SL 2007-556 532725

DEFENCE R&D DÉFENSE

Content

1. Problematic & selected high-level needs
2. The Vision
3. Chosen approach
4. Proposed solution options
5. Concluding remarks

1 Problematic & selected high-level needs

Problems:

- Officers on duty have *huge amount* of information to analyze in *limited periods of time*. They are subject to *high levels of stress and fatigue*
- Utilized C2ISs are *increasingly Cx* (more vulnerabilities, harder to debug, …)
- Utilized C2ISs are used in *hostile environments* such as Internet (C2ISs are subject to all kinds of hostile cyber attacks)

Some high-level needs:

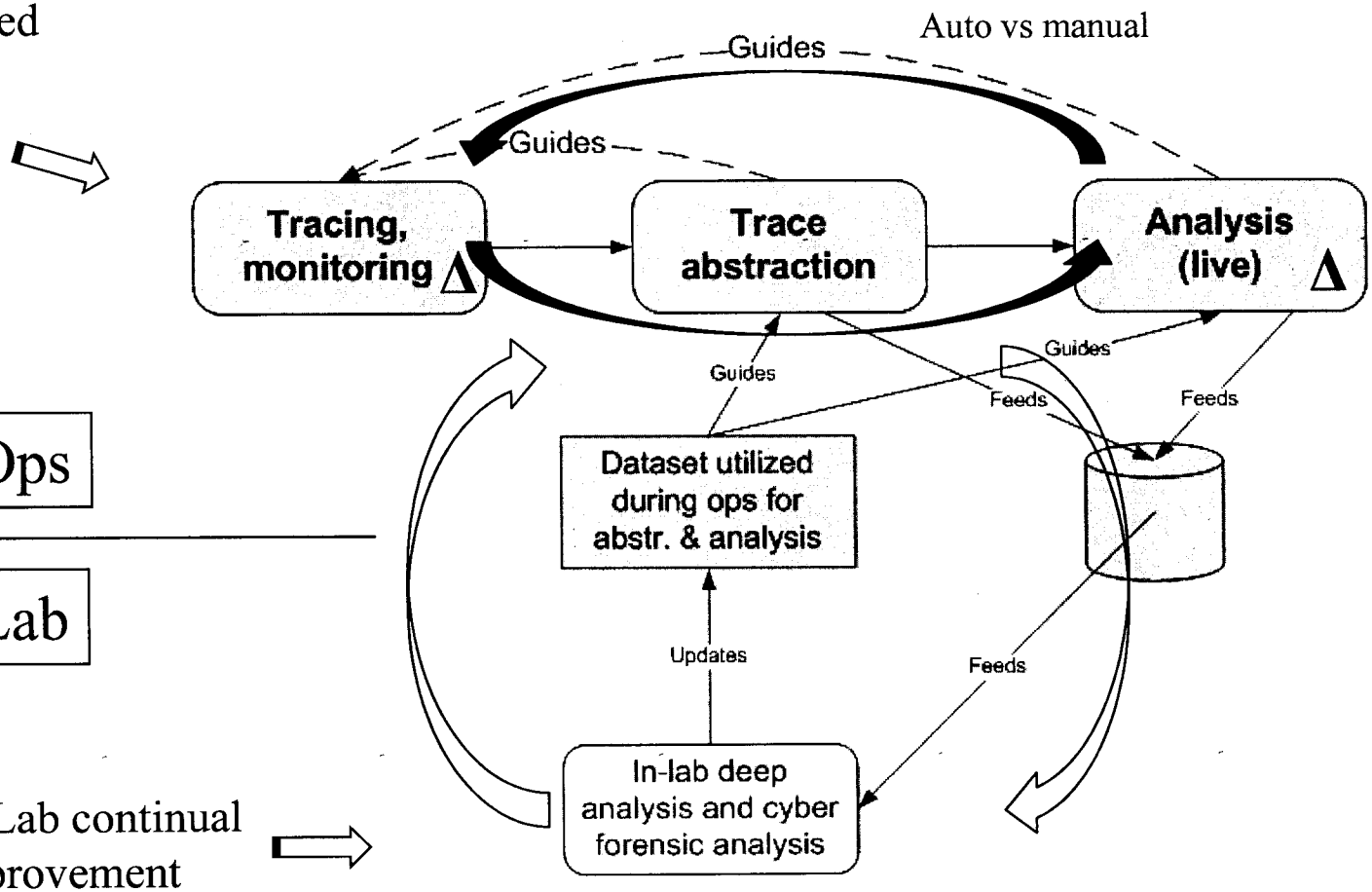
- Mechanisms to *ease the debugging* of complex C2ISs
- Mechanisms to *protect C2ISs against all kind of malfunctions or unwanted behaviours* during operation (all origins: design, cyber threat, operator miss-utilization, …)
- Do not impose a supplementary burden on officers' shoulders
- Give them an easy & quick to understand updated picture of their C2ISs' health

Propose them well understood solution options (& probable impacts) that will solve detected problems during operations

C2IS: Command and Control Information System

2 The Vision

Feedback-directed Diagnostic & Corrective System (*at Runtime*)



In-Ops

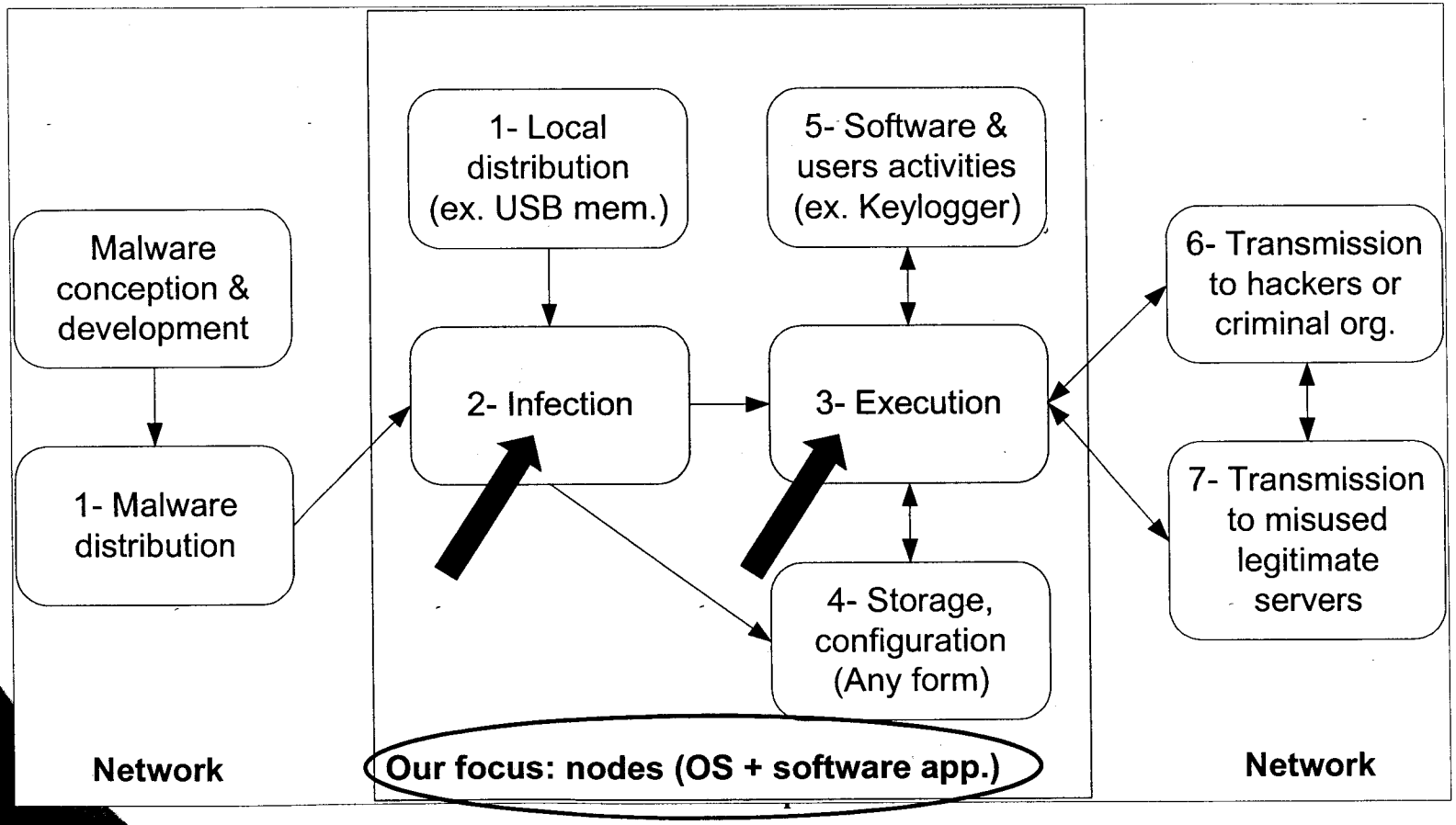
In-Lab

In-Lab continual improvement

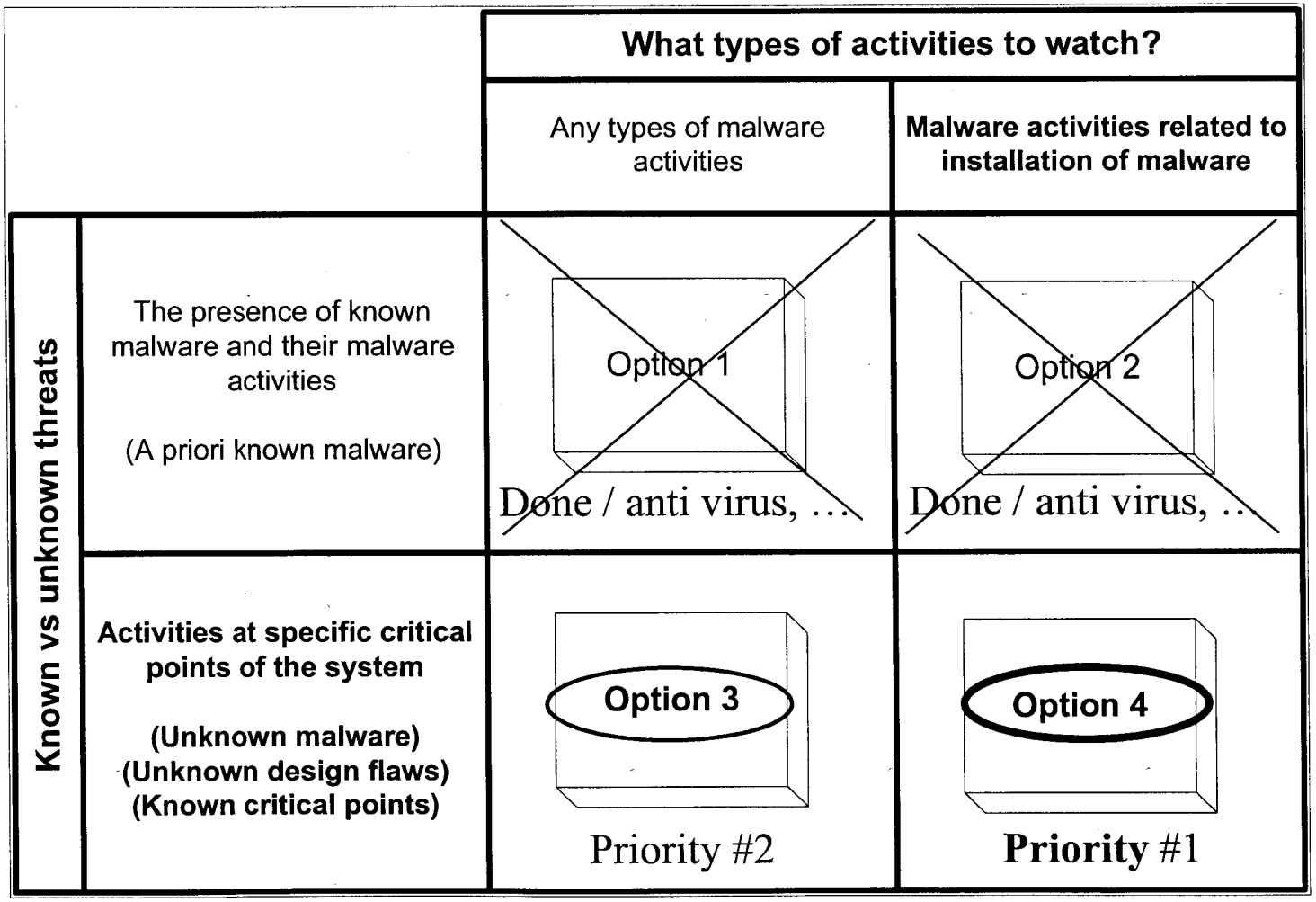
3 Chosen approach (1/2)

Goal: protection against all forms of malfunctions and unwanted behaviours during operations

ex: cyber attacks…



3 Chosen approach (2/2)

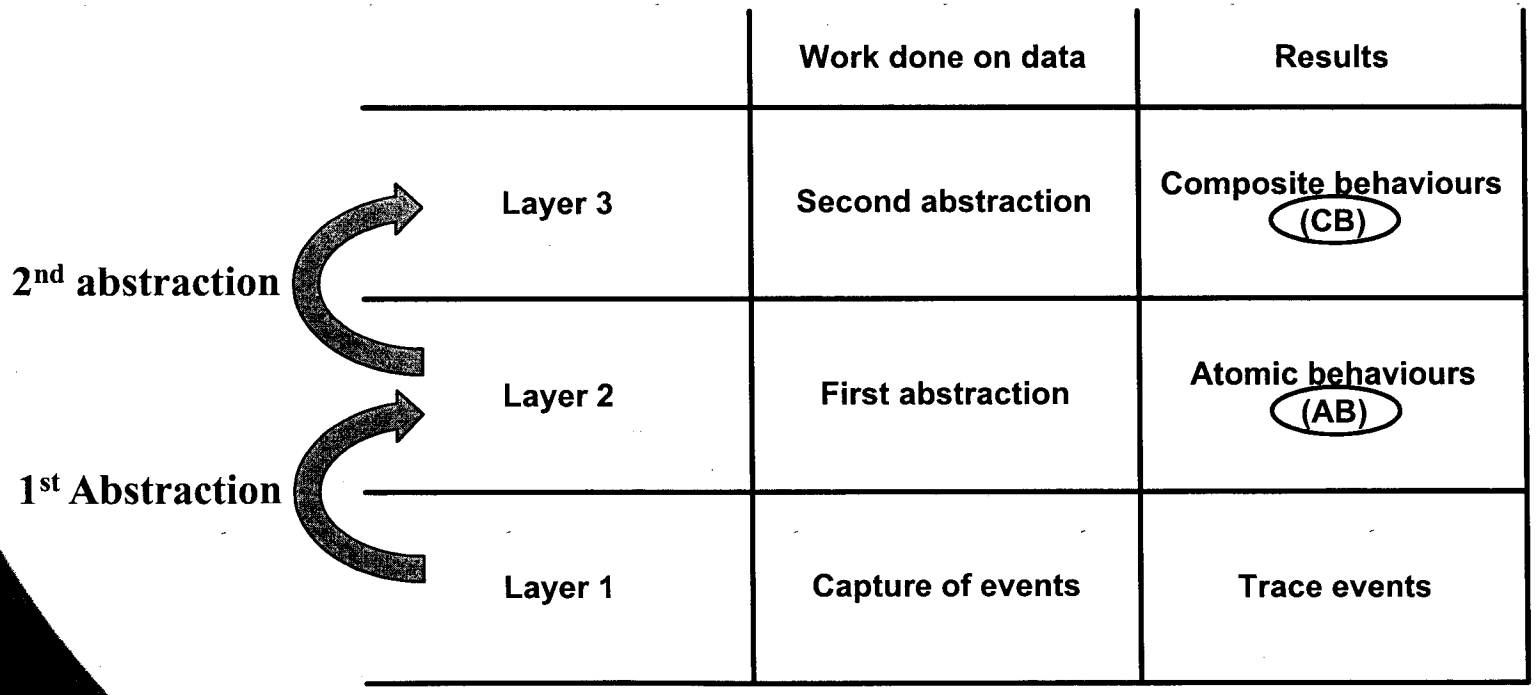


4 Proposed solution options (1/5)

Decouple the abstraction process in two faster processes:

1st abstraction: made *locally*, adapted, very fast

2nd abstraction: made *locally or remotely*



4 Proposed solution options (2/5)

1st abstraction (Events->ABs):

- Atomic Behaviours (ABs) represent *smallest "behaviours"* in a system
- ABs should be *"observable" on all utilized systems (behav. comparison)*
- ABs should *form the basis of a language* (for post-abstraction analysis)
- ABs should be saved (not trace events); cyber forensic analysis deals with ABs

AB = {suite of *related trace events* from the *execution trace*}

- Ex. a read on disk = {syst. call; call to VFS; read(); interrupt; …}

AB(t) = (Name, processes, t1, t2, *probability*, +…)

➔ ***Probability is not 100%:*** all events pertaining to AB at time "t" are not present in the execution trace. At least three possible reasons:

- Bad focus or resolution in the execution trace (trace events)
- Remaining events of an AB will be triggered in the near-future
- Abstraction process and mechanism not perfect

4 Proposed solution options (3/5)

2nd abstraction (ABs->CBs)

- *Composite behaviours (CBs)* represent what we want to study

CB(t) = {AB2, AB5, AB6, AB99, …}

CB(t) = (Name, processes, …, probability, …)

- CBs have associated probability as well
- Ex. (bot): {a socket was open from inside; critical data was sent over network}
- Comparison of CBs; (*/common Abs)*
- We can raise the "probability" of CB(t+1) *if we activate relevant LTTng probes*

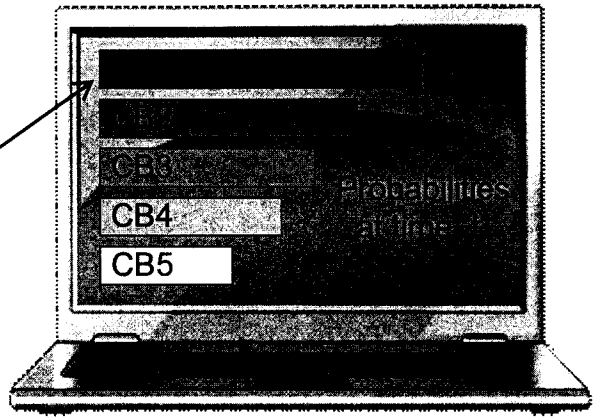
4 Proposed solution options (4/5

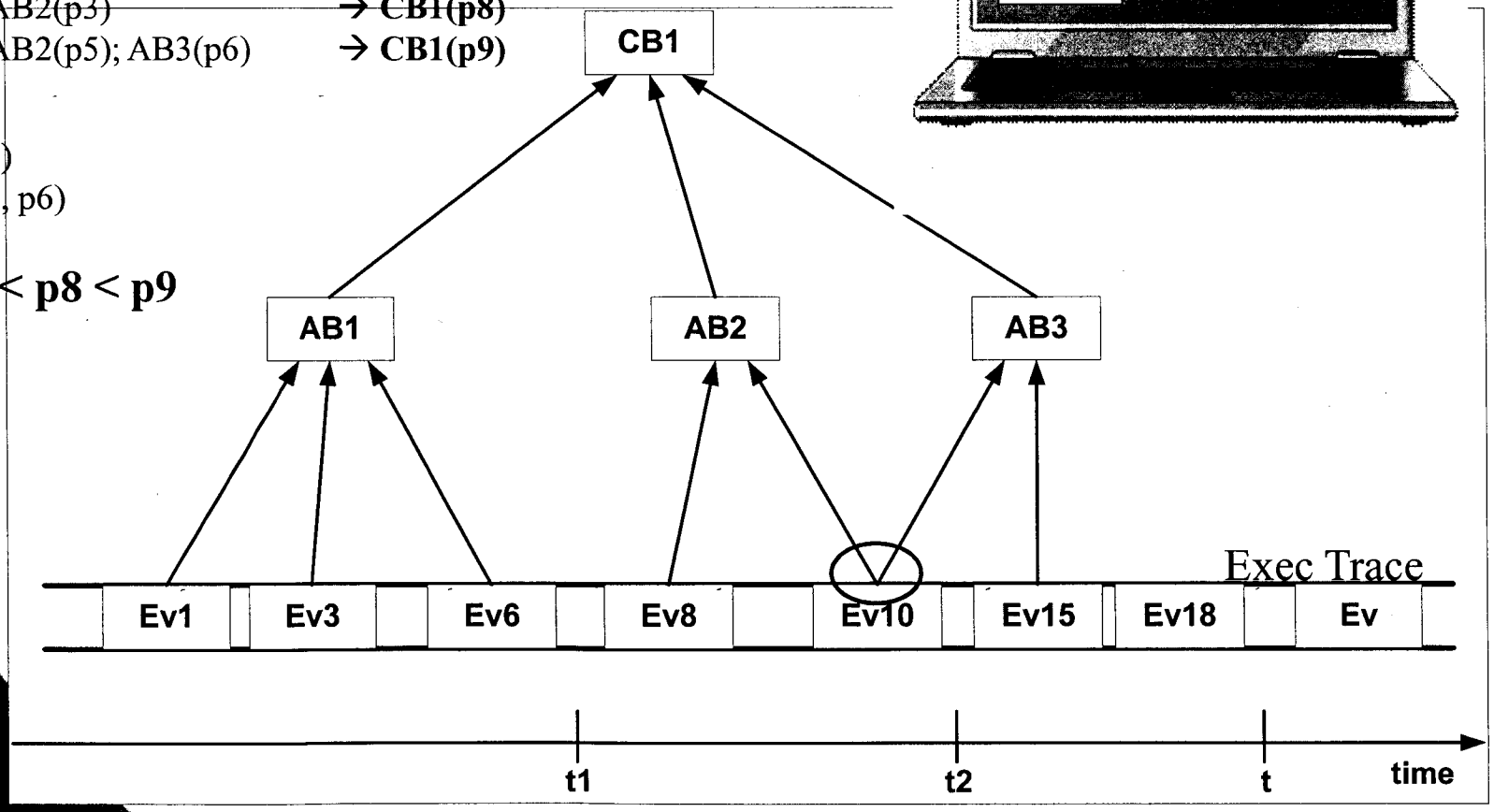
An example:

t1: AB1(p1) → **CB1(p7)**
t2: AB1(p2); AB2(p3) → **CB1(p8)**
t3: AB1(p4); AB2(p5); AB3(p6) → **CB1(p9)**

p7 = fn(p1)
p8 = fn(p2, p3)
p9 = fn(p4, p5, p6)

➔ **p7 < p8 < p9**





4 Proposed solution options (5/5)

Different types of analysis could be done using ABs and CBs:

- **Type 1**: ABs originate from *one node*
  - Behaviours analysis in the kernel of the OS
  - Detection of malware on this node
- **Type 2**: ABs originate from a *redundant architecture*
  - Behaviour comparison (CBs through ABs)
- **Type 3**: ABs originate from *distributed systems*
  - Global analysis of a *distributed system of systems*
- **Type 4**: ABs originate from a *multi-core/multi-processor system*
- **Type 5**: ABs originate from a *n-level system*
- **Type 6**: ABs originate from a *multi-threaded environment*
- Others (ex: ABs & CBs are easier to transfer on networks than trace events)

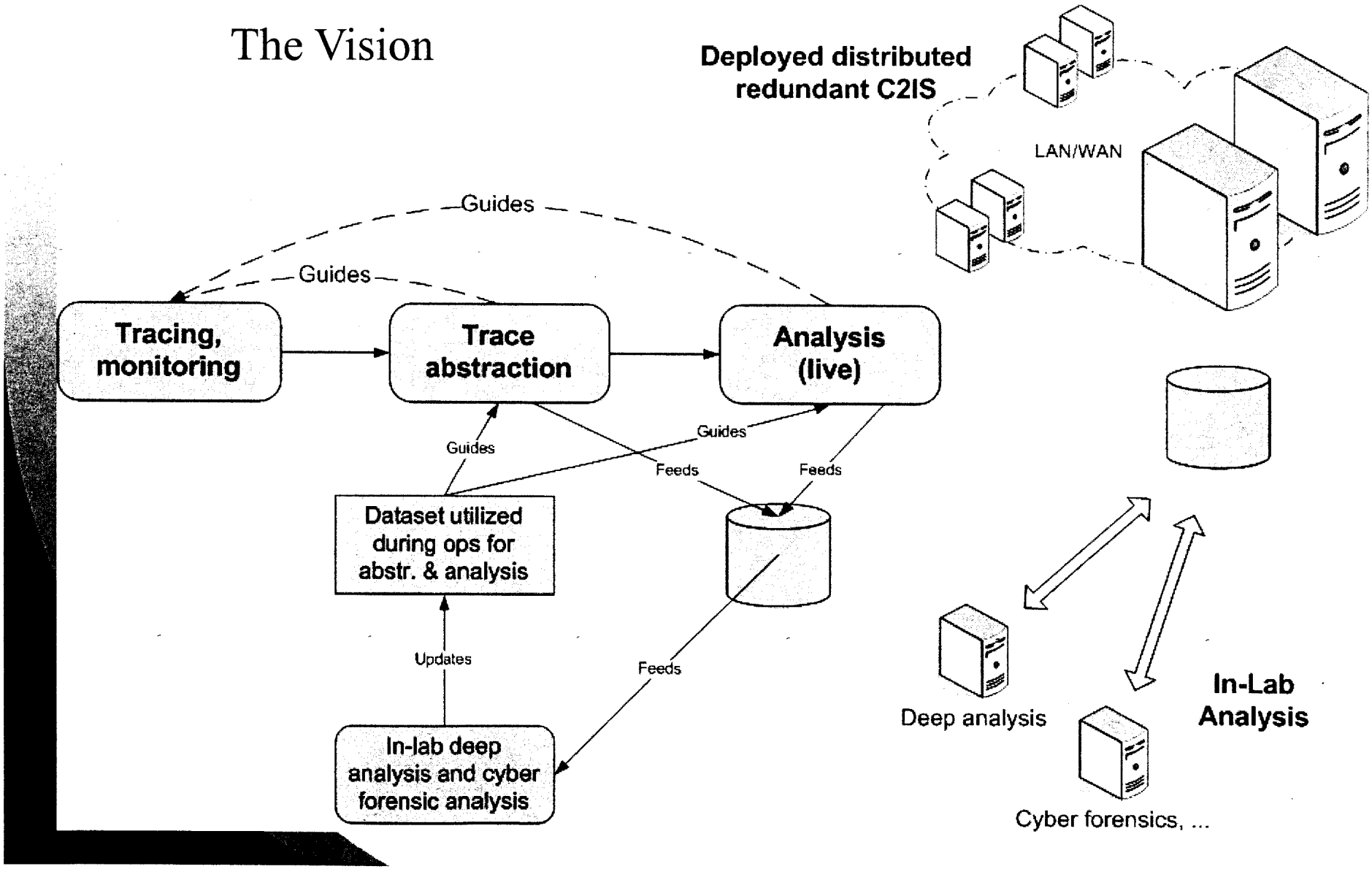
5 Concluding remarks

Ability to activate/deactivate LTTng probes at runtime is important:

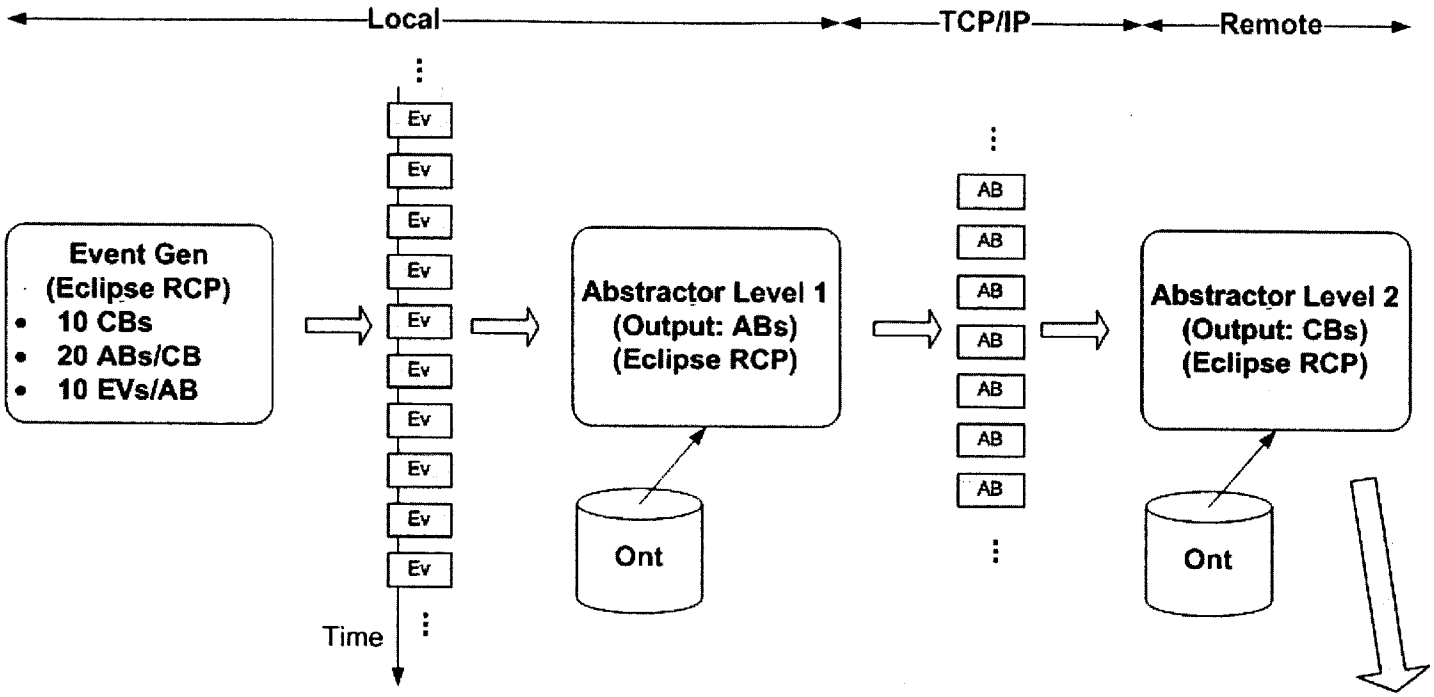
- Focus on desired targets in the kernel (with appropriate resolution)
- Raise "probability" of ABs (and thus CBs)
- *Earlier detection and reaction* to unwanted behaviours/states
- More precise dynamic analysis of the system (in-ops)
- More precise deep analysis and improvement of the system (in-lab)

Supplementary slide (1/2)

The Vision



Supplementary slide (2/2)



The Vision:
Short timescales: automatic reactions
Mid timescales: auto/manual reactions
Long timescales: manual reactions