IIG University of Freiburg

Web Security, Summer Term 2012

Insecure Cryptographic Storage

Dr. E. Benoist

Sommer Semester

Table of Contents

- Presentation
- Examples
  - Attacks
- Recommendations
- PCI Data Security Standard
- Conclusion

Insecure Cryptographic Storage

- **Data and Credential are rarely protected with cryptographic functions**
  - Data collected can be used by attackers
  - For Identity Theft
  - or other crimes like Credit Card Fraud
- **Most common problems**
  - Not encrypting sensitive data
  - Using home grown algorithms
  - Insecure use of strong algorithms
  - Continued use of proven weak algorithms (MD5, SHA-1, RC3, RC4, etc.)
  - Hard coding keys, and storing keys in unprotected stores

E-Commerce Web Site

- **Suppose we manage a e-shop**
  - We sell goods and clients pay using their credit cards
  - We have to store the address and references of all our clients for the legal issues.
  - Data stored: name, address, e-mail, phone, Credit Cards Numbers
- **Our web site is attacked**
  - Attackers access to our Database
  - They can harvest the whole content of our customer clients

E-Commerce Web Site (Cont.)
Damages?

- **For the Clients**
  - Use of Credit Cards Number by attackers
  - Privacy violation
  - Identity Theft
  - ...
- **For The Web Site**
  - Reputation
  - Clients data stollen (can be resold to a competitor)
  - Business secrets stollen
- **For the Credit Card Company**
  - Reputation

Stored Data are Very Sensitive

- **Should be protected with cryptographic tools**
- **Encryption**
  - If you need to read and write data: symmetric encryption (e.g. DES, AES)
  - If reading and writing are done by different entities: asymmetric encryption (e.g. RSA)
- **One-way hash functions**
  - One input has always the same output
  - Impossible to go from the output back to the input
  - No collision can be generated (two inputs having the same output)
  - Example : SHA-256

Example: Self Made Crypto Algorithm
Hash Function

- **We want to hash a Medical Record Number**
  - Highly Sensitive data
  - Require One-Way hashing
  - Needs to be implemented by a partner.
- **Partner delivers a self-made algorithm**
  - Based on Modulo
  - This function is so complicated that it can not be reversed.

Self Made Crypto Algorithm

- **Algorithm**
  - Transform all the chars in the string into numbers
  - Take an arbitrary number (always the same)
  - Add this number to the last char, and modulo to remains in interval where conversion of number and char is automatic
  - Add the obtained number to the penultimate char and modulo
  - etc.
  - The numbers obtained form a string
  - The string is “secure”
- **Attack**
  - Take the obtained string, start from the first
  - Substract the arbitrary name to the char, we obtain the original value
  - Go on the same
  - If the obtained number is negative, then modulo was used, attacker just needs to substract this value.

Recommendations

Recommendations

Use only Strong Crypto Algorithms

- **Do not create cryptographic algorithms**
  - Only use approved public algorithms such as:
  - AES, RSA public key cryptography and SHA-256 or better
- **Do not use weak algorithms**
  - MD5 / SHA1 hash functions have been proven weak
  - Favor safer alternatives such as SHA-256

Handle Keys with extra Care

- **Generate keys offline and store private keys with extreme care**
  - Never transmit private keys over insecure channels
- **Store if possible your private key encrypted**
  - Using a pass-phrase
  - Or in a Password Manager

Protect Infrastructure Credentials

- **Data Base credentials**
  - Use tight file system permissions and controls
  - Encrypt securely credentials
- **Encrypted data should not be easy to decrypt**
  - database encryption,
  - useless if database connection pool provides unencrypted access

PCI Data Security Standard

- **Payment Card Industry Data Security Standard**
  - Developed by major credit card companies (e.g. Visa, Mastercard, American Express)
  - to help organizations preventing credit card fraud
- **Must be implemented by any merchant using Credit Cards**
  - A company processing, storing or transmitting payment card data must be PCI DSS compliant
  - Risk: losing their ability to process credit card payment
- **Compliance must be validated periodically**
  - Validation conducted by auditors (Qualified Security Assessors (QSAs)
  - Smaller companies just fill a self-assessment questionnaire.

PCI-DSS Requirements

- **Build and Maintain a Secure network**
  - Install and maintain a firewall
  - Do not use vendor-supplied default password and other security parameters
- **Protect Card-holder Data**
  - Protect stored card-holder data
  - Encrypt transmission of card-holder data across open, public networks
- **Maintain a Vulnerability Management Program**
  - Use and regularly update anti-virus software
  - Develop and maintain secure systems and applications
- **Implement String Access Control Measures**
  - Restrict access to card-holder data by business need-to-know
  - Assign a unique ID to each person with computer access
  - Restrict physical access to card-holder data

PCI-DSS Requirements (Cont.)

- **Regularly Monitor and Test Networks**
  - Track and monitor all access to network resources and card-holder data
  - Regularly test security systems and processes
- **Maintain an Information Security Policy**
  - Maintain a policy that addresses information security

PCI DSS - Storage of data

- **Card-holder Data**
  - Primary Account Number (PAN, a.k.a. credit card number)
  - Card-holder name
  - Service Code
  - Expiration Date
  - *Can be stored*
  - *Require protection*
- **Sensitive Authentication Data**
  - Full Magnetic Stripe
  - CVC2/CVV2/CID
  - PIN
  - **Can in no case be stored**

Store only necessary data

- **Develop a data retention and disposal policy**
  - Limit storage and retention time to which is required
  - for business, legal, and/or regulatory
- **Protect PAN**
  - Truncate card-holder data if full PAN is not needed
  - Never send PAN in unencrypted e-mails
  - Mask PAN when displayed
- **Render PAN unreadable anywhere it is stored**
  - Strong one-way hash functions
  - Truncation
  - Index tokens and pads (pads must be securely stored)
  - Strong cryptography with associated key management processes and procedures

Conclusion

- **Insecure Cryptographic Storage**
  - No encryption of sensitive data
  - Use of home-made "crypto" algorithms
  - Use of weak algorithms
- **Protection**
  - Use only proven strong algorithms
  - Take care the way data are stored
  - Encryption is useless if anybody knows the key!
- **PCI Data Security Standard**
  - MUST HAVE for any merchant using credit-cards
  - Describe security measures
  - Verifies their implementation.

References

- **OWASP Top 10 - 2007**
  `http://www.owasp.org/index.php/Top_10_2007`
- **A Guide for Building Secure Web Applications and Web Services**
  `http://www.lulu.com/content/1401012`
- **Web Application Security Consortium: Threat Classification** (2004)
  `http://www.webappsec.org`
- **Wikipedia PCI DSS**
  `http://en.wikipedia.org/wiki/PCI_DSS`
- **PCI Security Standards Council** (download PCI DSS)
  `https://www.pcisecuritystandards.org/tech/download_the_pci_dss.htm`