Chapter 9: Protecting Secret Data

1. Secret data
    a. Encryption keys
    b. Signing keys
    c. Passwords
    d. → It is impossible to store the above securely with *current* PC hardware
    e. Difficult to do with software, as well
    f. Anyone with physical access to your machine can get at this information
    g. Keep secret data secret! If anyone else knows about it, it's not secret…
2. Attacking secret data
    a. Information disclosure
    b. Tampering
    c. Can read unencrypted data if it's in the registry or a file (even binaries)
    d. If data is encrypted, a key to decrypt must be stored somewhere (so where do you store it!?)
    e. Memory can also be read (including page files)
    f. Attacker has access to everything by running your program on her machine
3. Avoiding storing secrets
    a. A hash may be possible: run hash function on data to get a unique value based on data (message digest) – if attacker gets message digest, she can't get at the password
        i. Salted hash: a random number that's added to a hash that's used to avoid dictionary attacks (attacker tries every possible secret key to decrypt encrypted data)
            1. make sure random number is cryptographically generated (avoid linear congruential functions)
            2. to confirm user knows secret, take secret, add salt, hash, then compare to stored value you have (they should be identical)
        ii. PKCS (Public-Key Cryptography Standard) 5: hashes a salted password >100 times
            1. helps mitigate dictionary attacks (attacker must come up with a password and salt, then figure out how many times to iterate to produce result)
            2. http://www.ietf.org/rfc/rfc2898.txt for more info
4. Protecting Secrets (MS specific)
    a. Can use Data Protection API functions
        i. CryptProtectData
            1. uses a method authentication code (MAC) to detect tampering
            2. Can allow only data owner to access data (default) (to allow others, must adjust CRYPTPROTECT_LOCAL_MACHINE flag in registry (make sure you apply proper ACLs to this key))

            3. NOTE: if you use CRYPTPROTECT_LOCAL_MACHINE make sure you have a backup of your result in cyphertext form in the event computer fails and must be rebuilt
        ii. CryptUnprotectData to decrypt
    b. Know what tools are provided by the OS(s) your app will run on – the different versions of Windows do not provide the same tools
5. Managing Secrets in memory
    a. Four basic steps
        i. Acquire the secret data
        ii. Use the secret data
        iii. Discard the secret data
        iv. Scrub the memory (memset to 0)
    b. Do the above as fast as possible to lessen chance secret is viewed
    c. Watch for compiler optimization issues (var no longer used in code so call to clear memory removed) (dead code removal)
6. Locking memory: you can lock page file but should use VirtualLock (MS OSs) (stops addresses from being paged out)
    a. A straight lock will affect OS
    b. Page file can still be written to hibernate file or crash dump
    c. Attacker can still attach a debugger to process
7. Ways to secure secrets (poor to good)
    a. Embedded key and XOR encoding (attacker can break in minutes)
    b. Embedded key and 3DES encoding (attacker can break in minutes – just has to find key)
    c. 3DES and store pw in registry (registry must have proper ACLS, pw should be a good one or it will be easily guessed)
    d. 3DES and store strong key in registry (ACLs on registry necessary)
    e. 3DES, strong key, ACLs on registry, password required