Proof of Concept (PoC) Report

Tool Name: KimcilWare Decrypting Tool

History

1 Description:

i. **Ransomware Name:** KimcilWare
ii. **Discovered:** March 2016
iii. **Target:** Web servers running **Magento**, a popular e-commerce CMS
iv. **Impact:** Encrypted all website files, replacing content and appending .kimcilware extension
v. **Ransom Note:** Typically titled README_FOR_UNLOCK.txt, demanding ~1 BTC
vi. **Encryption Algorithm:** Believed to be **AES (Advanced Encryption Standard)**, though implemented poorly in some variants

Researchers identified that **early** KimcilWare **versions reused hardcoded keys and used weak encryption practices**, allowing the creation of decrypting tools.

2 What Is This Tool About?

The KimcilWare Decrypting Tool is a security utility developed to help victims recover files encrypted by the KimcilWare ransomware without paying the ransom. The tool works by reversing the AES encryption used by the ransomware, either via:

- Known static keys (hardcoded in the malware)
- Weak or predictable encryption routines

The tool is most effective on **early variants** of KimcilWare, which had flawed implementations of cryptography.

3 Key Characteristics / Features:

Feature	Description
AES Decryption Support	Uses AES decryption in ECB mode to recover .kimcilware files

Feature	Description
Bulk File Scanning	Automatically detects and decrypts all .kimcilware files in a specified folder
Known Key Usage	Utilizes a **hardcoded or discovered AES key** used by the original malware variant
Non-Destructive	Creates decrypted copies of the files (e.g., .decrypted extension), preserving the originals
Offline Functionality	Operates completely offline; no communication with C2 servers is required
Educational PoC Version	Community-created versions exist as proof-of-concept for malware analysis and recovery labs
Cross-Platform Support	Usually written in Python, allowing cross-platform usage (Windows/Linux) with minimal dependencies

4 Types / Modules Available:

Although most versions of the KimcilWare decrypting tool are standalone utilities or proof-of-concept scripts, we can logically break down its structure into **functional modules** based on how it operates.

Module / Component	Purpose
Key Handler	Manages static or discovered AES keys used for decryption. Early KimcilWare variants used hardcoded keys (e.g., "kimcilware123456").
File Scanner	Recursively scans directories to detect files with the .kimcilware extension.

Module / Component	Purpose
Decryptor Core	Implements AES decryption (commonly AES-ECB). Takes encrypted data, decrypts it, and outputs plaintext.
File Writer	Safely writes the decrypted output to new files (e.g., .decrypted) to avoid corrupting originals.
Error Handling / Logging	Captures decryption failures (e.g., incorrect key, corrupted input) and logs them for analysis.
CLI Interface (Optional)	Command-line interface that allows batch decryption from terminal or console.
Padding Stripper (Optional)	Removes null bytes or PKCS7 padding if needed (depends on variant behavior).

5. How Will This Tool Help?

Use Case	Description
File Recovery	Allows victims to **decrypt their encrypted web files** without paying the ransom (if weak variant).
Malware Analysis	Assists **reverse engineers and security analysts** in studying KimcilWare's encryption behavior.
Training/Education	Can be used in labs to teach **cryptanalysis, incident response, and ransomware mitigation**.
Data Sanitization Testing	Helps test how ransomware-infected environments behave post-decryption.

Use Case	Description
Automation-Friendly	Can be integrated into IR pipelines for automated detection and decryption during recovery.

6. Proof of Concept (PoC) Images:

7. 15-Liner Summary:

1. **Name:** KimcilWare Decrypting Tool
2. **Purpose:** Restore files encrypted by KimcilWare ransomware
3. **Targeted Ransomware:** KimcilWare (.kimcilware extension)
4. **Affected Platforms:** Magento-based web servers
5. **Encryption Used:** AES (mostly ECB, weak key handling)
6. **Key Type:** Often hardcoded/static in early variants
7. **Functionality:** Batch decrypts .kimcilware files
8. **Input:** Encrypted files with .kimcilware extension
9. **Output:** Decrypted file copies with .decrypted extension
10. **Modules:** File scanner, AES decryption, key handler, writer
11. **Interface:** CLI-based; optional logging and padding stripping
12. **Requirements:** Python 3, PyCryptodome library
13. **Limitations:** Only works on decryptable variants
14. **Use Cases:** Recovery, incident response, malware analysis
15. **Status:** PoC available; real-world tested on early strains

8. Time to Use / Best Case Scenarios:

Scenario	Why It's Ideal
Immediately after infection (no reboots)	File structure and infection pattern still intact; ransomware may not have deleted backups
Offline backups contain encrypted files	Files can be recovered safely without risking spread

Scenario	Why It's Ideal
During forensic analysis of a ransomware sample	Helps reverse engineer the encryption scheme and validate assumptions
In DFIR labs	Used as part of training simulations for ransomware response

9. When to Use During Investigation:

Investigation Phase	Role of the Tool
Identification	Detect encrypted files via .kimcilware extension
Containment	Analyze if the encryption method is vulnerable or recoverable
Eradication	Ensure the decryptor works after removing the ransomware binary
Recovery	Decrypt encrypted assets from offline or live systems
Lessons Learned	Use as a teaching tool in post-incident analysis to show attack vector and response effectiveness

10. Best Person to Use This Tool & Required Skills:

Role	Why They're a Good Fit
Incident Responder / DFIR Analyst	Has experience handling live infections and recovering compromised systems.
System Administrator (Linux/Windows)	Can operate command-line tools, navigate file systems, and manage backups.

Role	Why They're a Good Fit
Malware Analyst / Reverse Engineer	Can modify or extend the decryptor based on malware variants.
Cybersecurity Student / Trainer	Useful for ransomware lab simulations and educational recovery scenarios.

i. Required Skills

a) Basic understanding of **encryption (AES)** and how ransomware works.
b) Familiarity with **Python scripting** and **command-line interfaces**.
c) Knowledge of **file systems**, file extensions, and data recovery.
d) Optional: Ability to use **sandbox environments** for malware testing.

11.Flaws / Suggestions to Improve:

Flaw	Impact
Static key requirement	Only works if AES key is known (hardcoded or extracted from malware sample).
Assumes ECB mode	Will fail if ransomware uses CBC or dynamic IVs.
No file validation	Decryption is attempted blindly — no pre-check for file structure or integrity.
Single-threaded	Slower on large datasets or multi-core systems.
No logging system	Difficult to audit success/failure rate of batch decryption.
Lacks GUI	CLI-only interface limits usability for non-technical users.

12. Good About the Tool:

Strength	Description
Recovers files without paying ransom	Critical for early KimcilWare variants with weak encryption.
Lightweight & fast	Minimal dependencies, no installation needed beyond Python and PyCryptodome.
Open-source / customizable	Can be extended, improved, or ported to other ransomware cases.
Batch decryption	Supports decryption of entire infected directories.
Good for malware analysis	Useful for understanding ransomware behavior in educational or lab settings.
Offline operation	Does not require Internet or contact with any malicious infrastructure.

Prepared By:

Intern Name: Kush Dewal

Intern ID: 360