Outline for November 26, 2003

Reading: Chapter 22.3–22.5

Discussion Problem

When you examine a system for security problems, or design a system with strengthened security, creativity in anticipating problems is a major factor in the success of your work. Here are some creative answers to the riddle, "Why did the chicken cross the road?" A child might answer: "To get to the other side." But what would others answer?

Plato: For the greater good.

Aristotle: It is the nature of chickens to cross roads.

Karl Marx: It was an historical inevitability.

Timothy Leary: Because that's the only trip the establishment would let it take.

Captain James T. Kirk: To boldly go where no chicken has gone before.

Machiavelli: The point is that the chicken crossed the road. Who cares why? The end of crossing the road justifies whatever motive there was.

Jack Nicholson: 'Cause it freakin' wanted to. That's the freakin' reason.

Oliver Stone: The question is not, "Why did the chicken cross the road?" Rather, it is "Who was crossing the road at the same time, whom we overlooked in our haste to observe the chicken crossing?"

Albert Einstein: Whether the chicken crossed the road or the road moved beneath the chicken depends upon your frame of reference.

Ralph Waldo Emerson: The chicken did not cross the road; it transcended it.

Ernest Hemingway: To die. In the rain.

Colonel Sanders: What, I missed one?

Outline for the Day

1. Ideal: program to detect malicious logic
    a. Can be shown: not possible to be precise in most general case
    b. Can detect all such programs if willing to accept false positives
    c. Can constrain case enough to locate specific malicious logic
    d. Can use: writing, structural detection (patterns in code), common code analyzers, coding style analyzers, instruction analysis (duplicating OS), dynamic analysis (run it in controlled environment and watch)
2. Best approach: data, instruction typing
    a. On creation, it's type "data"
    b. Trusted certifier must move it to type "executable"
    c. Duff's idea: executable bit is "certified as executable" and must be set by trusted user
3. Practise: Trust
    a. Untrusted software: what is it, example (USENET)
    b. Check source, programs (what to look for); C examples
    c. Limit who has access to what; least privilege
    d. Your environment (how do you know what you're executing); UNIX examples
4. Practise: detecting writing
    a. Integrity check files such as binaudit, tripwire; go through signature block
    b. LOCUS approach: encipher program, decipher as you execute.
    c. Co-processors: checksum each sequence of instructions, compute checksum as you go; on difference, complain

d. Sandboxes: confine protection domain of process