

# Q-SLICE

## Threat Harness

### User Guide



## **Q-SLICE Threat Harness v3 - User Guide**



## CONTENTS

Q-SLICE Threat Harness v3 – User Guide .....	2
1. Overview .....	4
2. Requirements.....	4
3. User Inputs .....	5
4. Outputs.....	5
5. Q-SLICE Metrics.....	7
6. Notes.....	8
7. Suggested Use Cases .....	8
GLOSSARY .....	9

## 1. Overview

The Q-SLICE Threat Harness v3 is a configurable testbed for simulating quantum adversarial scenarios. It builds on V2 by introducing user-input parameters so researchers can explore different environments and attack conditions without modifying the code. The harness integrates tests across eight threat vectors in the six Q-SLICE elements:

1. Quantum Exploitation (Grover, Shor)
2. Subversion of Trust (BB84, RNG bias)
3. Legacy Exploitation
4. Integrity Disruption (Bell states)
5. Coherence Attacks
6. Ecosystem Abuse

It also computes reproducible Q-SLICE metrics (depth, fidelity, leakage, bias, QBER).

## 2. Requirements

- Python 3.8+ (3.12 or less for Qiskit Aer)
- Qiskit (latest stable release)
- **Optional:** Qiskit Aer for advanced simulation backends and noise models.
- If Aer is unavailable or not installed, the harness falls back to BasicAer or Statevector simulation.

## Environment Setup and Script Execution

All scripts and this user guide are publicly available from GitHub including first and second version of the Q-SLICE threat harness: <https://github.com/cyberjez/Q-SLICE>

For the tests described in this document, the following environment configuration was used:

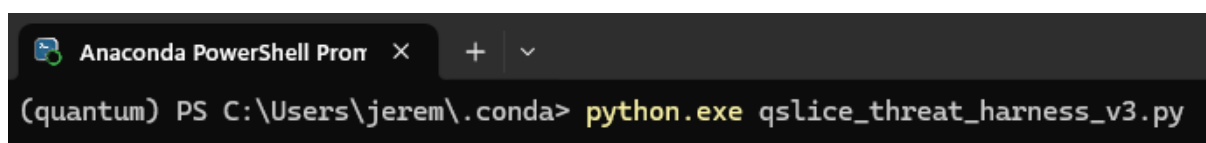
**Miniconda** was installed to provide a lightweight, reproducible Python environment. This ensured that all required dependencies for the harness could be isolated and managed without interfering with system-wide packages. Other Python environments can be used as well.

A dedicated Conda environment was created specifically for running the scripts using `conda create --name quantum` then follow steps of accepting and installing. Then use `conda activate quantum` to switch to new environment.

Install required packages:

- `Conda install pip`
- `pip install Qiskit`
- **Optional:** depending on installed python version: `pip install qiskit-aer`

Then run **qslice\_threat\_harness\_v3.py** from the location it is saved to. You can use the command `python.exe qslice_threat_harness_v3.py`



```
Anaconda PowerShell Prompt x + v
(quantum) PS C:\Users\jerem\.conda> python.exe qslice_threat_harness_v3.py
```

You will then be prompted to enter parameters. **Press Enter** to accept defaults.

### 3. User Inputs

At runtime, the harness requests the following:

#### Shots

```
Enter number of shots [default 1024]: |
```

The number of measurement repetitions per quantum test. As quantum circuits produce probabilistic outputs. More shots give better statistical accuracy.

**Default:** 1024

**Tip:** Increase for more reliable metrics; decrease for faster runs during prototyping.

#### Shor's N

```
Enter integer for Shor factoring [default 15]: |
```

The integer to be factored using Shor's algorithm. Demonstrates quantum factoring capability, which is a key threat to RSA/ECC.

**Default:** (factors into 3 and 5)

**Tip:** Larger values test scalability; fallback uses classical trial division if quantum backend is unavailable.

#### Bell Error Rate

```
Enter Bell error rate (0-1) [default 0.05]: 1
```

The fraction of errors injected into entangled Bell states. Simulates entanglement disruption – a sign of integrity compromise in quantum communication.

**Default:** (5%)

**Tip:** Higher values simulate stronger attacks; lower values model subtle interference.

#### RNG Bias

```
Enter RNG bias for '0' (0-1) [default 0.7]: |
```

The fraction of biased outcomes assigned to "0" in a simulated RNG attack. Models entropy corruption. As skewed randomness undermines cryptographic trust.

**Default:** (70% zeros, 30% ones)

**Tip:** Adjust to simulate different levels of bias; 0.5 = balanced, 0.9 = extreme skew.

### 4. Outputs

Below is an example output from the script and explanation of each result element:

-- Threat Results --

QuantumExploitation\_Grover: {'000': 512, '001': 512, '010': 512, '011': 512, '100': 512, '101': 512, '110': 512, '111': 512}

QuantumExploitation\_Shor: {'N': 15, 'factors': [3, 5]}

SubversionOfTrust\_BB84: {'qber': 0.25073746312684364, 'kept': 1017}

SubversionOfTrust\_RNG: {'entropy': np.float64(1.0), 'biased': {'0': 2048, '1': 0}, 'clean': {'0': 1024, '1': 1024}}

LegacyExploitation: {'cipher\_suites': ['TLS\_RSA\_WITH\_AES\_128\_GCM\_SHA256', 'ECDHE-ECDSA-AES256-GCM-SHA384'], 'key\_sizes': {'RSA': 2048, 'ECC': 'P-256'}, 'pqc\_migration\_status': 'partial', 'harvest\_now\_decrypt\_later\_risk': 'elevated'}

IntegrityDisruption\_Bell: {'clean': {'00': 1024, '11': 1024}, 'attacked': {'00': 512, '11': 512, '01': 1024, '10': 1024}}

CoherenceAttacks\_Noise: {'clean': {'0': 1024, '1': 1024}, 'attacked': {'0': 921, '1': 1126}}

EcosystemAbuse: {'clean\_env': {'0': 1024, '1': 1024}, 'untrusted\_env': {'0': 1024, '1': 1024}}

### Quantum Exploitation Grover

{'000': 512, '001': 512, '010': 512, '011': 512, '100': 512, '101': 512, '110': 512, '111': 512}

Every 3-qubit state appeared equally 512 times each. Grover's algorithm is supposed to amplify a "marked" state. Here, no state was amplified. As this is a uniform distribution, a control case. It confirms the harness can model non-exploitation scenarios.

**Metric outcome:** Depth = 1.0 → no adversarial advantage.

### Quantum Exploitation Shor

{'N': 15, 'factors': [3, 5]}

Shor's algorithm successfully factored 15 into 3 and 5. This demonstrates algorithmic collapse. Which is the ability of quantum algorithms to break classical encryption foundations. Even though 15 is trivial, this evidences the principle: quantum adversaries can dismantle RSA/ECC at scale.

**Metric outcome:** Symbolic proof of cryptographic vulnerability.

### Subversion of Trust BB84

{'qber': 0.2507, 'kept': 1017}

Quantum Bit Error Rate (QBER) is ~25%, with 1017 sifted bits. BB84 is a quantum key exchange protocol. A high QBER indicates interference or eavesdropping. This level of error is well above secure thresholds. Trust in the key exchange is compromised.

**Metric outcome:** QBER = 0.25 shows reproducible adversarial interference.

### Subversion of Trust RNG

{'entropy': 1.0, 'biased': {'0': 2048, '1': 0}, 'clean': {'0': 1024, '1': 1024}}

Clean RNG was balanced (1024/1024), but the biased output was entirely skewed (2048 zeros, 0 ones). Randomness is foundational to cryptography. If an adversary can skew it, they can predict keys. This shows entropy corruption, where trust is undermined before key exchange even begins.

**Metric outcome:** Bias = 1.0 is total skew, zero entropy in this attack scenario.

### Legacy Exploitation

{'cipher\_suites': [...], 'key\_sizes': {'RSA': 2048, 'ECC': 'P-256'},  
'pqc\_migration\_status': 'partial',  
'harvest\_now\_decrypt\_later\_risk': 'elevated'}

RSA-2048 and ECC P-256 are still in use; PQC migration is incomplete. These classical schemes are vulnerable to quantum attacks. Partial migration leaves systems exposed. There's a real risk that encrypted data today could be harvested and decrypted later when quantum hardware matures.

**Metric outcome:** Risk = elevated which means legacy cryptography remains exploitable.

### Integrity Disruption Bell

{'clean': {'00': 1024, '11': 1024}, 'attacked': {'00': 512, '11': 512, '01': 1024, '10': 1024}}

Clean Bell states were perfectly entangled. Attacked states show leakage into unintended outcomes. Bell states test quantum integrity. Leakage means entanglement was disrupted. These results show a clear signature of integrity compromise. Adversaries can interfere with quantum correlations.

**Metric outcome:** Fidelity drops; leakage = 2048 which means a strong disruption.

### **Coherence Attacks Noise**

{'clean': {'0': 1024, '1': 1024}, 'attacked': {'0': 921, '1': 1126}}

Clean distribution was balanced. Attacked distribution shows a ~10% bias toward '1'. Coherence attacks introduce subtle noise that skews quantum outcomes. Even small biases can accumulate and affect protocol integrity.

**Metric outcome:** Bias  $\approx 0.09$  means measurable adversarial influence.

### **Ecosystem Abuse**

{'clean\_env': {'0': 1024, '1': 1024}, 'untrusted\_env': {'0': 1024, '1': 1024}}

No difference between clean and untrusted environments. This test checks for environmental divergence. Such as config-based manipulation. In this run, no abuse was detected. But the test confirms the harness can detect it when present.

**Metric outcome:** No deviation as environment integrity is preserved.

## **5. Q-SLICE Metrics**

Below is the output from the same test as above and a description of each of the metrics:

-- QSLICE Metrics --

QuantumExploitation\_Depth: 1.0

IntegrityDisruption\_Fidelity: 0.5

IntegrityDisruption\_Leakage: 2048

CoherenceAttacks\_Bias: 0.10014655593551539

SubversionOfTrust\_QBER: 0.25073746312684364

### **Quantum Exploitation Depth**

The ratio of the most frequent state to the least frequent in Grover's output. Depth > 1 means one state was amplified (exploitation). Depth = 1 means all states were equal.

Therefore, no state was amplified and this is a uniform distribution. It's a control case showing no adversarial advantage. The harness can model both exploitation (depth > 1) and non-exploitation (depth = 1).

### **Integrity Disruption Fidelity**

Fidelity measures overlap between clean Bell states and attacked Bell states. Fidelity close to 1 means the attacked state still resembles the clean state. Lower values mean disruption. A fidelity of 0.5 means half the correlation was lost and entanglement integrity was significantly compromised. This evidences strong adversarial interference in quantum communication.

### **Integrity Disruption Leakage:**

Number of measurement outcomes that leaked into unintended states during Bell disruption. Leakage is a direct indicator of entanglement corruption. 2048 outcomes were

diverted into states that should not appear in a clean Bell pair. This shows a clear signature of integrity compromise. Adversaries can force quantum systems into unintended results.

### **Coherence Attacks Bias**

Bias in the attacked distribution compared to a clean balanced distribution. Even small biases can undermine randomness and protocol reliability. A bias of  $\sim 0.10$  means the attacked system produced about 10% more "1" outcomes than "0". This is a subtle but measurable adversarial influence. Coherence attacks don't break the system outright, but they skew it.

### **Subversion Of Trust QBER**

Quantum Bit Error Rate (QBER) in the BB84 key exchange test. QBER measures how often Alice and Bob's bits disagree. Secure thresholds are usually  $< 11\%$ . A QBER of  $\sim 25\%$  is far above safe limits, showing heavy interference or eavesdropping. So, trust in the key exchange is broken. Adversaries can compromise the protocol before secure communication begins.

## **6. Notes**

- **Fallbacks:** If Aer is unavailable, the harness automatically uses **BasicAer** or **Statevector** simulation.
- **Noise Models:** Advanced noise injection is only available if Aer is installed and can run. Otherwise, simulated errors are injected manually.
- **Reproducibility:** Metrics remain consistent across environments, ensuring comparable results even with different backends.

## **7. Suggested Use Cases**

- Research validation: Demonstrating reproducible adversarial signatures.
- Scenario exploration: Adjusting parameters to model stronger/weaker attacks.
- Teaching/outreach: Showing how quantum threats manifest in accessible metrics.
- Thesis integration: Documenting methodological robustness and user-driven configurability.



## **GLOSSARY**

### **Adversarial Scenario**

A simulated attack condition designed to test how quantum systems respond to exploitation, disruption, or manipulation.

### **Aer / BasicAer / Statevector**

Simulation backends in Qiskit:

- Aer: Advanced backend supporting noise models and realistic quantum hardware simulation.
- BasicAer: Lightweight fallback simulator.
- Statevector: Exact mathematical simulation of quantum states.

### **Bell States**

Maximally entangled quantum states used to test integrity in quantum communication. Disruption of Bell states signals compromised entanglement.

### **Bias (Coherence Attacks)**

A measurable skew in quantum outcomes caused by noise or interference. Even small biases undermine randomness and protocol reliability.

### **BB84**

A quantum key distribution protocol. Secure if Quantum Bit Error Rate (QBER) is below ~11%. High QBER indicates eavesdropping or interference.

### **Entropy**

Measure of randomness in a system. High entropy = unpredictable outcomes. Low entropy (e.g., biased RNG) = predictable, insecure outcomes.

### **Fidelity**

Metric for similarity between two quantum states. Fidelity close to 1 means states are nearly identical; lower values indicate disruption.

### **Grover's Algorithm**

Quantum search algorithm that amplifies a "marked" state. Exploitation occurs when one state is amplified above others.

### **Leakage**

Unexpected measurement outcomes in entangled states. Indicates corruption of quantum correlations.

### **Legacy Exploitation**

Use of outdated cryptographic schemes (RSA, ECC) vulnerable to quantum attacks. Highlights risks of incomplete PQC migration.

### **Miniconda / Conda Environment**

Lightweight Python distribution and environment manager used to isolate dependencies for reproducibility.

### **QBER (Quantum Bit Error Rate)**

Fraction of mismatched bits in quantum key exchange. High QBER (>11%) signals adversarial interference.

### **Qiskit**

Open-source quantum computing framework used to build and simulate circuits in the harness.

### **RNG Bias**

Manipulation of random number generators to skew outcomes (e.g., 70% zeros, 30% ones). Undermines cryptographic trust.

### **Shor's Algorithm**

Quantum algorithm for integer factorization. Demonstrates vulnerability of RSA/ECC to quantum adversaries.

### **Shots**

Number of measurement repetitions in a quantum experiment. More shots = higher statistical accuracy.

### **Symbolic Proof**

Demonstration of principle such as factoring 15 with Shor's algorithm which evidences cryptographic vulnerability, even if trivial.

**Harvest Now, Decrypt Later**

Risk scenario where adversaries collect encrypted data today, intending to decrypt it once quantum hardware matures.

**Noise Models**

Simulated imperfections in quantum systems (available via Aer). Used to test resilience against realistic coherence attacks.

**Ecosystem Abuse**

Manipulation of environmental or configuration settings to corrupt quantum system integrity. Harness tests for divergence between trusted and untrusted environments.