TOPIC

SYSTEM Z

Ravi Sandhu

This lecture is primarily based on:
John McLean, Roger R. Schell and
Donald L. Brinkley, "Security Models."
Encyclopedia of Software Engineering,

BLP

- S, fixed set of subjects
- O, fixed set of objects
- L, fixed lattice of security labels
- F: S U O $\rightarrow$ L, assignment of security labels to subjects and objects
- M: S X O $\rightarrow 2^{\{read,write\}}$, access matrix
- <F,M>, system state
- V is set of all possible system states
- A system consists of
  - An initial state v_0
  - A set of requests R
  - A state transition function T: V X R $\rightarrow$ V

BLP

- **<F,M> is read secure (simple security) iff for all s, o**

  $$\text{read in } M[s,o] \rightarrow F(s) \geq F(o)$$

- **<F,M> is write secure (star-property) iff for all s, o**

  $$\text{write in } M[s,o] \rightarrow F(s) \leq F(o)$$

- **<F,M> is state secure iff it is read secure and write secure**

BLP BASIC SECURITY THEOREM (BST)

Theorem 3. A system (v_0, R, T) is secure if and only if (**1**) v_0 is a secure state and (**2**) T is such that for every state v reachable from v_0 by executing a finite sequence of one or more requests from R, if $T(v, c) = v^*$, where $v = (F, M)$ and $v^* = (F^*, M^*)$, then for each $s \in S$ and $o \in O$:

- If $read \in M^*[s, o]$ and $read \notin M[s, o]$ then $F^*(s) \geq F^*(o)$;
- If $read \in M[s, o]$ and $F^*(s) \ngeq F^*(o)$, then $read \notin M^*[s, o]$;
- If $write \in M^*[s, o]$ and $write \notin M[s, o]$ then $F^*(o) \geq F^*(s)$; and
- If $write \in M[s, o]$ and $F^*(o) \ngeq F^*(s)$, then $write \notin M^*[s, o]$.

BLP WITH TRANQUILITY

- **F does not change**
- **$F^v(s) = F^{v0}(s)$**
- **$F^v(o) = F^{v0}(o)$**

- **BLP with tranquility is intuitively secure**
- **BLP with tranquility satisfies BST and thereby is formally "secure"**

BUT

- **System Z is intuitively (and egregiously) insecure**
- **System Z satisfies BST so BST is useless**

SYSTEM Z

- **Initial state v_0 is state secure**
- **Single transition rule: on any read or write request all subjects and objects are downgraded to system low and the access is allowed**
- **System Z satisfies Basic Security Theorem**

BLP WITH HIGH WATER MARK

- **F(o) does not change, $F^{v}(o) = F^{v0}(o)$**
- **F(s) can change but**
  - **only upwards, $F^{v}(s) \geq F^{v0}(s)$**
  - **only as far as user's clearance, $F^{v}(s) \leq F(user(s))$**
  - **every change upwards in F(s) requires removal of write from M[s,o] cells where after the change F(s) > F(o)**

- **BLP with high water mark is considered intuitively secure (and also satisfies BST)**

BLP WITH LOW WATER MARK

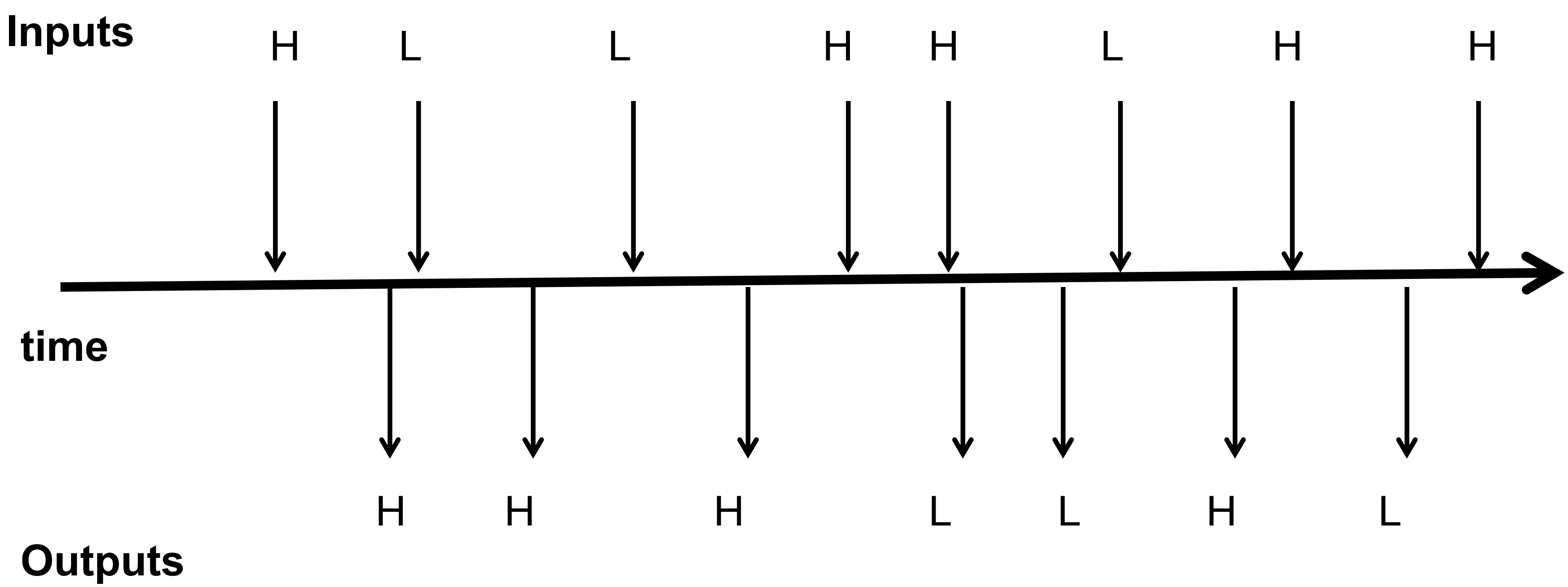
- **F(o) does not change, $F^v(o) = F^{v0}(o)$**
- **F(s) can change but**
  - **only downwards, $F^v(s) \leq F^{v0}(s)$**
  - **can downgrade all the way to system low**
  - **every change downwards in F(s) requires removal of read from M[s,o] cells where after the change F(s) < F(o)**

- **BLP with low water mark is considered intuitively insecure (and also satisfies BST)**
  - **memory of higher level reads can be retained in RAM, cache, CPU registers, program counter, etc**

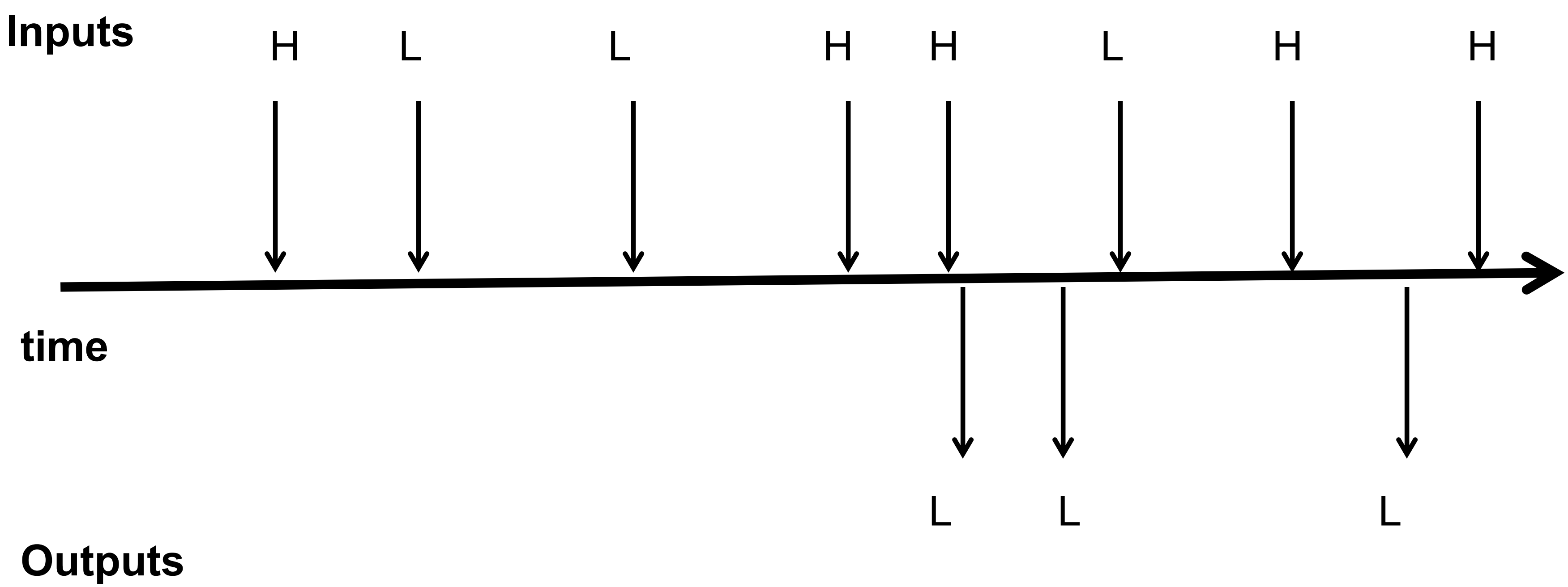
NON-INTERFERENCE

- **Views the system as a black box with input/output events that are caused by users**
- **McLean's paper assigns an input event the same level as the user's clearance. This is not correct. More correctly an input event can be caused by a user but its security level should be specifiable by the user.**
- **Reasonably intuitive and intuitively secure for deterministic systems**
- **For non-deterministic systems it pushes intuition boundaries**

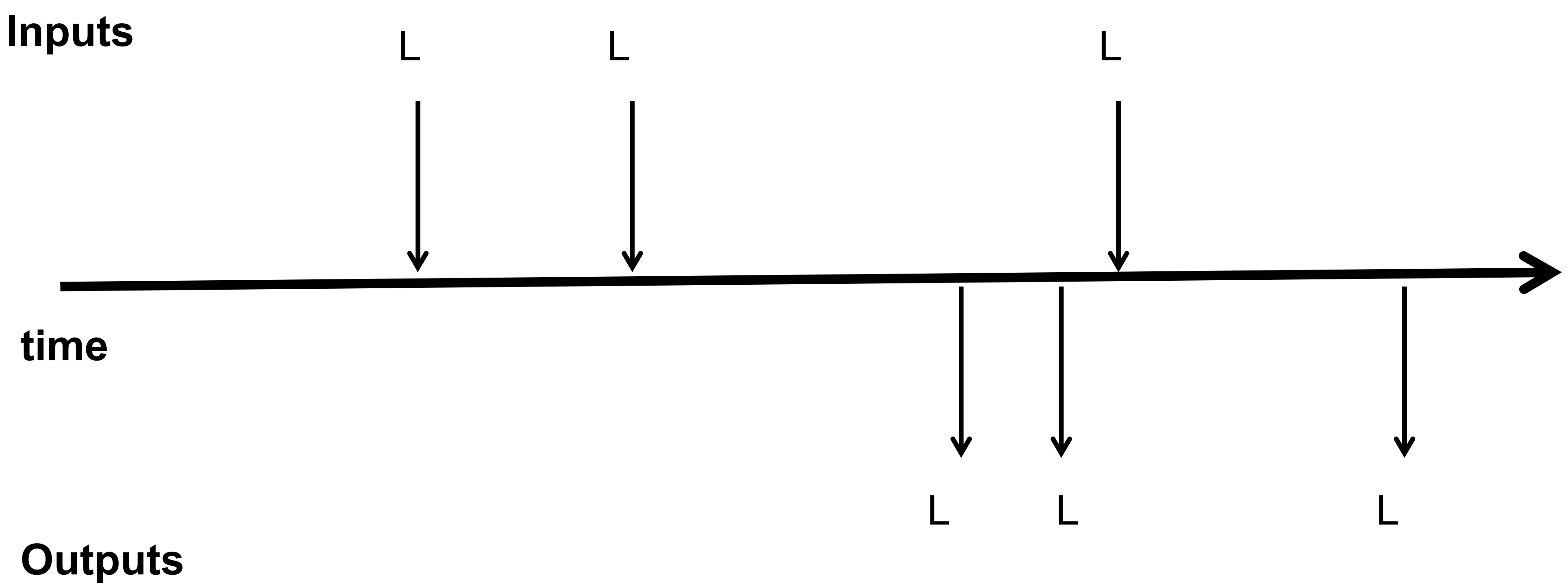
NON-INTERFERENCE



NON-INTERFERENCE



NON-INTERFERENCE



NON-INTERFERENCE

Goguen and Meseguer consider a deterministic system whose output to user u is given by the function $out(u, hist.read(u))$, where $hist.read(u)$ is an input history (trace) of the system whose last input is $read(u)$, a read command executed by user u.

NON-INTERFERENCE

Definition 7. Let *cl* be a function from *users* to *security levels* such that $cl(u)$ is the clearance of u. Further, let *purge* be a function from *users* $\times$ *traces* to *traces* such that

- $purge(u, \text{<>}) = \text{<>}$, where <> is the empty trace
- *purge(u,hist.command(w))* = *purge(u,hist).command* (w) if $command(w)$ is an input executed by user w and $cl(u) \geq cl(w)$, and
- $purge(u, hist.command(w)) = purge(u, hist)$ if command (w) is an input executed by user w and $cl(u) \not\geq cl(w)$.

A system satisfies *Noninterference* if and only if for all users u, all histories T, and all output commands c, $out(u, T.c(u)) = out(u, purge(u, T).c(u))$.

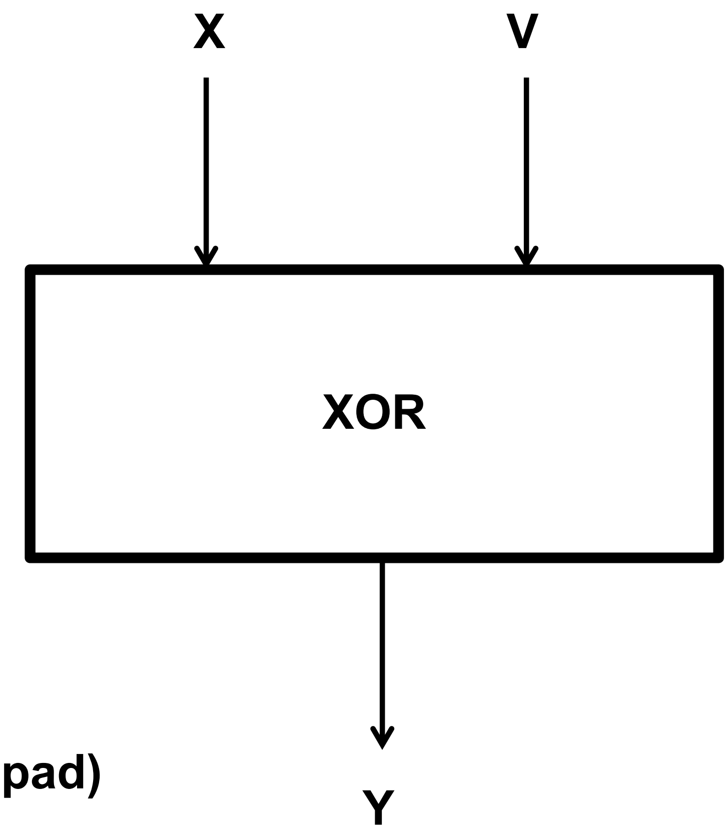
NON-INTERFERENCE vs BLP

(**1**) in general BLP is weaker than noninterference in that the latter prohibits many of the covert channels that the former would allow under the standard interpretation of its primitives, and

Generally understood that non-interference can deal with storage covert channels but not with timing covert channels

(**2**) Noninterference is weaker than BLP in that it allows low level users to copy one high level file to another high level file, which BLP would normally disallow as a high level *read* by the low level user.

NON-INTERFERENCE AND ENCRYPTION



X: plaintext
V: encryption key (one-time pad)
Y: ciphertext