Outline for April 6, 2000

1. Greetings and felicitations!
   a. Handouts

2. ACM and primitive operations
   a. Go over subjects, objects (includes subjects), and state \((S, O, A)\) where \(A\) is ACM
   b. Transitions modify ACM entries; primitive operations follow
   c. \(\text{enter } r \text{ into } A[s,o]\)
   d. \(\text{delete } r \text{ from } A[s,o]\)
   e. \(\text{create subject } s'\) (note \(A[s',x] = A[x,s'] = \emptyset\) for all \(x\))
   f. \(\text{create object } o'\) (note \(A[x,o'] = \emptyset\) for all \(x\))
   g. \(\text{destroy subject } s'\)
   h. \(\text{destroy object } o'\)

3. commands
   a. \textbf{command} \(c(s_1, ..., s_k, o_1, ..., o_k)\)
      \(\text{if } r_1 \text{ in } A[s_1,o_1] \text{ and } r_2 \text{ in } A[s_2,o_2] \text{ and} \)
      \(\text{...} \)
      \(r_m \text{ in } A[s_m,o_m] \text{ then } \)
      \(op_1; \)
      \(op_2; \)
      \(\text{...} \)
      \(op_n; \)\textbf{end.}
   b. Example 1: creating a file
      \textbf{command} \textit{create file}(p, f)
      \textbf{create object} \textit{f};
      \textbf{enter} \textit{Own} \text{ into } A[p, f]
      \textbf{enter} \textit{Read} \text{ into } A[p, f]
      \textbf{enter} \textit{Write} \text{ into } A[p, f]
   c. Example 2: granting one process read rights to a file
      \textbf{command} \textit{grant read}(p, q, f)
      \textbf{if} \textit{Own} \text{ in } A[p, f] \text{ then }
      \textbf{enter} \textit{Read} \text{ into } A[q, f]
   \textbf{end.}

4. What is the safety question?
   a. An unauthorized state is one in which a generic right \(r\) could be leaked into an entry in the ACM that did not previously contain \(r\). An initial state is safe for \(r\) if it cannot lead to a state in which \(r\) could be leaked.
   b. Question: in a given arbitrary protection system, is safety decidable?

5. Mono-operational protection systems: decidable
   a. Theorem: there is an algorithm that decides whether a given mono-operational system and initial state is safe for a given generic right.
   b. Proof: finite number of command sequences; can eliminate \textbf{delete}, \textbf{destroy}.
      Ignore more than one \textbf{create} as all others are conditioned on access rights in the matrix. (One exception: no subjects; then we need one \textbf{create subject}).
      Bound: \(s\) number of subjects (possibly one more than in original), \(o\) number of objects (same), \(g\) number of generic rights; number of command sequences to inspect is at most \(2^{gsog}\).

6. General case: It is undecidable whether a given state of a given protection system is safe for a given generic right.
   a. Represent TM as ACM; reduce halting problem to it
7. Take-Grant
   a. Introduce as counterpoint to NRU result
   b. Show bridges (as a combination of terminal and initial spans)
   c. Show islands (maximal subject-only tg-connected subgraphs)
   d. \(\text{can•share}(r, x, y, G_0)\) iff there is an edge from \(x\) to \(y\) labelled \(r\) in \(G_0\), or all of the following hold: (1) there is a vertex \(y''\) with an edge from \(y'\) to \(y\) labelled \(r\); (2) there is a subject \(y''\) which terminally spans to \(y''\), or \(y' = y''\); (3) there is a subject \(x'\) which initially spans to \(x\), or \(x' = x\); and (4) there is a sequence of islands \(I_1, ..., I_n\) connected by bridges for which \(x'\) is in \(I_1\) and \(y'\) is in \(I_n\).
   e. Describe can•steal; don’t state theorem