Members:

Μεθοδιος Ζαχαριουδακης 4384
Κλεομένης Ρουσιάς 4099
Μαρίνα Βασσάλου 4138

Summary of Part 1

This chapter focuses on the evolving landscape of cryptography in preparation for quantum computing and how in order to keep confidentiality organisations need to move toward finding quantum-safe cryptography.
The key driver is the rapid development of quantum computing, which poses a significant threat to current cryptographic systems as well as current public-key algorithms.

Key Terms and Concepts

1. **Harvest Now, Decrypt Later (HNDL)**
   A strategy used to capture and store encrypted data today, with the intention of decrypting it once quantum computing resources are capable of breaking current cryptographic algorithms.Since there is the potential that encrypted data today will be usable/exploitable in the future.

2. **Cryptographic Agility**
   The capability of a system to be able switch between cryptographic algorithms with minimal changes.This system ability ensures organizations can respond swiftly to new threats without needing to remake algorithms or make them from scratch, instead they can replace part of them and make them reusable.

3. **Cryptographic Primitive**
   The fundamental building blocks of cryptographic protocols.They form the basis of security services such as secure communications, digital signatures, and secure key exchange.

4. **Post-Quantum Cryptography**
   Cryptographic algorithms designed to protect against both present (current) and future (quantum) attacks.
   Some current public-key cryptographic algorithms are vulnerable to quantum computers.

5. **Information Security Management**
   An approach to protecting organizational information assets using policies, procedures, and technologies.
   Security management frameworks help integrate new cryptographic standards smoothly while maintaining the current structure as well as the security of it.

6. **Interoperability**
   The ability of different systems, technologies, and organizations to work together. With a goal of making systems that can support and integrate new quantum-safe algorithms without causing incompatibilities in existing infrastructure.

7. **Cybersecurity Compliance**
   Adhering to legal, regulatory, and industry requirements designed to protect data integrity and privacy.
   Future regulatory mandates may require or strongly encourage quantum-safe cryptographic solutions to safeguard data confidentiality in the long term.

8. **Technology Roadmap**
   A plan outlining the adoption of new technologies and the steps needed to integrate them.
   In the PQC realm, a roadmap typically includes identifying vulnerable systems, testing new algorithms, and setting timelines for phased rollouts of quantum-safe solutions.

9. **NIST Guidelines**
   - Inventory & Assessment: Identify all uses of cryptography in your organization.
   - Priority: Develop a plan to migrate from quantum-vulnerable algorithms.
   - Build Agility: Implement architectures that can accommodate both current and future cryptographic algorithms.
   - Test & Validate: Ensure new algorithms meet performance and security requirements.

10. **European Guidelines**
    A coordinated approach across EU member states for adopting post-quantum cryptography, emphasizing collaboration, risk assessment, and maintaining interoperability.

11. **PQC Migration Handbook**
    Offers detailed, practical steps and best practices for transitioning to PQC, including case studies, stakeholder guidance, and real-world migration scenarios.

3. Findings

1. **Urgency of Action**
   Quantum threats may not be imminent, but data harvested now can be exposed later. Organizations should **plan migrations proactively** rather than wait for fully realized quantum attacks.
2. **Adoption of Crypto-Agile Systems**
   Cryptographic agility is paramount for smooth migrations to new algorithms and for mitigating potential security vulnerabilities arising from sudden cryptographic breakthroughs.

3. **Importance of Compliance and Standards**
   Evolving regulations and industry best practices will increasingly recognize and require quantum-safe solutions. Staying current with NIST and European Commission guidelines ensures readiness.

4. **Collaboration and Roadmapping**
   Comprehensive roadmaps guide organizations through the transition process, including thorough inventory, testing, and phased adoption of new algorithms.
5. **Continuous Monitoring and Improvement**
   Post-quantum cryptography is an evolving field. Organizations need iterative assessments of cryptographic health and must remain flexible as new standards or vulnerabilities emerge.

4. Conclusion

Part 1 underscores the growing necessity for a well-planned transition to post-quantum cryptography. Key concepts such as *Harvest Now, Decrypt Later*, *Cryptographic Agility*, and *PQC* lay the groundwork for understanding why this transition cannot be delayed.

Meanwhile, guidelines from **NIST**, the **European Commission**, and detailed advice from the **PQC Migration Handbook** provide actionable steps to ensure organizations maintain data security and compliance in the face of evolving quantum threats.

This foundation sets the stage for subsequent chapters, where we will delve into the technical, organizational, and regulatory implications of implementing post-quantum cryptography solutions.

Summary of Part 2

1. Introduction

The Cryptographic Asset Inventory Tool helps identify and manage cryptographic vulnerabilities in source code. It scans folders, detects weak cryptographic primitives, and evaluates their severity using a GUI and database.

2. Key Features

2.1 Folder Scanning

Scans all files in a folder and subdirectories, supporting Python, Java, C/C++, and more.

2.2 Vulnerability Detection

Detects:

- **MD5, SHA1**: High severity (outdated, insecure).
- **DES, 3DES**: High/Medium severity (weak or deprecated).
- **AES-128**: Low severity (not quantum-safe).
- **Hardcoded Keys**: Medium severity (exposure risk).

2.3 Risk Assessment

Findings are ranked by severity:

- **High**: Critical, immediate action needed.
- **Medium**: Moderate, address soon.
- **Low**: Non-critical, recommended improvements.

2.4 Results Management

Results are displayed in a sortable table and stored in a SQLite database (`crypto_inventory.db`).

2.5 Reporting and Statistics

Includes:

- Files scanned and vulnerable files detected (categorized by file type).
- Explanation of risk levels.
- Resettable results.

3. Tool Usage

3.1 Starting the Tool

1. Save the script as `CryptoInventoryTool.py`.

Run the script with:
python CryptoInventoryTool.py

2.

3.2 Scanning a Folder

1. Use **Browse** to select a folder.
2. Click **Scan** to start scanning.

3.3 Viewing Results

- Results table displays file path, issue, and severity.
- Sort results by column headers.

3.4 Printing Statistics

- Click **Print Statistics** to view:
    - Total files scanned.
    - Vulnerable files found by type.

3.5 Clearing Findings

- Click **Clear Findings** to reset results.

4. Future Enhancements

1. **Advanced Pattern Matching**: Use regular expressions for complex vulnerabilities like hardcoded keys.
2. **Expanded File Type Support**: Add support for more programming languages.
3. **Customizable Best Practices**: Allow users to define/update weak cryptographic primitives.
4. **Report Export**: Generate PDF or CSV reports summarizing scan results.

5. Conclusion

The tool efficiently identifies and prioritizes cryptographic vulnerabilities, providing actionable insights to improve code security.

Summary of Part 3

Migration Plan for TechServe Ltd.

Step 1: Assessment

1. **Inventory Existing Cryptographic Systems**:
    - Email encryption: Uses **TLS 1.0** (deprecated).
    - Internal file encryption: Relies on **AES-128** with weak key rotation practices.
    - User authentication: SHA-1 for password hashing.
    - VPN encryption: Uses outdated **IPsec with DES**.
2. **Vulnerability Assessment**:
    - **TLS 1.0** is vulnerable to protocol downgrade attacks.
    - **AES-128** has no immediate vulnerabilities but needs stronger key rotation policies and should transition to AES-256 for long-term security.
    - **SHA-1** is prone to collision attacks, making password hashes susceptible to compromise.
    - **DES in VPN** is weak and easily crackable.

Step 2: Prioritization

Criteria:

- **Impact**: A compromised VPN could expose client data in transit.
- **Compliance**: GDPR mandates strong encryption for sensitive customer data.
- **Criticality**: Password security is essential for all internal and customer-facing systems.

Ranked List:

1. Replace VPN encryption (**High Priority**).
2. Transition from SHA-1 to SHA-256 for password hashing (**High Priority**).
3. Upgrade TLS from 1.0 to 1.2 or higher (**Medium Priority**).
4. Transition to AES-256 for file encryption (**Low Priority**).

Step 3: Implementation Planning

1. **Define Replacements**:
    - VPN: Move from IPsec with DES to IPsec with AES-256.
    - Authentication: Replace SHA-1 with SHA-256
    - TLS: Upgrade to TLS 1.2.
    - File encryption: Transition to AES-256 with periodic key rotation.
2. **Phased Plan**:
    - Phase 1: Update the VPN encryption for remote workers.
    - Phase 2: Implement new password hashing algorithm for internal systems.
    - Phase 3: Upgrade TLS protocols on email servers.
    - Phase 4: Enhance file encryption policies.

Step 4: Testing

1. Crete python files for the replacements
    a. IPsec with AES-256
    b. SHA-256
    c. TLS 1.2
    d. AES-256
2. Run to tool to scan vulnerabilities
3. Found none

Step 5: Deployment

1. **Phase 1: VPN Update**:
    - Schedule downtime for VPN maintenance during non-peak hours.
    - Roll out IPsec with AES-256 to all remote workers and partners.
2. **Phase 2: Password Hashing Update**:
    - Migrate existing SHA-1 hashes to SHA-256 using a secure re-hashing process.
    - Notify users of the change and enforce strong password policies.
3. **Phase 3: Email Server Upgrade**:

    - Switch from TLS 1.0 to 1.2/1.3 with fallback support for older clients.
    - Conduct test email deliveries before full deployment.
4. **Phase 4: File Encryption**:
    - Transition files to AES-256 encryption and implement key rotation policies.

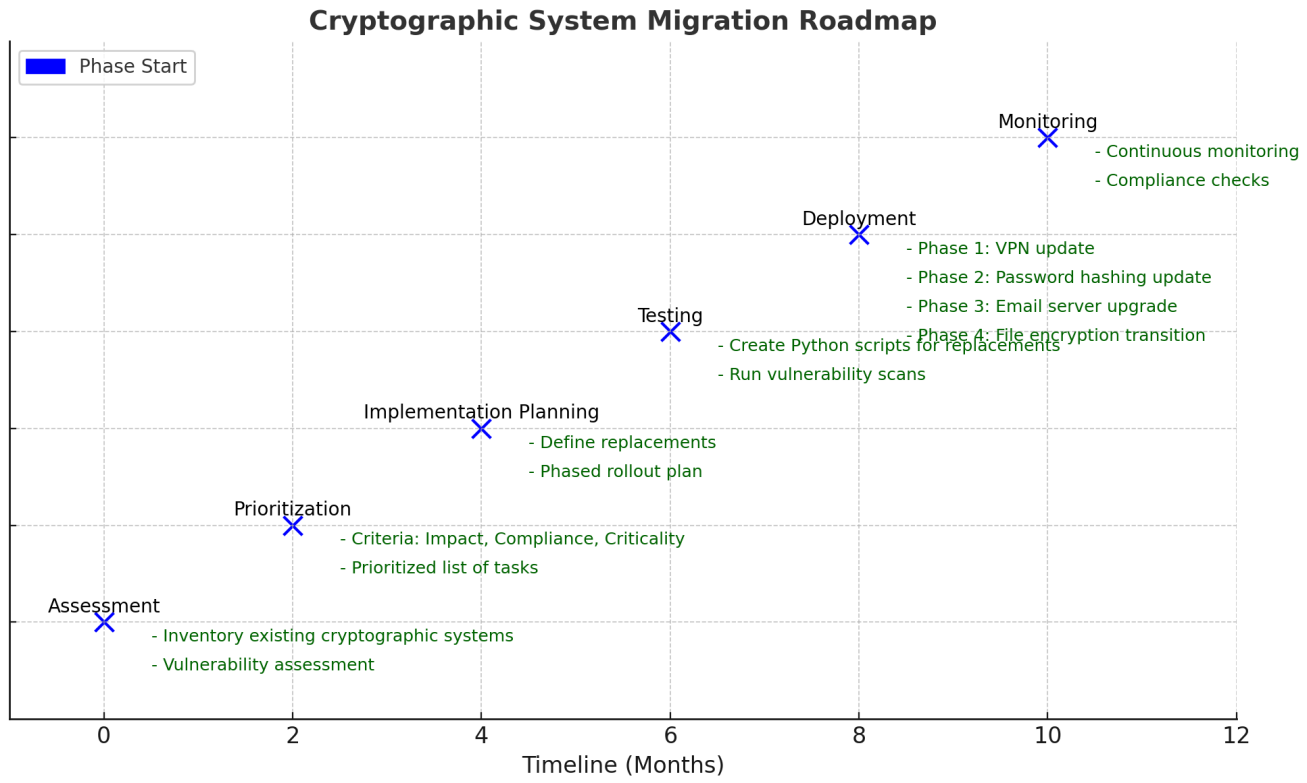
Step 6: Monitoring

1. **Continuous Monitoring**:
    - Deploy automated tools like Splunk to monitor cryptographic system performance and vulnerabilities.
    - Regularly audit VPN usage logs and password strength metrics.
2. **Compliance Checks**:
    - Conduct quarterly compliance reviews against GDPR and other relevant regulations.

Explore how these factors influence the adoption of stronger cryptographic systems and the transition to PQC

This phased migration plan, influenced by security, compliance, and resource factors, ensures TechServe Ltd. transitions to robust cryptographic systems and prepares for the quantum era while maintaining business continuity and compliance.

Roadmap



Summary of Part 4

1. **Introduction**

The Cryptographic Vulnerability Simulator is a tool designed to analyze and identify weak cryptographic implementations in source code files. It retrieves stored vulnerabilities from the Crypto Inventory database and presents users with identified issues. The tool also provides manual fix suggestions for each file through an interactive GUI.

2. **Prerequisites**

Before using the simulator, ensure that:

- The `CryptoInventoryTool` has been executed to scan files and store vulnerabilities in `crypto_inventory.db`.
- The simulator script is available and correctly configured.
- Python and required libraries (sqlite3, tkinter) are installed.

3. **Running the Simulator**
    1. Open a terminal or command prompt.
    2. Navigate to the directory containing the simulator script.
    3. Run the script using:
       python crypto_simulator.py
    4. The simulator window will launch, displaying a table of vulnerable files.

4. Using the Simulator

1. Viewing Vulnerabilities:
    - The main window displays a list of files, cryptographic issues, and their severity levels.
    - The table provides an overview of all discovered vulnerabilities.
2. Checking Suggested Fixes:
    - Double-click any file entry in the list to view recommended manual fixes.
    - A pop-up window will provide a brief description of what changes are needed.
3. Interpreting Severity Levels:
    - High: Immediate action required to prevent security risks.
    - Medium: Should be addressed soon to enhance security.
    - Low: Non-critical but recommended for best practices.

5. Compliance Monitoring

- The tool categorizes vulnerabilities by severity to help prioritize fixes.
- Users can address high-risk issues first to maintain compliance with security best practices.

6. Summary

The simulator provides an easy-to-use interface for reviewing cryptographic weaknesses and obtaining guidance on improving security. It complements the Crypto Inventory Tool by presenting stored vulnerabilities interactively. Users should follow the suggested fixes to enhance the security of their applications.

Conclusions & Lessons Learned

Conclusions:

- **Quantum Computing Threat**: Traditional cryptography is at risk, making the transition to Post-Quantum Cryptography (PQC) essential.
- **Cryptographic Agility**: Organizations must adopt flexible cryptographic systems to ensure smooth migrations.
- **Compliance & Standards**: Following NIST and EU guidelines helps maintain security and regulatory compliance.
- **Security Tools**: The Crypto Inventory Tool and Cryptographic Vulnerability Simulator assist in identifying and mitigating weaknesses.
- **Phased Migration Strategy**: A structured roadmap ensures a smooth transition to quantum-safe encryption.
- **Continuous Monitoring**: Regular security audits and automated monitoring tools (e.g., Splunk) are critical for maintaining security.

Lessons Learned:

- **Proactive Migration is Key**: Waiting until quantum attacks become practical increases risks—early planning is crucial.
- **HNDL is a Real Concern**: Data encrypted today may be decrypted in the future, making PQC adoption urgent.
- **Testing & Validation Matter**: Before deployment, cryptographic changes must be thoroughly tested for vulnerabilities.
- **Security is an Ongoing Process**: Continuous monitoring, auditing, and adapting to new threats is essential.
- **Interoperability Challenges:** Organizations must ensure that new encryption methods integrate smoothly without breaking existing systems.

List of Tools And References Used

- ChatGPT
- Wikipedia