Nondeterminism and Information Flow
c.a. 1986-1997

Michael Clarkson
CS 711
10/29/2003

Limited Chronology

• 1986 – Nondeducibility (ND)
• 1987 – Generalized noninterference (GNI)
• 1988 – Forward correctability (FC)
• 1990 – Restrictiveness (RES), Flow model (FM), Nondeducibility on strategies (NDS)
• 1994 – Separability (SEP)
• 1997 – Perfect security property (PSP)

Relative Power

• Depends on exact definitions and assumptions
  – These vary widely, especially for GNI
• Under input totality:
  SEP ⇒ PSP ⇒ RES ⇒ FC ⇒ GNI
  [Zakinthinos & Lee 97]
• Without input totality:
  SEP PSP FC RES GNI
  [Mantel 02]

Nondeducibility

Can also formulate in terms of statistical independence:

Pr(l) > 0 and Pr(h) > 0 ⇒ Pr(h | l) > 0
Pr(l) > 0 and Pr(h) > 0 ⇒ Pr(l | h) > 0
Generalized Noninterference

[McCullough 87]

• Changes in high level input events do not cause changes in low level events
  – Fixes bug in ND
  – If all inputs happen before all outputs, generalizes:
    \( \sigma_1 \approx_L \sigma_2 \Rightarrow [S] \sigma_1 \approx_L [S] \sigma_2 \)
  • Domain of \([S]\) becomes sets of states
  • \(\approx_L\) becomes low-view set equality

If inputs and outputs are interleaved:
For all \(t \in \text{traces}(S)\), \(s = \text{change-high-input}(t_1)\), There exists \(t' \in \text{traces}(S)\) s.t.
\(t = uw, s = uw', t' = uw'' \) & \(w' = \text{echo} w''\)

• Not preserved under composition:
  – Machine A:
    • on receive H input: echo to H output
    • on receive “reset” (L input):
      – echo to L output
      – cancel all pending H outputs
      – if there were none, nondeterministically choose
    × output “nothing to reset”
    × no output
  – Machine B:
    • Same as A except no echo of “reset” to L output

• Composition of A and B doesn’t satisfy GNI
  – Because of ND choice, can’t tell whether change in H inputs has caused any change in L outputs

A and B satisfy GNI
  – Machine A:
    • on receive “reset”:
      – echo to L output
      – cancel all pending H outputs
      – if there were none, nondeterministically choose
    × output “nothing to reset”
    × no output
  – Machine B:
    • Same as A except no echo of “reset” to L output

Odd number of H inputs makes it impossible for both machines to output “nothing”

Individual systems secure; composition insecure
Composition of safety, liveness properties well-understood [Alpern & Schneider 85]
Why not security?
  – Many security properties outside of safety/liveness domain [McLean 94]
Composition Paradox

- Can we develop a theory of composition?
  - [McLean 94], [Zakinthinos & Lee 97], [Mantel 00, 02]
- Before these, there were various ad hoc compositional properties
- Not the only paradox...

Restrictiveness

- Fix GNI so that it is composable
- Recall GNI is:
  For all $t \in \text{traces}(S)$, $s = \text{change-high-input}(t)$,
  There exists $t' \in \text{traces}(S)$ s.t.
  $t = uw, s = uw', t' = uw''$ & $w' \approx w''$
- Changing H inputs may require introducing new H outputs
- After composition, new outputs from A become secret inputs to B
- Result: cascade of changes

Restrictiveness

- McCullough’s solution:
  - Restrict exchange of messages
  - Results in compositability
  For all $t \in \text{traces}(S)$, $s = \text{change-high-input}(t)$,
  There exists $t' \in \text{traces}(S)$ s.t.
  $t = uw, s = uw', t' = uw''$ & $w' \approx w''$ & $w = xy, w' = x'y', w'' = x'y''$
  & $x, x'$ contain only inputs

Restrictiveness

- Reject systems that require:
  - Insertion of new H output
  - Before/during the sequence of inputs after 1st change in H input
- Machine B from GNI was not restrictive:

Refinement Paradox

- A ND program is secure, but a refinement of it is not:
  ```
  l := h [0] 0 [1]; // secure
  l := h [0] 0;    // insecure
  ```
- Recent work [Zdancewic & Myers 03] shows how to solve this problem
- More on this later in the course
Restrictiveness

- Reject systems that require:
  - Insertion of new H output
  - Before/during the sequence of inputs after 1st change in H input
- Machine B from GNI was not restrictive:

Flow Model

- Recall another bug in ND:
  - ND disallows some safe flows
    - e.g., auditing: flows from L to H
  - Results from symmetry of independence:
    \[ \Pr(h \mid l) = \Pr(l \mid h) = \Pr(h) \equiv \Pr(l) \]

Flow Model

- Solution: break the symmetry using time
  - Not ok for \( h_t \) to affect \( l_{t+1} \)
  - But ok for \( l_t \) to affect \( h_{t+1} \)
- "Affect": statistical and causal dependency

Flow Model

- FM requires that \( l_t \) be independent of \( h_{0..t-1} \):
  \[ \Pr(l_t \mid l_{0..t-1} \& h_{0..t-1}) = \Pr(l_t \mid l_{0..t-1}) \]
- High events can be correlated with low events if caused by previous low event

Flow Model

- GNI and its extensions ignore causal dependencies
  - Overly restrictive
- FM allows more useful programs than GNI, ND, etc.
  - But still ensures high level of security
- FM later extended to quantitative information flow [Gray 91]
Nondeducibility on Strategies
[Wittbold and Johnson 90]

• Recall problem with ND
  – Nondeducible on a single run
  – Leaks information every \( n \) runs
  – Strategy exists to leak information
• Can require system to be nondeducible on any strategy
  – Formulated using information theory
  – System is NDS iff there exist no noiseless communication channels

Separability
[McLean 94]

• Absolutely no possibilistic information flow
• Like running the system as two separate, non-communicating processes
  – One process for each security level
• Few (useful) systems can satisfy this property

PSP
[Zakinthinos & Lee 97]

• Weaken SEP to allow high outputs to depend on low events
  – But all high inputs still possible
• Flow occurs when some high trace is not possible
  – Construction guarantees that low user can’t tell what that trace is
• Provably weakest such property
• Also composable

Conclusion

• Not too hard to generalize information flow to nondeterministic systems
• Hard to find balance between security and utility

\[ \text{ND} \rightarrow \text{GNI} \rightarrow \text{NDS} \rightarrow \text{RES} \rightarrow \text{PSP} \rightarrow \text{SEP} \]

more useful programs allowed

more secure

• Coming up: restricting, quantifying nondeterminism