International Journal of Engineering & Extended Technologies Research (IJEETR)

|ISSN: 2322-0163| www.ijeetr.com | A Bimonthly, Peer Reviewed, Scholarly Indexed Journal |

| Volume 6, Issue 2, March- April 2024 |

DOI:10.15662/IJEETR.2024.0602006

Adaptive Cryptographic Orchestration Against Learning-Enabled Adversaries

Sanjay Mishra

Engineering Manager, Swift Inc., Washington DC Metro Area, USA

sanjay.amu28@gmail.com

ABSTRACT: Artificial intelligence (AI) has transformed the cyber threat landscape by enabling adversaries to automate reconnaissance, optimize probing strategies, and adapt attacks using feedback. While modern cryptographic primitives remain mathematically secure under standard assumptions, real-world cryptographic deployments often expose operational signals—such as timing behavior, request patterns, error responses, and negotiation outcomes—that learning-enabled adversaries can exploit to improve attack efficiency.

This paper addresses the systems-level problem of **predictable cryptographic deployments enabling adversarial learning**. We propose **Adaptive Cryptographic Orchestration (ACO)**, a closed-loop defensive framework that dynamically reconfigures cryptographic and protocol parameters in response to telemetry-driven threat estimation, while preserving the use of standardized and vetted cryptographic primitives. We formalize attacker–defender interaction as a non-stationary learning environment, demonstrate why adaptive reconfiguration disrupts reinforcement-learning convergence, and present a simulation-based evaluation showing that ACO significantly reduces attacker learning efficiency under bounded operational overhead. ACO complements traditional cryptographic guarantees by reducing operational predictability in AI-accelerated attack settings.

KEYWORDS: Adaptive security, cryptographic orchestration, moving target defense, reinforcement learning, AI-enabled adversaries, non-stationary systems

I. INTRODUCTION

Cryptographic primitives such as AES-GCM, RSA, and elliptic-curve cryptography are designed to resist adversaries under well-defined computational assumptions. In practice, however, many security failures arise not from weaknesses in these primitives, but from **predictable deployment behavior** and **operational leakage**. Cryptographic services often expose timing characteristics, request patterns, retry behavior, error responses, and stable configuration choices that persist over long periods of operation.

Recent advances in artificial intelligence amplify the exploitation of such predictability. Learning-enabled adversaries can automate probing, correlate subtle signals across large datasets, and iteratively refine attack strategies using feedback. Reinforcement learning and large language model (LLM)-driven automation further accelerate this process by enabling adaptive exploration and large-scale orchestration. Static cryptographic configurations therefore create a **stationary environment** that facilitates adversarial learning and convergence toward effective probing strategies.

This paper addresses the following problem:
How can cryptographic deployments reduce adversarial learning efficiency and attack optimization without weakening cryptographic primitives or violating formal security assumptions?
We propose **Adaptive Cryptographic Orchestration (ACO)**, a systems-layer defense that treats cryptographic configuration as a dynamic, feedback-driven control problem. ACO continuously monitors telemetry, estimates threat levels, and adaptively reconfigures cryptographic and protocol parameters—such as key rotation cadence, cipher suite negotiation, rate limiting, and hardened error handling—within a vetted configuration space. By introducing controlled non-stationarity, ACO disrupts the learning assumptions of AI-enabled adversaries while preserving cryptographic correctness and operational stability.

International Journal of Engineering & Extended Technologies Research (IJEETR)

|ISSN: 2322-0163| www.ijeetr.com | A Bimonthly, Peer Reviewed, Scholarly Indexed Journal |

| Volume 6, Issue 2, March- April 2024 |

DOI:10.15662/IJEETR.2024.0602006

II. BACKGROUND AND RELATED WORK

Traditional cryptographic security focuses on mathematical hardness and formal proof models. These guarantees remain essential but assume idealized interfaces that abstract away deployment realities. In operational systems, cryptographic services are embedded within protocols, APIs, and distributed infrastructures that generate observable signals beyond ciphertext alone.

Machine learning has been extensively studied in offensive contexts, including side-channel analysis, traffic classification, and attack automation. Defensive applications of AI have largely focused on intrusion detection and anomaly detection rather than **runtime cryptographic adaptation**. Separately, moving target defense (MTD) techniques have been applied to networks, operating systems, and software diversity, demonstrating that controlled variability can increase attacker cost.

ACO bridges these domains by applying MTD principles **within cryptographic orchestration**, operating above the primitive layer. Unlike prior work, ACO explicitly models learning-enabled adversaries and uses adaptation to invalidate the stationarity assumptions required for efficient adversarial learning.

III. THREAT MODEL

3.1 Adversary Capabilities

The adversary can interact with cryptographic services through exposed interfaces such as APIs, protocol handshakes, or authentication endpoints. The adversary observes **operational outputs**, including:

- response timing and variance
- error codes and failure rates
- request throughput and rate-limit behavior
- negotiation outcomes and protocol responses

The adversary employs learning techniques to optimize probing strategies over time. Importantly, the adversary **does not** compromise cryptographic keys, internal memory, or hardware security modules, and does not violate the mathematical security assumptions of the underlying primitives.

3.2 Defender Capabilities and Assumptions

The defender controls cryptographic configuration parameters selected from a vetted, compliant configuration space. The defender can observe telemetry derived from service behavior and apply runtime reconfiguration subject to operational stability constraints.

IV. FORMAL MODEL OF ADAPTIVE CRYPTOGRAPHIC ORCHESTRATION (DUAL MODE)

This section formalizes Adaptive Cryptographic Orchestration (ACO) while providing intuitive explanations to clarify how the mathematical model maps to real systems and implementation.

4.1 Telemetry Observation

Mathematical definition

At discrete time step t, the cryptographic system observes a telemetry vector:

$$\boldsymbol{x}_t \in \mathbb{R}^d$$

x_t represents a collection of *d* observable operational measurements (e.g., request rate, failures, timing behavior) captured from the cryptographic system at time *t*.

Explanation

At every moment, the system collects **measurements about how cryptography is being used**. These are *not secrets*, but operational signals such as:

- number of encryption/decryption requests
- failed authentication or decryption attempts
- request burst patterns
- timing and latency statistics

International Journal of Engineering & Extended Technologies Research (IJEETR)

|ISSN: 2322-0163| www.ijeetr.com | A Bimonthly, Peer Reviewed, Scholarly Indexed Journal |

| Volume 6, Issue 2, March- April 2024 |

DOI:10.15662/IJEETR.2024.0602006

All these measurements are grouped into a single list (vector) called x_t.

Code intuition:

```
x_t = [
   request_rate,
   failure_ratio,
   avg_latency,
   retry_count
]
```

4.2 Threat Estimation Function

Mathematical definition

Telemetry is mapped to a scalar threat score using a detection function:

$$\boldsymbol{\theta}_t = \boldsymbol{g}(\boldsymbol{x}_t), where\ \mathbf{0} \leq \boldsymbol{\theta}_t \leq \mathbf{1}$$

where higher values indicate greater likelihood of adversarial activity.

Explanation

The system takes the collected telemetry and asks:
"Does this behavior look normal or suspicious?"
The function *g(·)* converts telemetry into a normalized threat score.
The function *g(·)* can be:

- an anomaly detector,
- a rules-based heuristic,
- or a machine-learning model.

The output is a **threat score**:

- $\theta_t \approx 0$ → **benign behaviour**
- $\theta_t \approx 1$ → **highly suspicious behaviour**

Code intuition (python)

```
def threat_detector(telemetry):
   return threat_score  # value between 0 and 1
```

4.3 Cryptographic Configuration Space

Mathematical definition

Let the system operate under a cryptographic configuration:

$$\boldsymbol{c}_t \in \boldsymbol{\mathcal{C}}$$

where \mathcal{C} is a finite set of vetted, standards-compliant configurations.

Explanation

The system does **not invent new cryptography**.
Instead, it switches between **pre-approved safe modes**, for example:

- **Baseline:** normal operation
- **Hardened:** faster key rotation, stricter rate limits
- **Max-Hardened:** strongest settings, strict negotiation, aggressive throttling

At any time, the system is in exactly **one** of these modes.

Code intuition (python)

```
C = ["BASELINE", "HARDENED", "MAX_HARDENED"]
current_config = "BASELINE"
```

4.4 Policy Controller (Decision Logic)

Mathematical definition

The next cryptographic configuration is selected by a defender policy:

$$\boldsymbol{c}_{t+1} = \boldsymbol{\pi}^{D}(\boldsymbol{c}_t, \boldsymbol{\theta}_t)$$

The policy π^D decides the next cryptographic configuration based on the current configuration and the estimated threat level.

Explanation

This equation simply states:
"Given the current configuration and the current threat level, decide what configuration to use next."

International Journal of Engineering & Extended Technologies Research (IJEETR)

|ISSN: 2322-0163| www.ijeetr.com | A Bimonthly, Peer Reviewed, Scholarly Indexed Journal |

| Volume 6, Issue 2, March- April 2024 |

DOI:10.15662/IJEETR.2024.0602006

Typical logic:
- low threat → stay in baseline mode
- medium threat → harden configuration
- high threat → maximum hardening

The policy is intentionally simple and explainable.

Code intuition (python)

```
def policy(current_config, threat):
    if threat < 0.3:
        return "BASELINE"
    elif threat < 0.6:
        return "HARDENED"
    else:
        return "MAX_HARDENED"
```

4.5 Threat Smoothing (Stability Mechanism)

Mathematical definition

To avoid reacting to short-lived anomalies, the threat score is smoothed:

$$\theta_t = \alpha \cdot \theta_t + (1 - \alpha) \cdot \theta_{t-1}$$

where $0 < \alpha \leq 1$ controls responsiveness.

Explanation

One unusual spike should **not** trigger drastic cryptographic changes.

Instead, the system:
- remembers past threat levels
- adjusts gradually

This prevents oscillations and unnecessary reconfiguration.

Code intuition (python)

```
smoothed_threat = alpha * current_threat + (1 - alpha) * previous_threat
```

4.6 Reconfiguration Cost and Stability Constraint

Mathematical definition

Let each configuration change incur a cost. Over a window W: (Over a rolling window of *W* steps, reconfiguration cost is bounded:)

$$\sum_{t=t}^{(t+W)} cost(c_t \rightarrow c_{t+1}) \leq B$$

where B is a stability budget.

Explanation

Changing cryptographic settings has overhead:
- CPU cost
- session renegotiation
- operational complexity

So the system enforces a rule:
"Do not reconfigure too frequently within a short time window."
This ensures ACO remains practical and deployable.

Code intuition (python)

```
if sum(recent_costs) > budget:
    keep_current_config()
```

4.7 Attacker Learning Model

Mathematical definition

The attacker is modeled as a learning agent maximizing cumulative reward:

$$max\ \mathbb{E}[\sum_{t=0}^{T} \gamma^t \cdot r_t]$$

where r_t represents information gain or attack progress.

International Journal of Engineering & Extended Technologies Research (IJEETR)

|ISSN: 2322-0163| www.ijeetr.com | A Bimonthly, Peer Reviewed, Scholarly Indexed Journal |

| Volume 6, Issue 2, March- April 2024 |

DOI:10.15662/IJEETR.2024.0602006

Explanation
The attacker tries to learn:
"Which probing strategy gives me the most useful feedback?"
This captures reinforcement learning, adaptive scripting, and automated probing behavior.

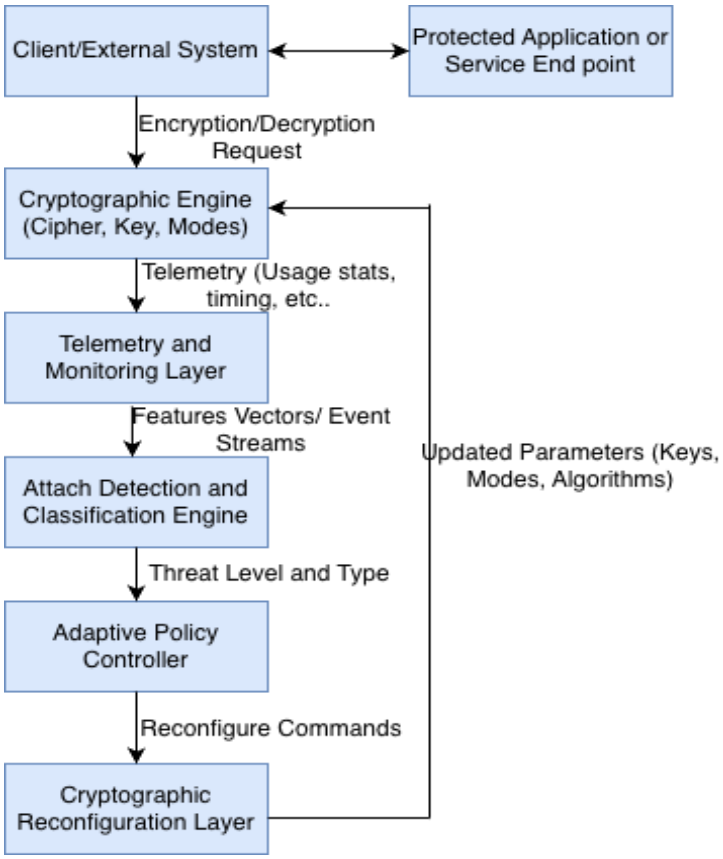
4.8 Why Adaptive Cryptography Works
Mathematical insight
In a static system:
P(r | a) = constant
In an adaptive system:
P(r | a, c_t)
Since c_t changes over time, the environment becomes **non-stationary**.
Explanation
If cryptographic behavior never changes:
- attackers learn what works
- attacks become optimized and automated

If cryptographic behavior *does* change:
- yesterday's successful strategy may fail today
- learning becomes unstable and slower

This disruption of attacker learning is the **core security benefit** of ACO.

V. ADAPTIVE CRYPTOGRAPHIC ORCHESTRATION ARCHITECTURE

ACO consists of four components:
1. **Telemetry Collection:** gathers operational metrics without exposing sensitive data.
2. **Threat Detection:** maps telemetry to a threat score.
3. **Policy Controller:** selects defensive actions based on threat level and stability constraints.
4. **Reconfiguration Engine:** safely applies configuration changes without weakening cryptographic guarantees.



International Journal of Engineering & Extended Technologies Research (IJEETR)

|ISSN: 2322-0163| www.ijeetr.com | A Bimonthly, Peer Reviewed, Scholarly Indexed Journal |

| Volume 6, Issue 2, March- April 2024 |

DOI:10.15662/IJEETR.2024.0602006

Adaptive Cryptographic Orchestration (ACO) as a closed-loop control system. Telemetry from cryptographic operations is analyzed to estimate threat levels, which drive policy-based cryptographic reconfiguration, introducing controlled non-stationarity that disrupts attacker learning.

ACO operates strictly above the primitive layer, ensuring compatibility with existing standards and compliance requirements.

VI. EXPERIMENTAL EVALUATION

We evaluate ACO using a controlled simulation in which the attacker is modeled as a learning agent and the defender operates under either static or adaptive configurations.

6.1 Methodology

The attacker selects probing strategies to maximize expected reward. The defender either maintains a fixed configuration or applies ACO. Metrics include cumulative attacker reward, learning convergence behavior, and reconfiguration cost.

6.2 Results

Simulation results show that static configurations enable rapid attacker convergence and high cumulative reward. In contrast, ACO introduces controlled non-stationarity that significantly reduces attacker learning efficiency while maintaining bounded operational overhead.

To support reproducibility, we provide a reference implementation of the simulation environment, adaptive controller, and figure-generation scripts used in this evaluation in a public repository
https://github.com/sanjay-amu/adaptive-crypto-orchestration/

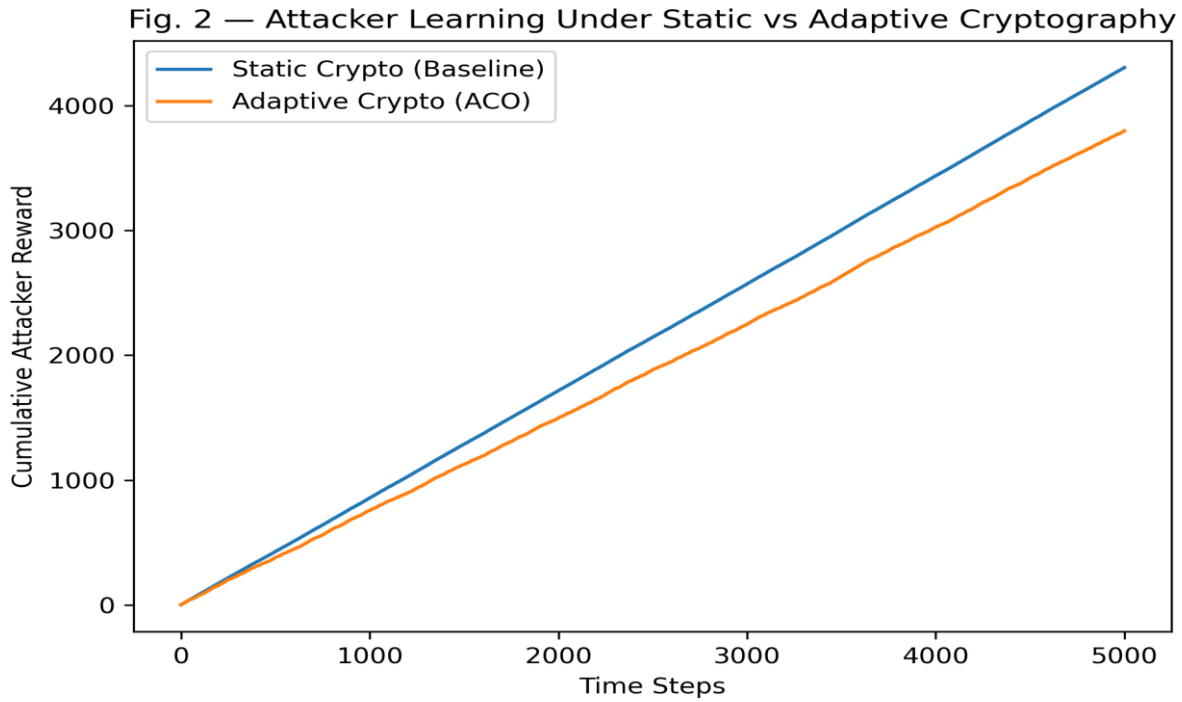


Figure 2 compares cumulative attacker reward under a static cryptographic configuration and under Adaptive Cryptographic Orchestration (ACO). In the static case, attacker reward grows nearly linearly, reflecting rapid convergence toward an optimized probing strategy. In contrast, ACO consistently reduces the attacker's reward accumulation rate, resulting in a widening performance gap over time. This behavior indicates that adaptive reconfiguration introduces non-stationarity that degrades adversarial learning efficiency without eliminating system functionality.

|ISSN: 2322-0163| www.ijeetr.com | A Bimonthly, Peer Reviewed, Scholarly Indexed Journal |

| Volume 6, Issue 2, March- April 2024 |

DOI:10.15662/IJEETR.2024.0602006

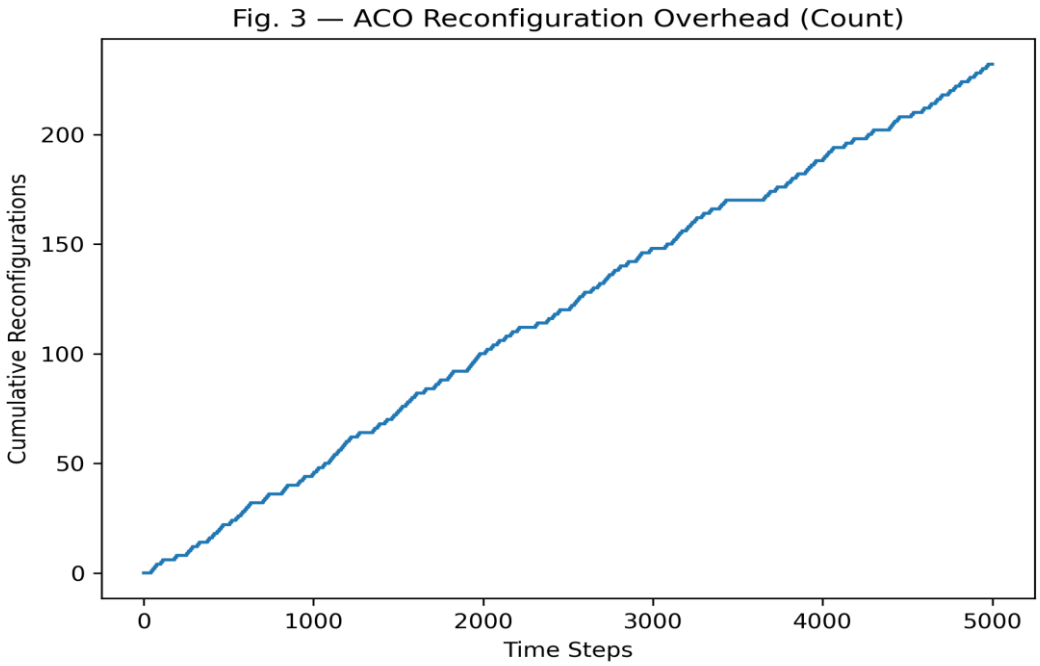


Figure 3 shows the cumulative number of cryptographic reconfigurations performed by ACO. Reconfiguration events occur infrequently relative to the total number of time steps, demonstrating that adaptation is selective and bounded. This result confirms that ACO achieves learning disruption with manageable operational overhead, avoiding excessive configuration thrashing.

6.3 Security Guarantees and Non-Goals

Adaptive Cryptographic Orchestration (ACO) does not alter cryptographic primitives or their formal security guarantees. The framework operates strictly at the orchestration and configuration layer, leveraging vetted cryptographic algorithms and standards-compliant parameters. ACO is not intended to prevent key compromise, implementation-level vulnerabilities, or cryptanalytic breakthroughs. Instead, its objective is to reduce operational predictability and adversarial learning efficiency in cryptographic deployments subject to AI-enabled probing and automation.

VII. DISCUSSION AND LIMITATIONS

ACO does not replace cryptographic proofs or protect against key compromise, insider threats, or implementation vulnerabilities. Its effectiveness depends on telemetry quality and appropriate policy tuning. Adaptive adversaries may attempt to model reconfiguration behavior, motivating future work on randomized and game-theoretic policies.

VIII. CONCLUSION

This paper demonstrates that **adaptive cryptographic orchestration** can reduce the effectiveness of learning-enabled adversaries by eliminating the stationarity that enables attack optimization. By combining formal modeling, algorithmic design, and simulation-based evaluation, we show that adaptation at the systems layer complements traditional cryptography without weakening its guarantees. As AI continues to accelerate cyber attacks, adaptive orchestration provides a practical and principled path toward more resilient cryptographic deployments.

REFERENCES

[1] E. Biham and A. Shamir, “Differential cryptanalysis of DES-like cryptosystems,” *Journal of Cryptology*, vol. 4, no. 1, pp. 3–72, 1991.

[2] R. Anderson and M. Kuhn, “Tamper resistance — a cautionary note,” in *Proc. 2nd USENIX Workshop on Electronic Commerce*, 1996.

|ISSN: 2322-0163| www.ijeetr.com | A Bimonthly, Peer Reviewed, Scholarly Indexed Journal |

| Volume 6, Issue 2, March- April 2024 |

DOI:10.15662/IJEETR.2024.0602006

[3] Y. Yarom and K. Falkner, "Flush+Reload: A high resolution, low noise, L3 cache side-channel attack," in *Proc. USENIX Security Symposium*, 2014.
[4] M. M. Alani, "Machine learning for cryptanalysis: A survey," *Journal of Information Security*, vol. 11, no. 3, pp. 143–165, 2020.
[5] Z. Liu, L. Zhang, and J. Yang, "Machine learning-based side-channel attacks," *IEEE Transactions on Computers*, vol. 66, no. 9, pp. 1489–1502, 2017.
[6] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. Cambridge, MA, USA: MIT Press, 2016.
[7] N. Papernot et al., "The limitations of deep learning in adversarial settings," in *Proc. IEEE European Symposium on Security and Privacy (EuroS&P)*, 2016.
[8] S. Suresh et al., "Reinforcement learning for cybersecurity: A survey," *ACM Computing Surveys*, vol. 55, no. 3, pp. 1–38, 2022.
[9] P. Okhravi et al., "Survey of cyber moving target defenses," *IEEE Security & Privacy*, vol. 12, no. 2, pp. 72–83, 2014.
[10] S. Jajodia et al., *Moving Target Defense: Creating Asymmetric Uncertainty for Cyber Threats*. New York, NY, USA: Springer, 2011.
[11] E. Even-Dar, S. Mannor, and Y. Mansour, "Online learning in non-stationary environments," in *Proc. Conference on Learning Theory (COLT)*, 2006.
[12] C. Hartland et al., "Change detection for multi-armed bandit problems," in *Proc. European Conference on Machine Learning*, 2006.
[13] R. Anderson, *Security Engineering: A Guide to Building Dependable Distributed Systems*, 2nd ed. Hoboken, NJ, USA: Wiley, 2008.