NIST VIRTUAL WORKSHOP

Secure Development of GenAI Systems

An AWS Perspective

Mark Ryland

Director, Amazon Security

AWS’s approach

Security remains our top priority, now extended into this important and dynamic space

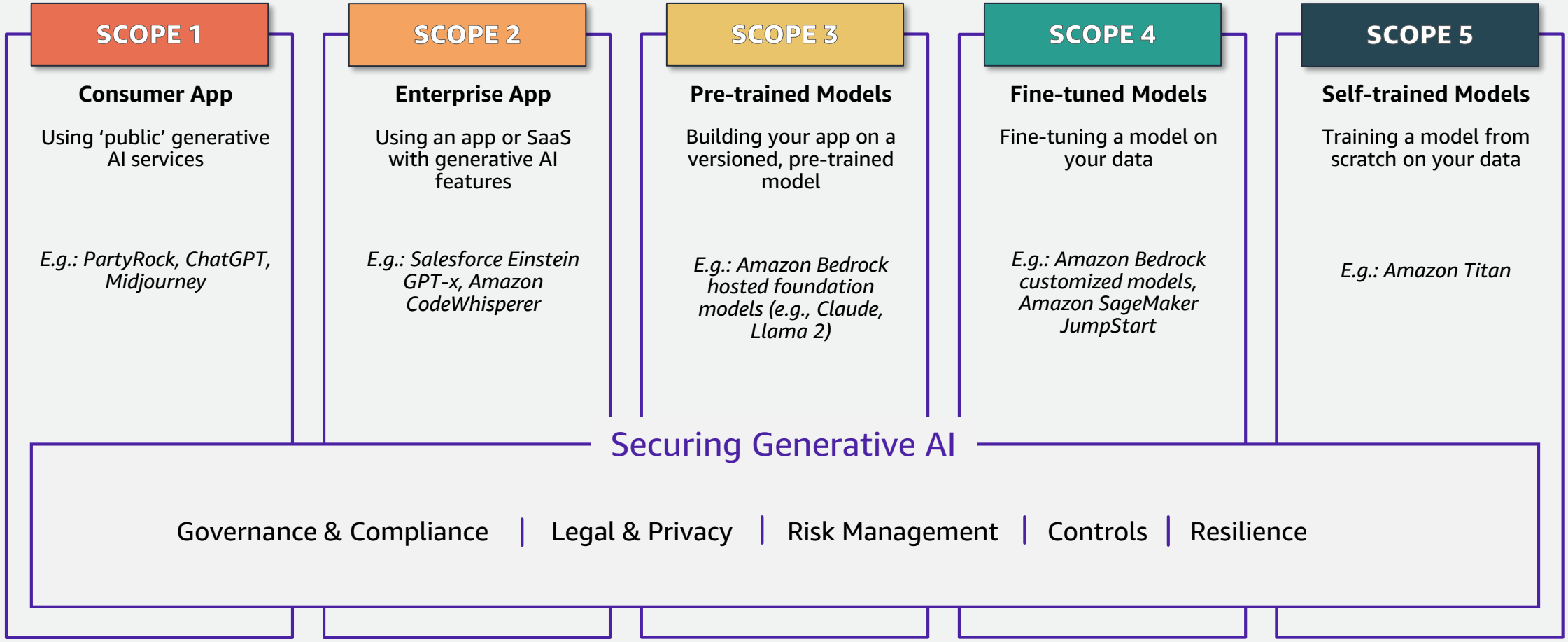
Fast-evolving technology with emerging best practices for risk identification, mitigation; still much to learn and to put into practice

Committed to continued rapid innovation while collaborating with government and industry to support safe, secure, and responsible AI

Lots of good additional detail here: https://aws.amazon.com/uki/cloud-services/uk-gov-ai-safety-summit/

Generative AI Security Scoping Matrix

A MENTAL MODEL TO CLASSIFY USE-CASES



Preliminary remarks

Today's focus is on foundation models (of all types), but lessons apply to all types of machine learning

Security is the focus, but themes and learnings are useful for broader responsible AI topics

"Red-teaming" often used in this space, but think more broadly in terms of general testing and adversarial testing

Transparency is important, thus the industry adoption of "service cards" or "model cards"

Prepare the Organization

Focus on integration of software engineering and data science

Define policies for data sourcing, wrangling, cleansing, quality control

Re-use/refine existing data classification policies and schemes

Take an end-to-end system approach; foundation model security and safety only part of the story

- Shared responsibility among data owners, model owners, fine-tuners, RAG designers, and application developers (as well as ML infrastructure provider(s))

Protect the Software(+)

Data focus: provenance/protection of data now first-class requirement

- Use traditional ML technologies to filter and classify input (and output) data

Separate duties and apply least privilege concepts and technologies must be refined and updated to include data and data lifecycle

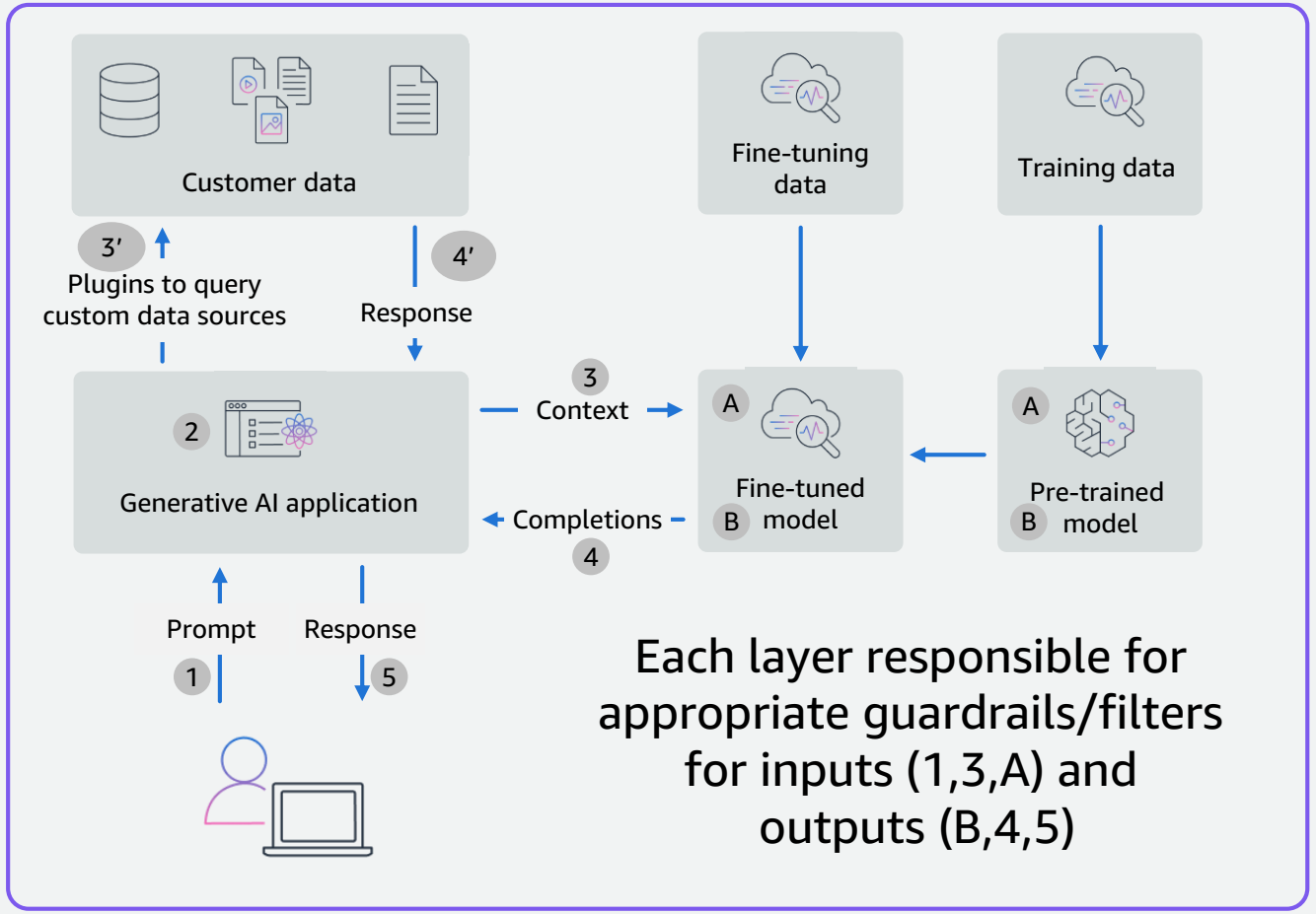
- E.g., model weights are sensitive data with no need for direct human access

Expand Secure DevOps models to “MLOps,” to include data science workstreams and security capabilities

Build in guardrails/filters and other kinds of input/output “circuit-breakers” as large models are hard to modify

Example data flows in a generative AI application

1. App receives input from user
    - [Optional] App queries data from custom data sources
2. App formats user input and customer data into a prompt
3. Prompt is completed by a model (fine-tuned or pre-trained)
4. Completion is processed by app
5. Response is sent to the user



Produce Well-Secured Software(+)

Securing both the software and data “supply chain” (lineage, provenance, confidentiality, integrity)

Testing at each stage (equivalent of integration and unit tests)

Structured, consistent, constantly improving use of internal and external pen-testing, automation; true “red-teaming” as well

Adversarial testing: use human experts but constantly increase automation

- E.g., Vary prompt attacks with logical equivalent of “fuzz testing”

Understand and accept today’s limitations of GenAI technology!

Respond to Vulnerabilities(+)

Well-tested programs and process for vulnerability (and other flaws) reporting (internal, external, bug bounty, etc.)

Large models too expensive to rebuild frequently, so…

- Rollback strategies for system updates that expose flaws
- Guardrails/filtering at each layer,[1] “disgorgement”[2] strategies

Incorporate learnings into next round of model building, as well as other layer updates

[1] *E.g.*, NeMo Guardrails, https://github.com/NVIDIA/NeMo-Guardrails; Bedrock Guardrails, https://aws.amazon.com/bedrock/guardrails/

[2] *E.g.*, “AI Model Disgorgement: Methods and Choices,” https://arxiv.org/pdf/2304.03545.pdf

NIST VIRTUAL WORKSHOP

Thank you

Mark Ryland

markry@amazon.com