Usability, Security and Privacy

Computer Science and Telecommunications Board

Butler Lampson
Microsoft Research

July 21, 2009

Usable Security: Things Are Really Bad

- Users don't know how to think about security
- User experience is terrible
  - Lots of incomprehensible choices
    - Lots of chances to say "OK"
  - A few examples:
    - Windows Vista User Account Control
    - Windows root certificate store
    - User interface for access control on files
    - Password phishing
    - Client certificates for SSL
    - Signed or encrypted email
- In general, more secure = less usable

The Best is the Enemy of the Good

- Security is fractal
  - Each part is as complex as the whole
  - There are always more things to worry about
    - See Mitnick's *Art of Deception*, ch. 16 on social engineering
- Security experts always want more—
  - More options : There's always a plausible scenario
  - More defenses: There's always a plausible threat
- Users just want to do their work
  - If it's not simple, they will ignore it or work around it
  - If you force them, less useful work will get done

USP Is About Economics

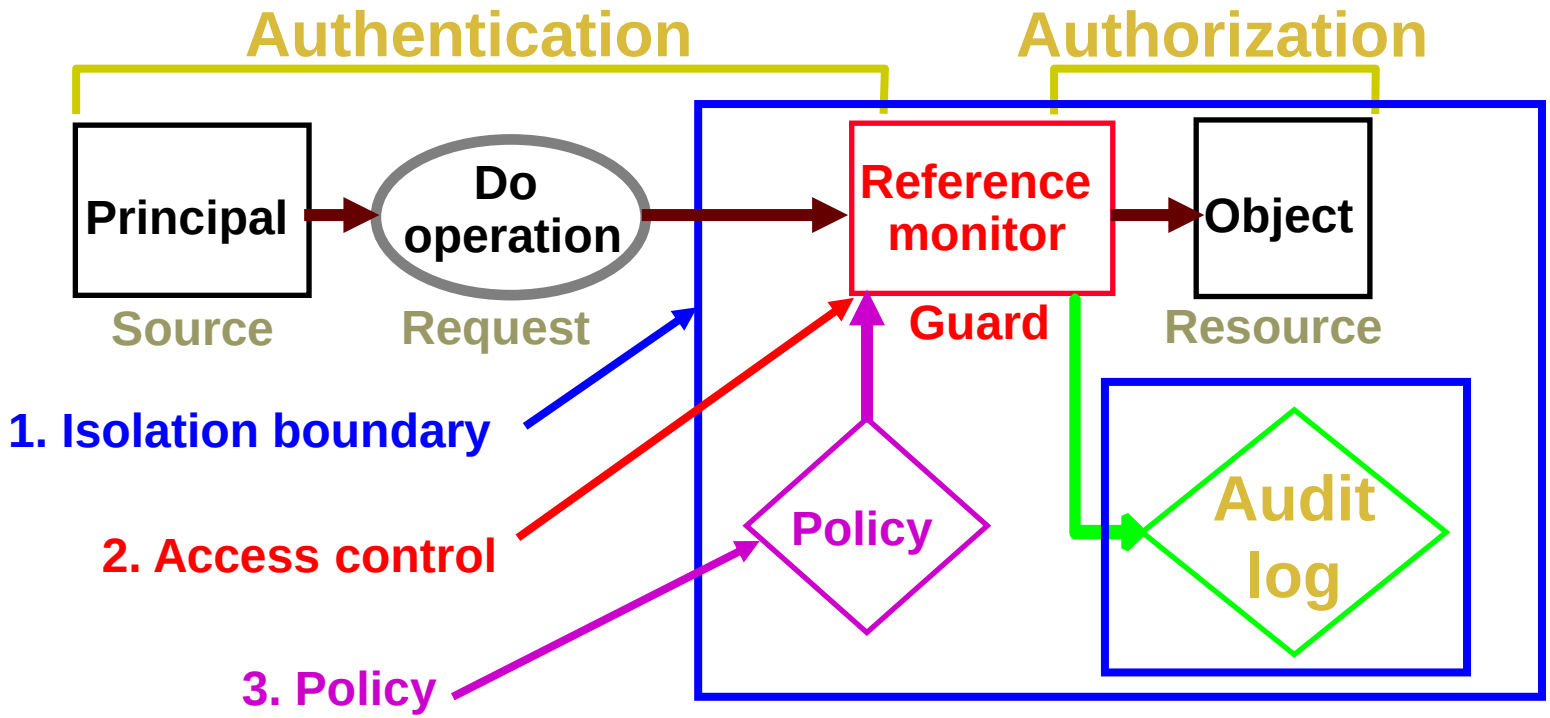
- Security is about risk management, not an absolute
  - There's benefit, and there's cost
    - We don't measure either one
    - Compare credit cards: fraud detection, CCVs, chip-and-PIN
    - The cost is *not* mostly in budgeted dollars
      - If you want security, you must be prepared for inconvenience.
        —General B. W. Chidlaw, 12 Dec. 1954
- Sloppy users are doing the right thing
  - Given today's lousy usability
  - Since the benefits of better security are not that big
- Providers have no incentive for usable security
  - They mostly just want to avoid bad publicity
- Tight security → no security

Technical Context

- **Security** is about
  - Secrecy — Who knows it?
  - Integrity — Who changed it?
  - Availability — Is it working?
  - Accountability — Who is to blame?
- **Privacy** is about controlling personal information
  - What is known—very hard
  - How it is used—mainly by regulation
- Two faces of security: Policy vs. bugs
  - **Policy**: user's rules for security / privacy
  - **Bugs** : ways to avoid policy

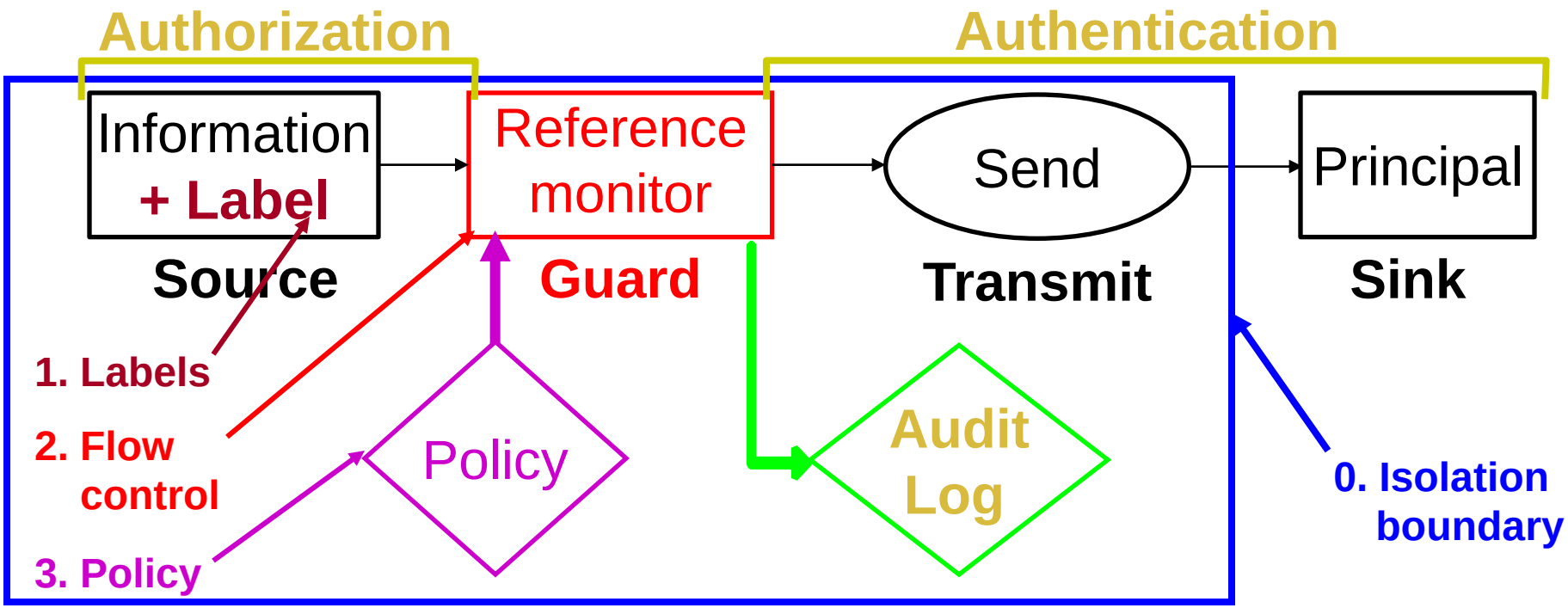
Context: The Access Control Model

1. **Isolation boundary** limits attacks to channels (no bugs)
2. **Access Control** for channel traffic
3. **Policy** management



Context: The Information Flow Model

0. **Isolation boundary** limits flows to channels (no bugs)
1. **Labeled** information
2. **Flow control** based on labels
3. **Policy** says what flows are allowed



User Models

- Users need a model they can understand
  - It has to be *simple* (with room for elaboration)
  - It has to (usually) not cause much *hassle*
  - It has to be *true* (given some assumptions)
  - It does *not* have to match the implementation
    - It gets compiled or interpreted, just like a language
- A user model is for saying what happens
  - Vocabulary: Objects and actions (nouns and verbs)
  - Policy: what should happen
    - General rules + exceptions
    - Must be meaningful, and small enough to audit
  - History: what did happen

Metrics

- Cost of getting security / privacy
  - Sand in the gears
  - Time spent setting policy
  - Budgeted dollars for software, firewalls, ...
- Expected cost of not having security / privacy
  - *Cost* and *risk* of a breach
  - Both are hard to come by

Examples of "Ideal" Usability

- Authentication
  - Easy two factor: Prox card / phone + fingerprint / PIN
- Authorization
  - Access tied to place: Public, family, private folders
  - Declarative policy: Account owner can transfer cash
  - Information flow labels: Money, medical, private, ...
- Recovery
  - Time machine; reset software
- Privacy
  - Information flow + auditing

Accountability

- Real world security is about deterrence, not locks
- On the net, can't find bad guys, so can't deter them
- Fix? End nodes enforce **accountability**
  - Refuse messages that aren't accountable enough
    - or strongly isolate those messages
  - Senders are accountable if you can **punish** them
    - With dollars, ostracism, firing, jail, ...
  - **All trust is local**
- Need an ecosystem for
  - Senders becoming accountable
  - Receivers demanding accountability
  - Third party intermediaries

Accountability vs. Access Control

- "In principle" there is no difference

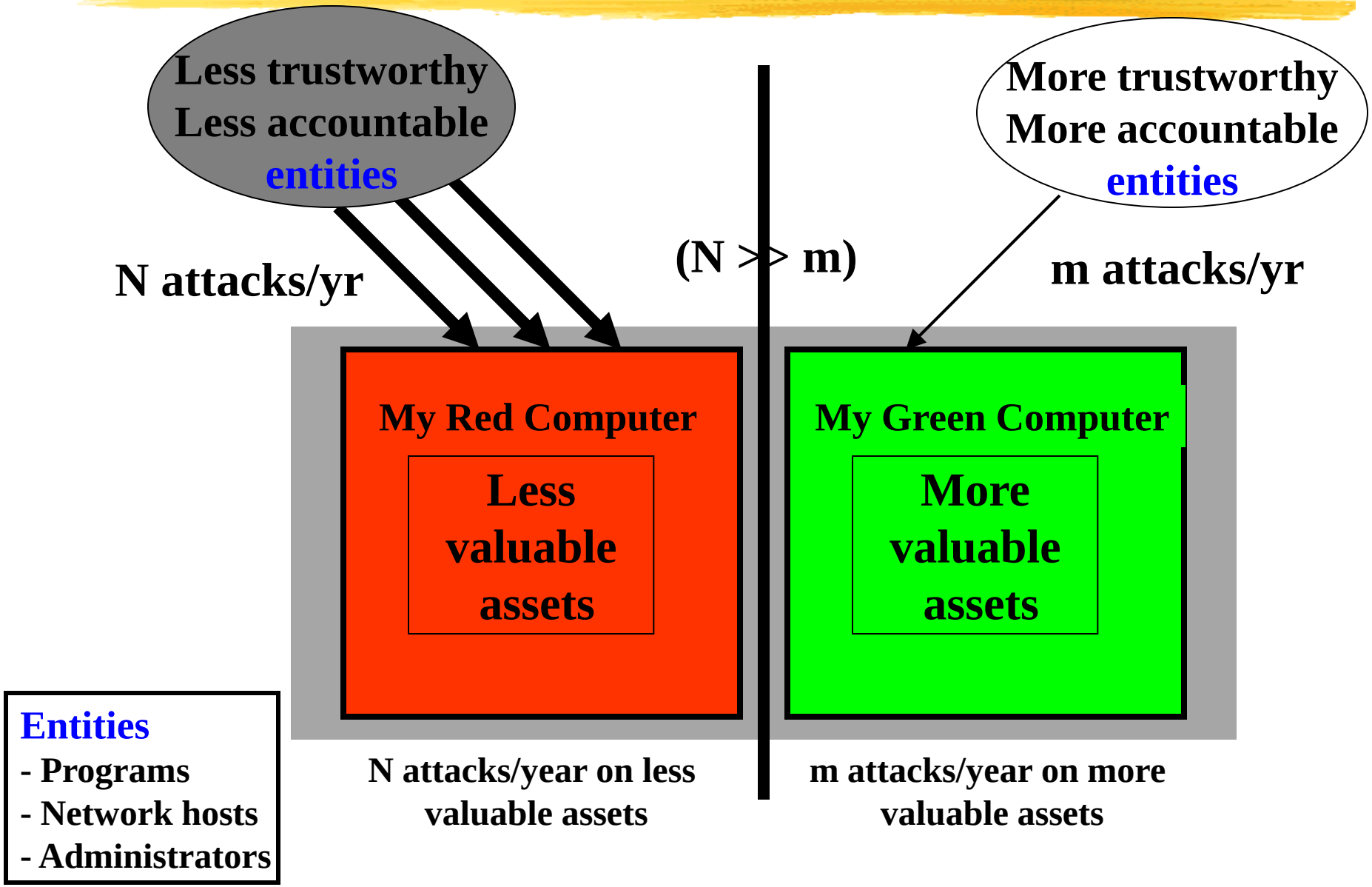
  but

- Accountability is about **punishment**, not access
  - Hence audit is critical
  - But coarse-grained control is OK—fix errors later

Freedom with Accountability?

- Partition world into two parts:
  - Green: More safe/accountable
  - Red : Less safe/unaccountable
- Red / green has two aspects, mostly orthogonal
  - User experience
  - Isolation mechanism
- Green world needs professional management

Red | Green


Less trustworthy
Less accountable
entities
More trustworthy
More accountable
entities
N attacks/yr
(N >> m)
m attacks/yr
My Red Computer
Less valuable assets
My Green Computer
More valuable assets
N attacks/year on less valuable assets
m attacks/year on more valuable assets
Entities
- Programs
- Network hosts
- Administrators

What Can Research Do?

- A way to measure the cost of inconvenience
  - Even better: A knob to adjust the cost/security tradeoff
- Some good user models for security and privacy
  - Even better: One model that people agree on
- Some "ideal" solutions for basic scenarios
  - Perhaps not feasible today, but not rocket science
- An infrastructure for accountability
  - That allows users to make choices they can understand
- Incentives for providers to make security usable

Conclusions

- Things are really bad for usable security & privacy
  - Need to focus on essentials, not on frills
- The root cause is economics
  - Users don't care much about security
  - We don't measure the costs
    - Either of getting security, or of not having it
  - Providers have no incentive to make security usable
    - They mostly want to avoid bad publicity
- Users need a model they can understand
  - It has to be *simple* (with room for elaboration)
- In this workshop: Ideas, not hand-wringing