and attention to detail are required to write good code, but we also use automation and mechanistic processes to protect against human error. See also: ARC337-R.

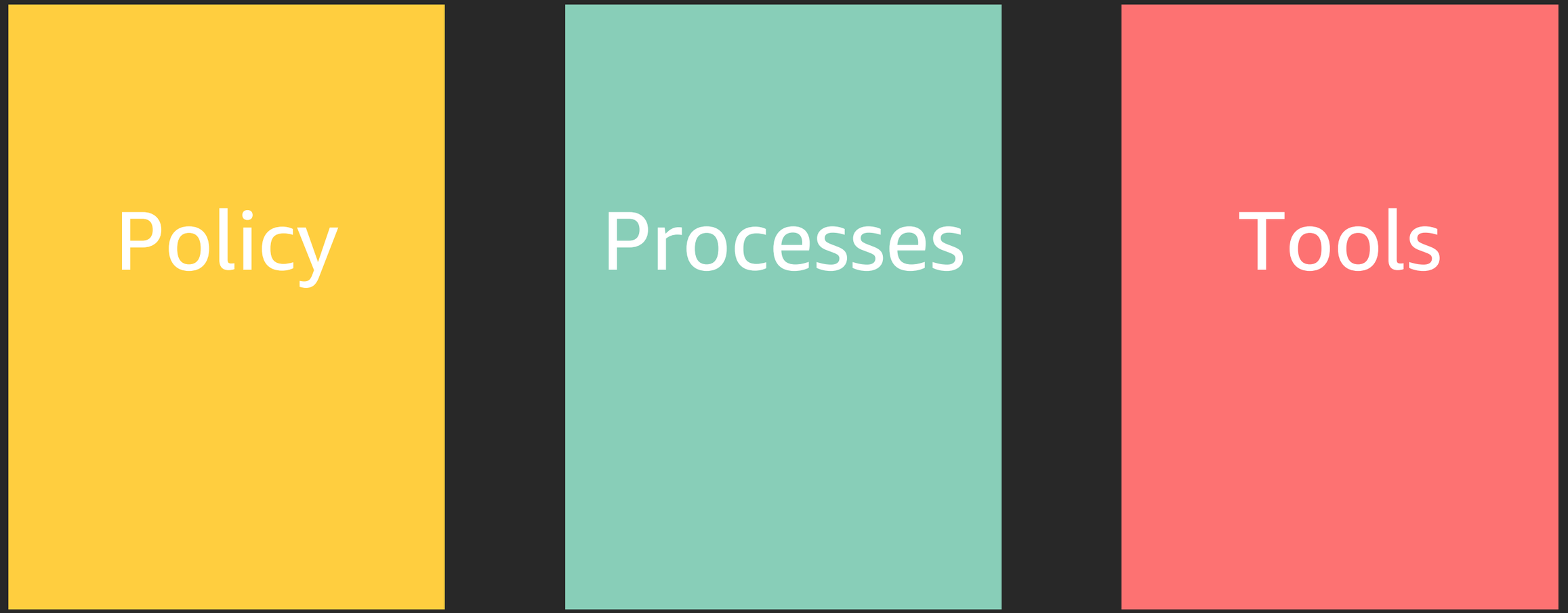


Technical fearlessness

Security requires humility and a radical acceptance of reality. Everyone must feel safe to admit faults and must have no incentive to deny the severity of a security risk.



The three pillars of security



The three pillars of security

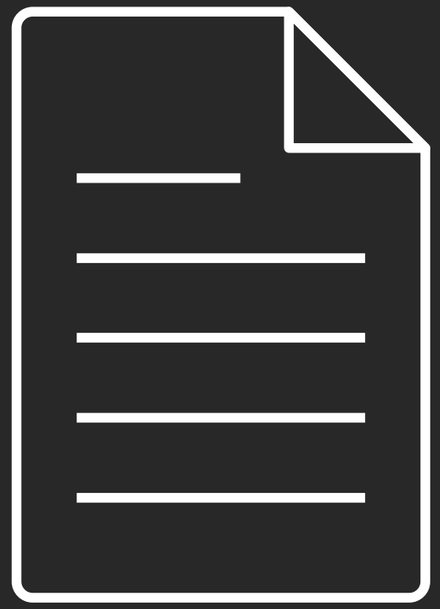


Building security into our services

© 2019, Amazon Web Services, Inc. or its affiliates. All rights reserved.

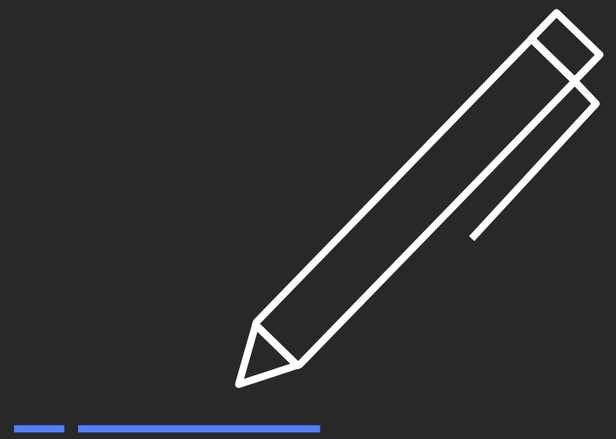
Security policies

We have security policies, security training, data encryption standards, PII and CII handling standards, compliance requirements, etc.



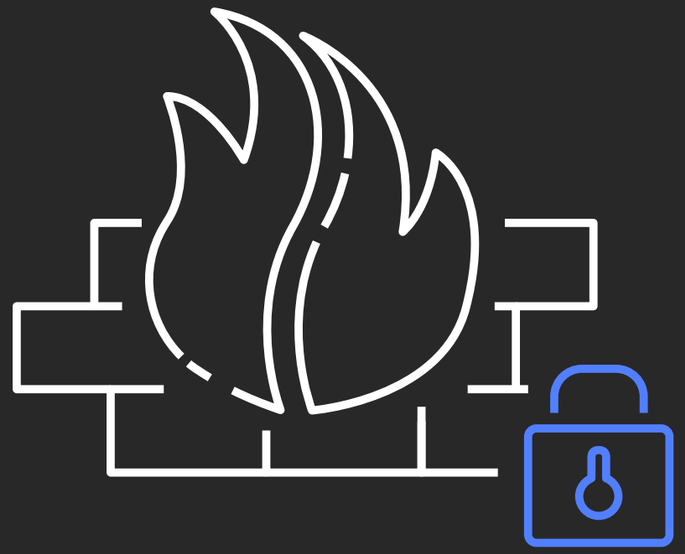
Security processes

We have security reviews for all new features and services. These reviews start early and continue as the service develops. We also do point-in-time reviews for smaller things.



Penetration testing

We perform penetration tests, fuzz testing, and other forms of vulnerability scanning



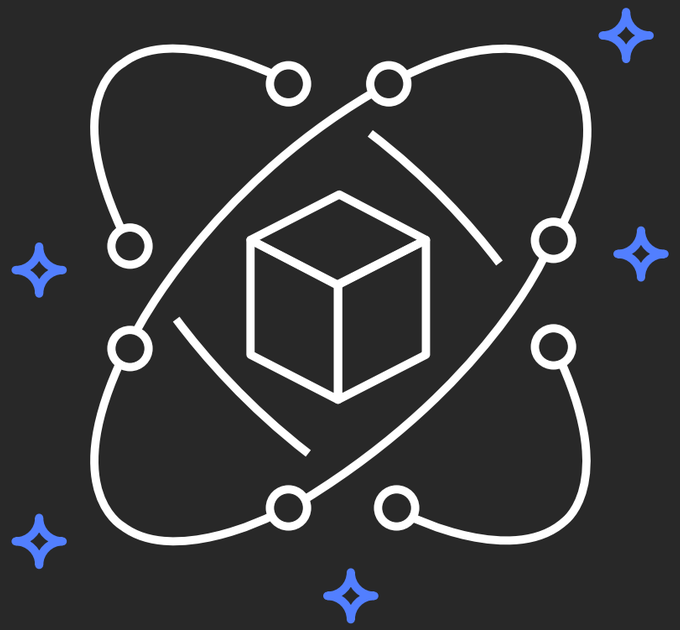
Formal verification

For especially sensitive projects, we work with our Automated Reasoning Group to formally verify the correctness of our work

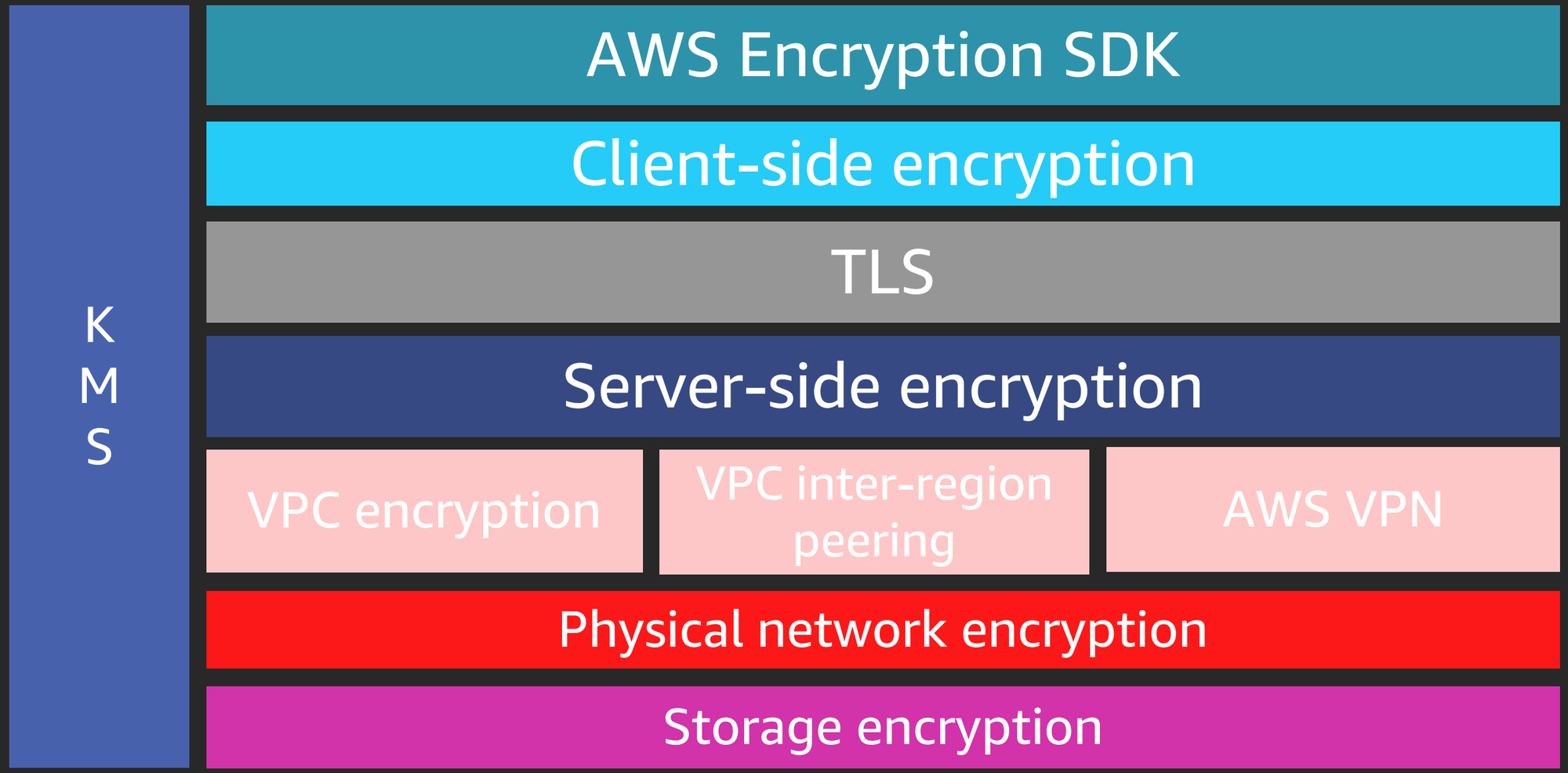


Building security in

Every new service supports TLS/SSL, SIGv4, AWS Identity and Access Management (IAM), AWS CloudTrail, AWS Config, and more, almost without any effort on the part of the development team



Building security in



Building security in

Every team also gets tools to manage their operations security. Internal security services include authentication, access control, audit trails, patch management, vulnerability notification, and more.



Reacting to new and emerging threats

AWS re:Invent

© 2019, Amazon Web Services, Inc. or its affiliates. All rights reserved.

aws

Hire and develop the best



Hire and develop the best

AES-GCM-SIV: Specification and Analysis

Shay Gueron[1], Adam Langley[2], and Yehuda Lindell[3⋆]

[1] University of Haifa, Israel and Amazon Web Services
[2] Google, Inc.
[3] Bar-Ilan University, Israel

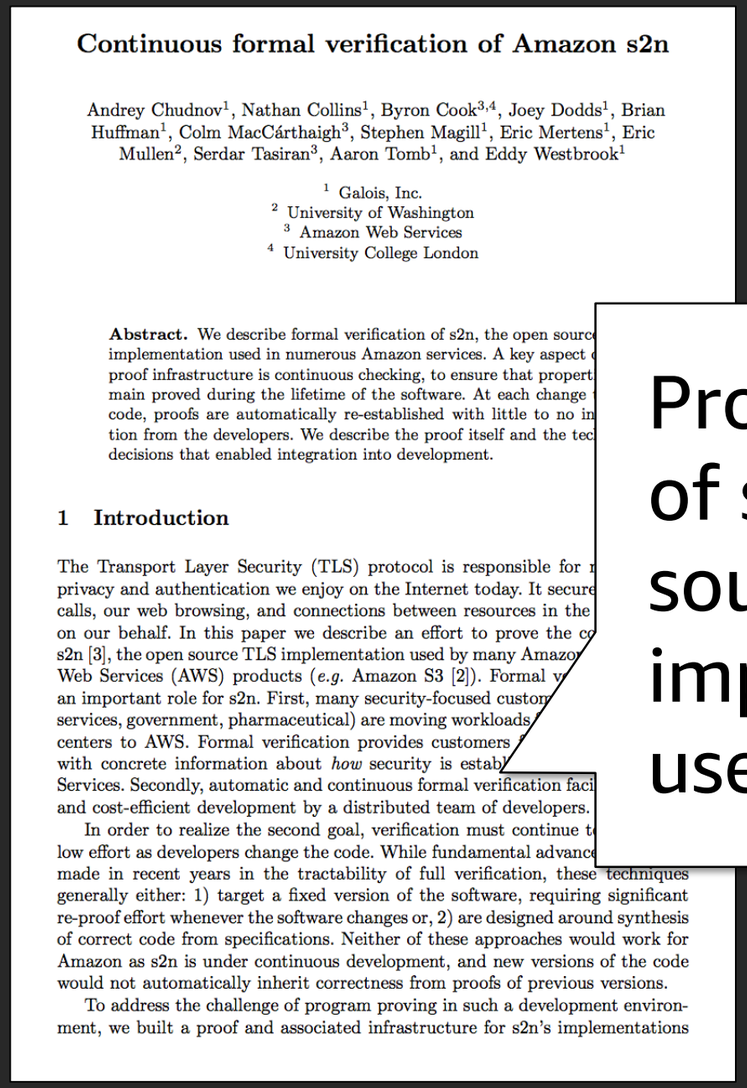
December 14, 2018

Selfie: reflections on TLS 1.3 with PSK

Nir Drucker and Shay Gueron

University of Haifa, Israel,
and
Amazon, Seattle, USA

Hire and develop the best



Continuous formal verification of Amazon s2n

Andrey Chudnov[1], Nathan Collins[1], Byron Cook[3,4], Joey Dodds[1], Brian Huffman[1], Colm MacCárthaigh[3], Stephen Magill[1], Eric Mertens[1], Eric Mullen[2], Serdar Tasiran[3], Aaron Tomb[1], and Eddy Westbrook[1]

[1] Galois, Inc.
[2] University of Washington
[3] Amazon Web Services
[4] University College London

Abstract. We describe formal verification of s2n, the open sourc
implementation used in numerous Amazon services. A key aspect
proof infrastructure is continuous checking, to ensure that propert
main proved during the lifetime of the software. At each change
code, proofs are automatically re-established with little to no in
tion from the developers. We describe the proof itself and the tec
decisions that enabled integration into development.

1 Introduction

The Transport Layer Security (TLS) protocol is responsible for
privacy and authentication we enjoy on the Internet today. It secure
calls, our web browsing, and connections between resources in the
on our behalf. In this paper we describe an effort to prove the co
s2n [3], the open source TLS implementation used by many Amazo
Web Services (AWS) products (*e.g.* Amazon S3 [2]). Formal v
an important role for s2n. First, many security-focused custom
services, government, pharmaceutical) are moving workloads
centers to AWS. Formal verification provides customers
with concrete information about *how* security is estab
Services. Secondly, automatic and continuous formal verification faci
and cost-efficient development by a distributed team of developers.

In order to realize the second goal, verification must continue t
low effort as developers change the code. While fundamental advance
made in recent years in the tractability of full verification, these techniques
generally either: 1) target a fixed version of the software, requiring significant
re-proof effort whenever the software changes or, 2) are designed around synthesis
of correct code from specifications. Neither of these approaches would work for
Amazon as s2n is under continuous development, and new versions of the code
would not automatically inherit correctness from proofs of previous versions.

To address the challenge of program proving in such a development environment, we built a proof and associated infrastructure for s2n's implementations

Proving correctness of s2n, the open-source TLS implementation used by AWS

Open and collaborative with security researchers

Security researchers are not our adversaries. We're all on the same side. Be thankful for security researchers' work, congratulate them on their findings, and work with them to develop mitigations and advance the state of the art. See also: OPN219.



Handling a security incident

© 2019, Amazon Web Services, Inc. or its affiliates. All rights reserved.

April 7, 2014



Anatomy of Heartbleed incident response

Early morning: The Heartbleed issue leaks, and the embargo “breaks”



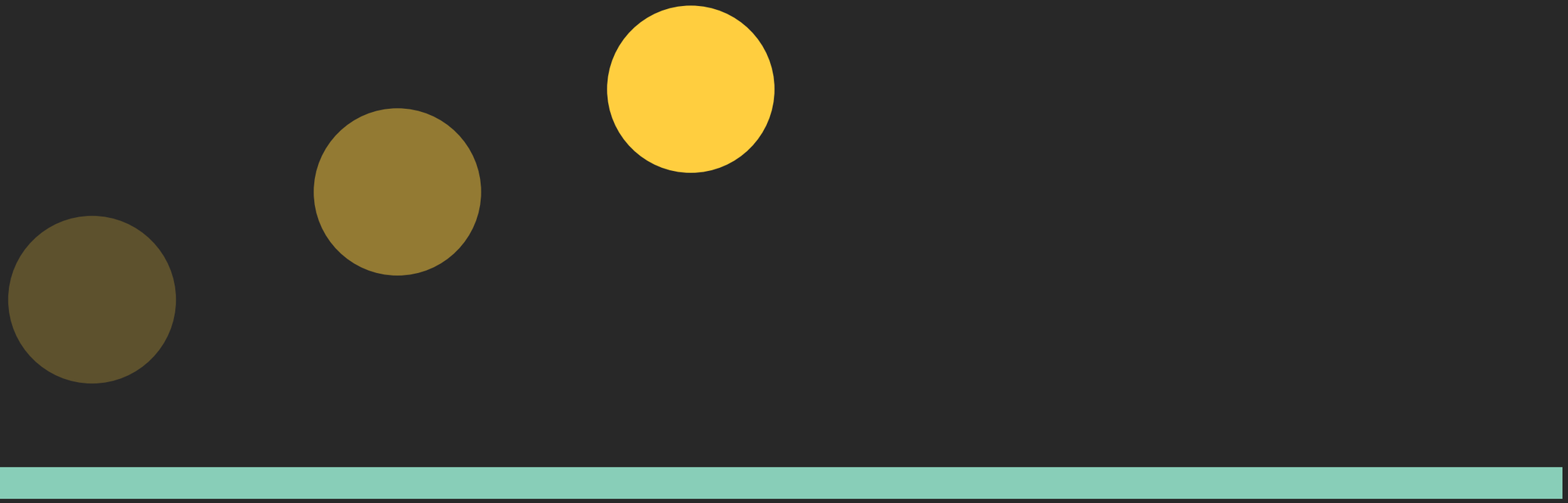
Anatomy of Heartbleed incident response

By 11 a.m., Amazon has activated the emergency response process



Anatomy of Heartbleed incident response

By noon, all deployments across Amazon have been temporarily frozen



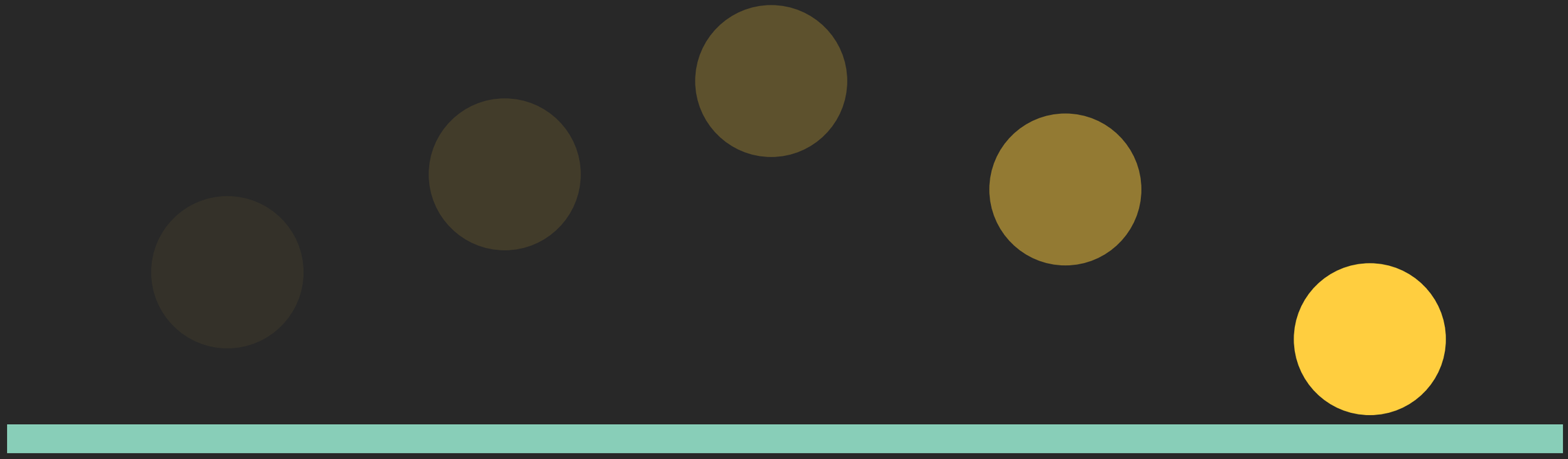
Anatomy of Heartbleed incident response

By early afternoon, we have prepared minimal security fix patches



Anatomy of Heartbleed incident response

Millions of deployments begin in the evening and are finished by 2 a.m., with no customer impact



Anatomy of Heartbleed incident response

SSL/TLS netfilter module

THIS MODULE IS AN EXPERIMENTAL WORK IN PROGRESS

nf_conntrack_tls is an experimental connection tracking module for linux which tracks the SSL/TLS state associated with an encrypted connection.

This module tracks the SSL/TLS state machine and drops connections that no longer appear to obey some semantics of the SSL/TLS protocols.

Initially, three protocol validations are performed:

1. This module understands TLS heartbeats and can detect and stop a heartbleed attempt.
2. All-non-handshake and heartbeat records are subject to an optional "suspicious sequence length" check. This check places a limit on how many bytes without their first bit set (ie byte value is < 128) can appear in a contiguous sequence. An SSL connection should consist of random-looking data, a suspiciously long sequence of zeroed bytes or ascii data may indicate that something nefarious is going on. The default length limit is 128, however this can be changed at module load time.
3. Directionality is enforced on SSL2 handshake messages.

Further validations are possible, including: TLS handshake directionality, protocol version tracking, protocol downgrade mitigation. These and more are future work, and contributions are welcome.

Anatomy of Heartbleed incident response

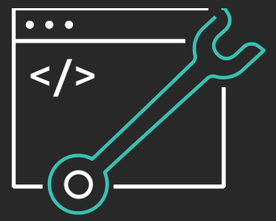


s2n is a C99 implementation of the TLS/SSL protocols that is designed to be simple, small, fast, and with security as a priority. It is released and licensed under the Apache License 2.0.



Learn DevOps with AWS Training and Certification

Resources created by the experts at AWS to propel your organization and career forward



Take free digital training to learn best practices for developing, deploying, and maintaining applications



Classroom offerings, like DevOps Engineering on AWS, feature AWS expert instructors and hands-on activities



Validate expertise with the **AWS Certified DevOps Engineer - Professional** or **AWS Certified Developer - Associate** exams

Visit aws.amazon.com/training/path-developing/

© 2019, Amazon Web Services, Inc. or its affiliates. All rights reserved.

aws training and certification

Thank you!

Colm MacCárthaigh

colmmacc@amazon.com
Twitter: colmmacc

AWS re:Invent

© 2019, Amazon Web Services, Inc. or its affiliates. All rights reserved.

aws



Please complete the session survey in the mobile app.

© 2019, Amazon Web Services, Inc. or its affiliates. All rights reserved.