# Quantitative Information Flow

Michael Clarkson
CS 711
11/17/2003

## An Insecure Program

```
u_H := get_pin_from_user();
c_H := get_pin_from_card();
auth_L := (u_H == c_H);
```



## An Insecure Program

```
u_H := get_pin_from_user();
c_H := get_pin_from_card();
auth_L := declassify(u_H == c_H);
```

- Could add declassify
- But why is that justified?



## Other Programs

- `l := (h == x);`
- `l := (h < x);`
- `l := (h == 0);`
- `l := (h + z) mod 2;`
- `h := rnd(); l:= h;`
- `k := rnd 2; l := k xor h;`
- `l := enc(h, k);`



## Richer Security Policies

- Information downgraded because of (e.g.) access control policy
- But this may leak other high security information
- Properties seen so far either
  - Require user's uncertainty to remain constant, which disallows downgrading
  - Allows uncertainty to be reduced arbitrarily low, releasing information
- Want to *bound* change in uncertainty



## Quantitative Information Flow

- Determine *how much* information flows
  - Rather than *whether* – qualitative
- Necessary class of policies
  - Many real systems require interference to function
  - Password checkers, cryptographic functions, aggregation functions, …
- Difficult to define a good metric, corresponding analysis

## Survey

- Several papers from 1987-2002
- Begin with information theory



## Information Theory

- System of events
  - $S = (E_1, \ldots, E_k)$
  - Probabilities of events $p_1, \ldots, p_k$
- Self-information: how rare an event is
  - $\mathcal{I}(E_k) = -\log p_k$
- Entropy: uncertainty in a system
  - $H(S) = E[\,\mathcal{I}\,]$



## Information Theory

- Mutual information
  - Amount of information about one system learned by observing another system
  - $\mathcal{I}(S,T) = \mathcal{H}(S) + \mathcal{H}(T) - \mathcal{H}(S \cap T)$
- Channel
  - Device by which signal is transmitted
- Capacity
  - Maximum amount of information transmitted reliably



## Covert Channel Capacity

[Millen 87]

- Relates NI to information theory

  **Theorem:** $H_{in}$ is NI with $L_{out}$ $\Rightarrow$ $\mathcal{I}(H_{in}, L_{out}) = 0$
- Channel capacity is maximum of $\mathcal{I}$ over all distributions of $H_{in}$, $L_{out}$



## Covert Channel Capacity

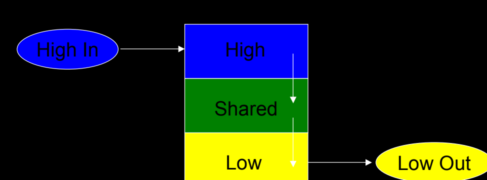
- `l := (h + z) mod 2;`
- NI does not hold
- Suppose
  - `h` and `z` independent H inputs
  - parity of `z` uniformly distributed
- Then l,h are independent:
  - (Given `h`)? either value of l is equally likely
  - $\mathcal{I}((h,z), l_t) = 0$



## Limited Declassification

[Weber 88]

- Deliberate declassification creates *shared* state



## Limited Declassification

- *n*-limited security:
  - Flow restrictions enforced
  - L user can distinguish *n* shared states
- Leaks at most $\log_2 n$ bits per observation
- Composable:
  - If S is *n* limited, T is *m* limited,
  - Then $S \circ T$ is *mn*-limited



## Nondeducibility on Strategies

[Wittbold & Johnson 90]

- *Strategy*: communication protocol between H (Trojan) and L users
  - Function from history of system to next H input
- NDS: no strategy can be excluded by low observations
- System is NDS iff no noiseless communication channels exist
  - Noiseless: inputs and outputs perfectly correlated
  - When formulated as *resource contention system*



## AFM

[Gray 91]

- Recall FM [McLean 90]:
  - Probability of low output cannot depend on previous high inputs or outputs
  - Gray formalizes with probabilistic state machines
- Gives security condition
- Shows SC implies bound on channel capacity
- Gives VC that implies SC



## AFM

- Let
  - $L_t$ be low output at time t
  - $T_k$ be trace of system from time 0..k
  - $\pi_L(T)$ be a projection of L events from T
- Security condition

$$\Pr(L_t \mid T_{t-1}) = \Pr(L_t \mid \pi_L(T_{t-1}))$$

- Pr is a prob measure defined in terms of event distributions



## AFM

- Channel capacity:
  - Maximum over average of $\mathcal{I}(\pi_H(T_i), L_i \mid \pi_L(T))$, $0 \le i \le n$, as $n \to \infty$
- **Theorem:** If H does not interfere with L then channel capacity from H to L is 0.
  - Proof: Security condition implies $\mathcal{I}(\pi_H(T_i), L_i \mid \pi_L(T)) = 0$

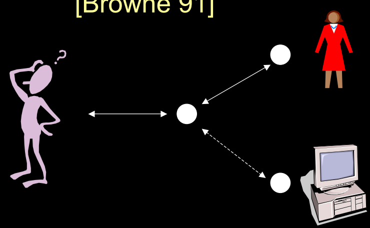


## AFM

- Verification
  - Security condition requires checking an uncountable number of expressions
  - Instead, use VC that implies SC
  - VC defined solely in terms of system transition function
    - Doesn't use Pr
  - Suppose $T_{t-1} \approx_L T'_{t-1}$
  - VC implies $\Pr(L_t \mid T_{t-1}) = \Pr(L_t \mid T'_{t-1})$
  - Which shows $\Pr(L_t \mid T_{t-1}) = \Pr(L_t \mid \pi_L(T_{t-1}))$

# Turing Test

[Browne 91]



Before interaction, distribution P; after, P'
System passes test if P = P'

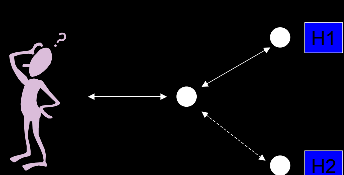


# Turing Test

Information flows when the uncertainty of the source of the output is reduced.



# Turing Test




Before interaction, distribution P; after, P'
System passes test if P = P'



# Turing Test

- Attacker has prob dist P over all traces of system
- Attacker observes current state
  - Set of states S is possible
- *TT*: P should be independent of S
  - Observations of system shouldn't change uncertainty of sources
- **Theorem:** System passes *TT* iff for all finite lengths of time, information flow is zero



# Interlude: Nondeterminism

- "I don't know"
  - Implementer, attacker have no control
  - Probabilistic, with unknown probabilities
- "I don't care"
  - Implementer left unspecified
  - Can be resolved probabilistically, possibilistically



# Information Flow Quantity

[Lowe 02]

- Also based on counting distinguishable behaviors
- More from Nate on 11/24

## Approximate Non-Interference
[Di Pierro, Hankin & Wiklicky 02]

Measure the difference between two probabilistic processes

- Processes are distribution transformers
- Difference of two processes is supremum norm of their resulting distributions
  - $P_1$: (.3, .5, .2, .1)
  - $P_2$: (1, 0, .5, .5)
  - $\varepsilon = \|P_1 - P_2\| = .7$
- When $\varepsilon = 0$, *probabilistically confined*



## Approximate Non-Interference

- Additional processes (*spies*) in the system may try to distinguish processes
  - Spies restricted to be passive, memoryless
  - Attacker restricted to finite number of tests *n*
- Attacker uses statistical hypothesis testing
  - Determine likelihood it has correctly distinguished
  - Pr(correct) $\propto \varepsilon \sqrt{n}$
- Effectiveness of spies depends on scheduler



## Approximate Non-Interference

- Define denotational semantics to compute final distributions of processes
  - Unsuitable for static analysis
  - Requires enumerating all traces
- Define abstract semantics to approximate $\varepsilon$
  - Probabilistic abstract interpretation



## Imperative Programs
[Clark, Hunt & Malacaria 02]

- Measure information leakage in **while** language, sans **while**
- Leakage: how surprising is output, given knowledge of input?
  - $\mathcal{L}(L_O) = \mathcal{H}(L_O \mid L_I)$
  - Upper bound: $\mathcal{H}(H_I \mid L_I)$
  - For deterministic programs, equivalent to AFM



## Imperative Programs

- Input to analysis
  - Bound [*a*,*b*] for each variable *x* s.t. $a \le \mathcal{L}(x) \le b$
- Analysis computes changes to bounds based on program
  - Conservative approximations necessary
    - But many rules over-approximate
  - Equality tests require solution of non-linear equations



## Conclusions

- Existing security policies too strong for useful programs
- Richer policies that bound uncertainty are needed
- Quantifying information flow by bounding channel capacity is promising