Chapter 11: Canonical Representation Issues

1. Do not make security decisions based on specific names if at all possible (if you must do it, accept only what you consider to be good – and use a regular expression to help validate it)
2. Remember that all input is evil, so you should not trust names (filenames) from untrusted source – remember your trust boundary
3. All canonicalization bugs lead to spoofing threats
    a. This can lead to information disclosure
    b. This can lead to elevation of privilege
4. Canonical: in its simplest or standard form
    a. C:\dir\test.dat, test.dat, ..\test.dat can all mean the same thing
    b. Don't make the wrong decisions based on a non-canonical representation of a name
5. Canonical Filename Issues
    a. Don't allow partial filenames as a match (ala a search engine)
    b. Watch for case sensitive versus case insensitive tools or services provided by OS (one service could say no, but the other yes and let a bad person in)
    c. Device name access can lead to DOS – know device names on OSs on which your app will run
    d. Watch for symbolic links to restricted files
        i. Link can have a lesser set of permissions which might then be applied to a restricted file (/etc/passwd)
        ii. Make sure data is properly canonicalized before passing on to other services
    e. All canonicalization issues exist because an app defaults to an insecure mode when a request for a resource does not match a known pattern (so remember secure by default – or to fail securely)
    f. Windows specific (mostly)
        i. 8.3 versus long filenames: long file names are shortened to fit 8.3 old DOS format – using the 8.3 version can bypass the check that disallows access to the long version – disable 8.3 support in your app if at all possible
        ii. be very wary if your code makes decisions based on extensions
        iii. trailing characters: Win32 likes to strip trailing dots from filenames (assuming they are not needed/intended) – thus an attacker can supply a file that ends with a ., then have it stripped and get access to a restricted file
    g. disallow path usage if at all possible – if you must do so, severely restrict it
6. Canonical Web-Based Issues
    a. Watch for (disallow) escape characters and hex representations
        i. 7 vs 8 bit ASCII
        ii. Hex codes
        iii. UTF-8 variable width encoding
        iv. UCS-2 Unicode encoding

        v. Double encoding (involves re-encoding encoded data)
        vi. HTML escape codes
    b. Dotless IP bug: single number representation for IP address (IE had this vulnerability
    c. Newlines in filenames can lead to data tampering (log files)
7. Visual Equivalence Attacks
    a. Using characters that look like know characters but are not
        i. Often used in phishing attacks
        ii. Many Cyrillic characters look like common characters
        iii. Watch for 0 versus O
        iv. Watch for 1 versus l
8. Don't make decisions based on names
9. Use regular expressions
10. Stop 8.3 filename generation
    a. HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Control\FileSystem – NtfsDisable8dotNameCreation: REG_DWORD : 1
11. Don't make decisions based on invalid requests – you won't think of all possible invalid requests