Early Warning on a National Level – Project AMSEL

Martin Apel, Joachim Biskup, Ulrich Flegel, Michael Meier

technische universität dortmund

Computer Science Department

Chair VI, Information Systems and Security

Overview

- early warning systems
- project overview
- architecture
- challenges and technologies
  - efficient and effective classification and detection
  - enablement of required cooperation
- summary

Early Warning Systems [1]

- aim at
  - detecting yet unclassified but potentially harmful system behavior
  - based on preliminary indications
  - establish hypotheses, predictions and advices in not yet completely understood situations
  - include two meanings of „early“
    - “fast”: start early in time in order to avoid/minimize damage
    - “incomplete”: process uncertain and incomplete information

[1] 08102 Manifesto -- Perspectives Workshop: Network Attack Detection and Defense. Dagstuhl, 2008.

Project AMSEL - Goals

- development of an EWS for automatic
  - privacy and confidentiality preserving
  - **detection**
    - of known and unknown
    - automatized attacks (malware)
  - **reporting of**
    - incidents
  - integration into a situation picture

Automatisch **M**alware **S**ammeln und **E**rkennen **L**ernen

automatically collect and learn to detect malware

Approach

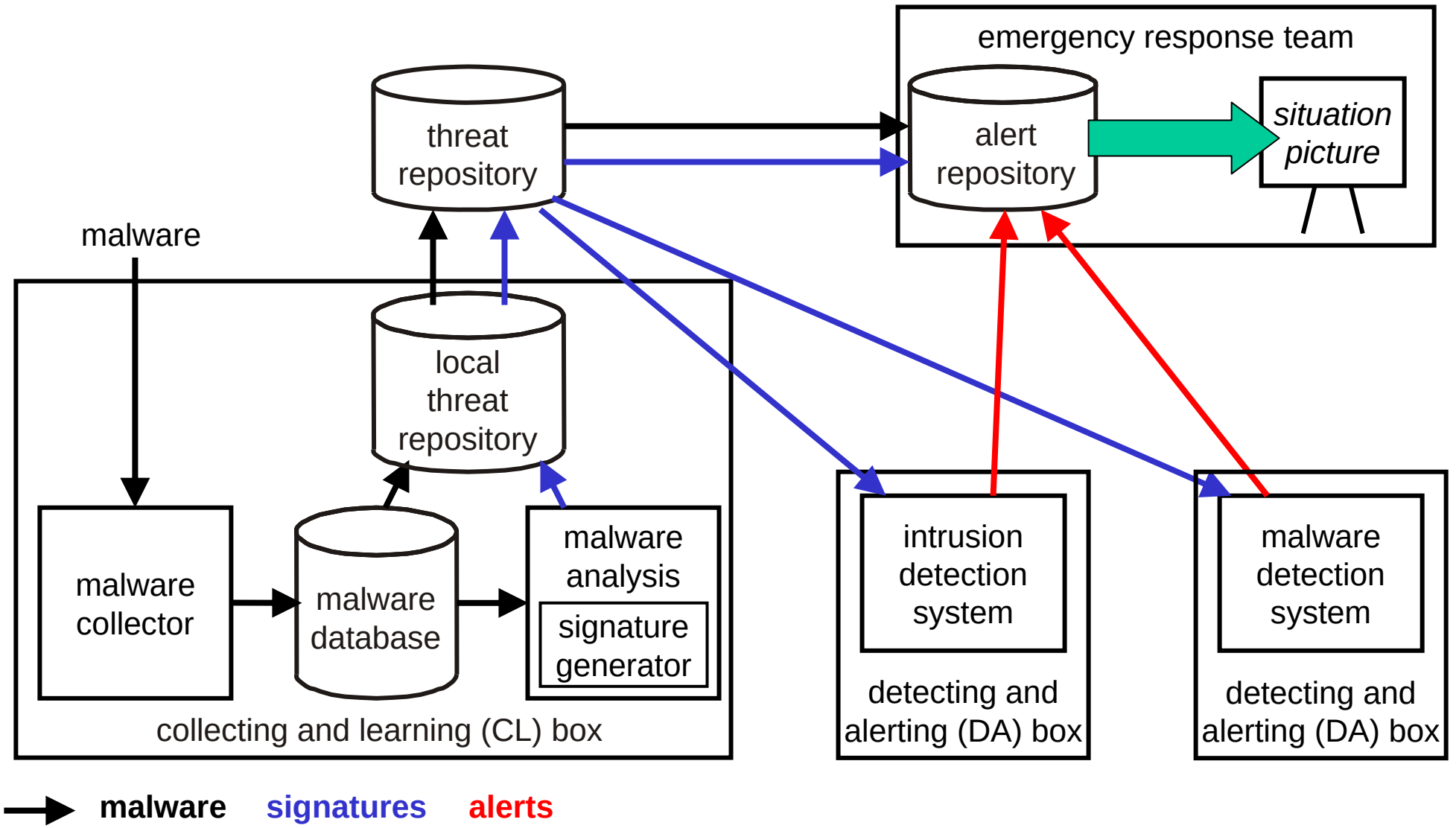
- coupling of technologies in an automatized process
  - sub-process detection
    - honeypot technology
      - malware collection
    - malware analysis technology
      - controlled execution and observation of malware
    - machine learning
      - generation of detection criteria (signatures)
  - sub-process reporting
    - central provision of detection criteria
    - update of detection systems
    - central reporting of detected incidents

Idea of a Malware EWS

- automatically
  - collect malware
  - analyze malware
  - generate signatures
  - distribute and deploy signatures
  - report alerts centrally

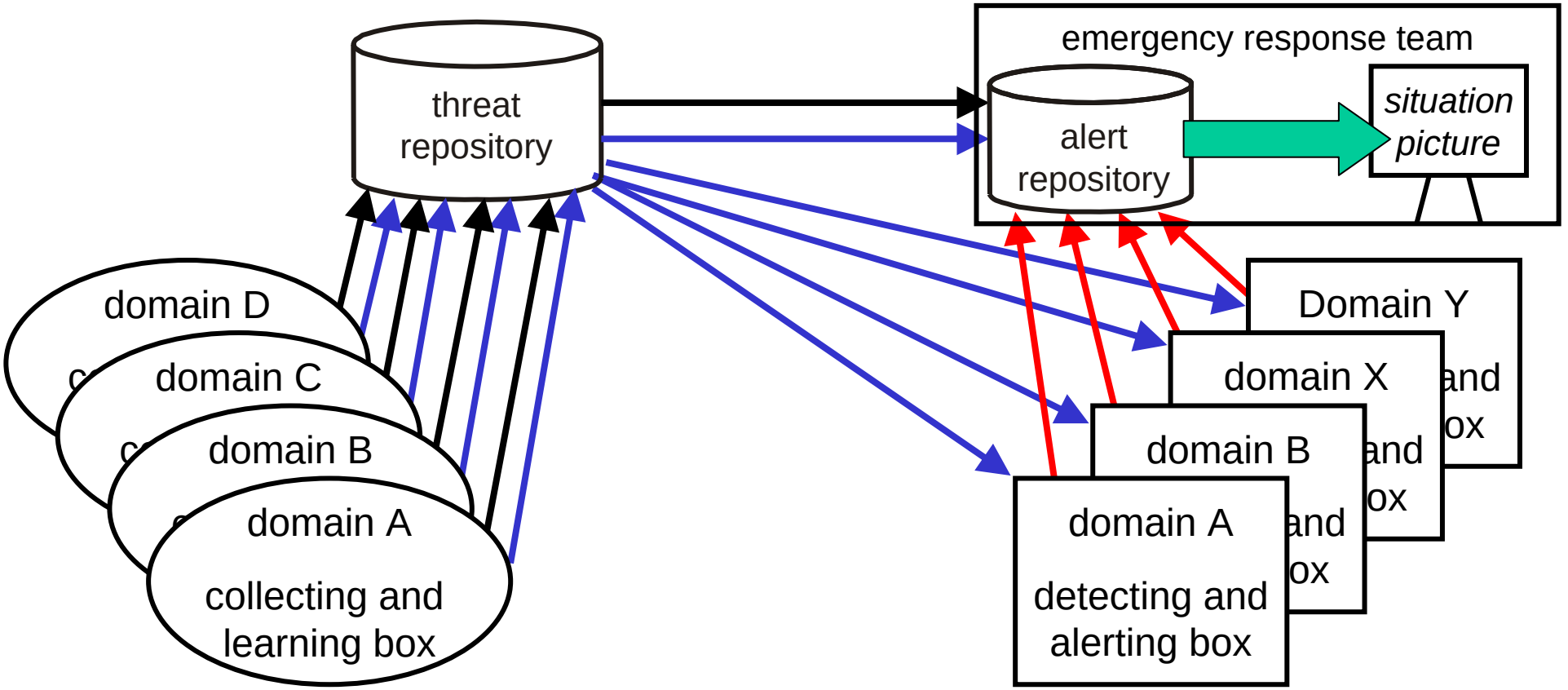
⇨ combination of misuse detection and anomaly detection techniques
  - provide specific alerts with low false positive rates
  - detect a priori unknown attacks

Architecture



Deployment Scenario

- protection level achieved and quality of situation picture depend on number and placement of deployed CL and DA boxes
  - ⇨ cooperative information exchange required



Challenges and Technologies

- efficient and effective classification and detection
  - malware collection
  - malware analysis
  - signature generation
  - malware detection
- enablement of required cooperation
  - balance/resolve conflicting interests

Malware Collector

- collecting new malware as early as possible
- Nepenthes and Amun
  - low-interaction server honeypots
  - emulate vulnerabilities
  - catches/collects malware binaries
    - typically downloaded after initial compromise
- possible extensions
  - honey clients
    - collecting drive-by-downloaded files
  - spam traps
    - collecting attached files or targets of URLs

Malware Analysis

- inspecting and extracting appropriate features characterizing and distinguishing malware and benign programs
- static analysis
  - static features: directly extracted from malware samples
    - byte sequences of code or data segments
    - control flow graphs extracted by disassemblers
  - morphing/obfuscation techniques and tools
    - generate programs of equal/similar functionality but different static feature instantiations
    - 30.000 new unique (wrt. static features) malware samples a day
      - polymorphic variants of a few malware types
    - would require to handle 30.000 new signatures a day

Malware Analysis

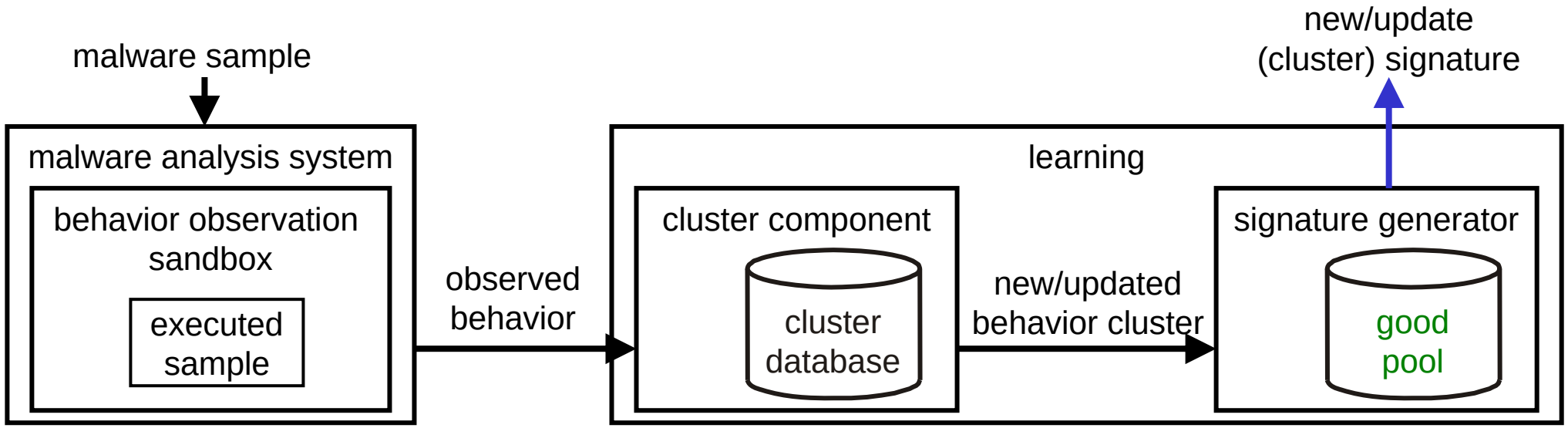
- dynamic analysis
  - dynamic features: behavior observed during execution
    - e.g. trace of systems calls
  - logic bombs
    - difficult to trigger the malicious execution path during analysis

⇨ dynamic analysis more promising for malware analysis and detection

- CWSandbox is used as dynamic analysis system
  - execution in a controlled and monitored environment
  - behavior report: chronologically ordered list of system calls performed by the program during analysis

Automatic Signature Generation

1. group similar malware behavior reports
   - (get the few malware types of the 30.000 malware samples a day)
   - clustering of behavior reports
2. create a signature for each group
   - incorporates behavior reports of known benign programs (good pool) to avoid false positives



Clustering

- requires a distance/similarity metric for program behavior reports
  - candidates, e.g.
    - edit distance
    - normalized compression distance
    - Manhattan distance (n-gram vectorization)
  - based on experimental evaluation [1] we chose Manhattan
- hierarchical clustering algorithms, e.g., single-link, complete-link, WPGMA, UPGMA, fuzzy clustering
  - currently under investigation: complete-link

[1] Measuring Similarity of Malware Behavior. 5th IEEE LCN Workshop on Security in Communications Networks, Oct. 20th 2009, Zurich.

Signature Generation

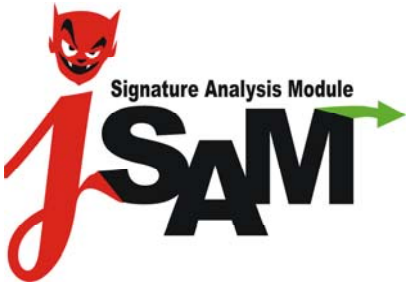
- given a cluster C determine sequences of system calls
  - that are shared among all behavior reports of cluster C
  - but are absent in behavior reports of the good pool

- determine shared substrings using Ukkonen‘s algorithm
- create a signature that matches, if all shared substrings occurred

Malware Detection System

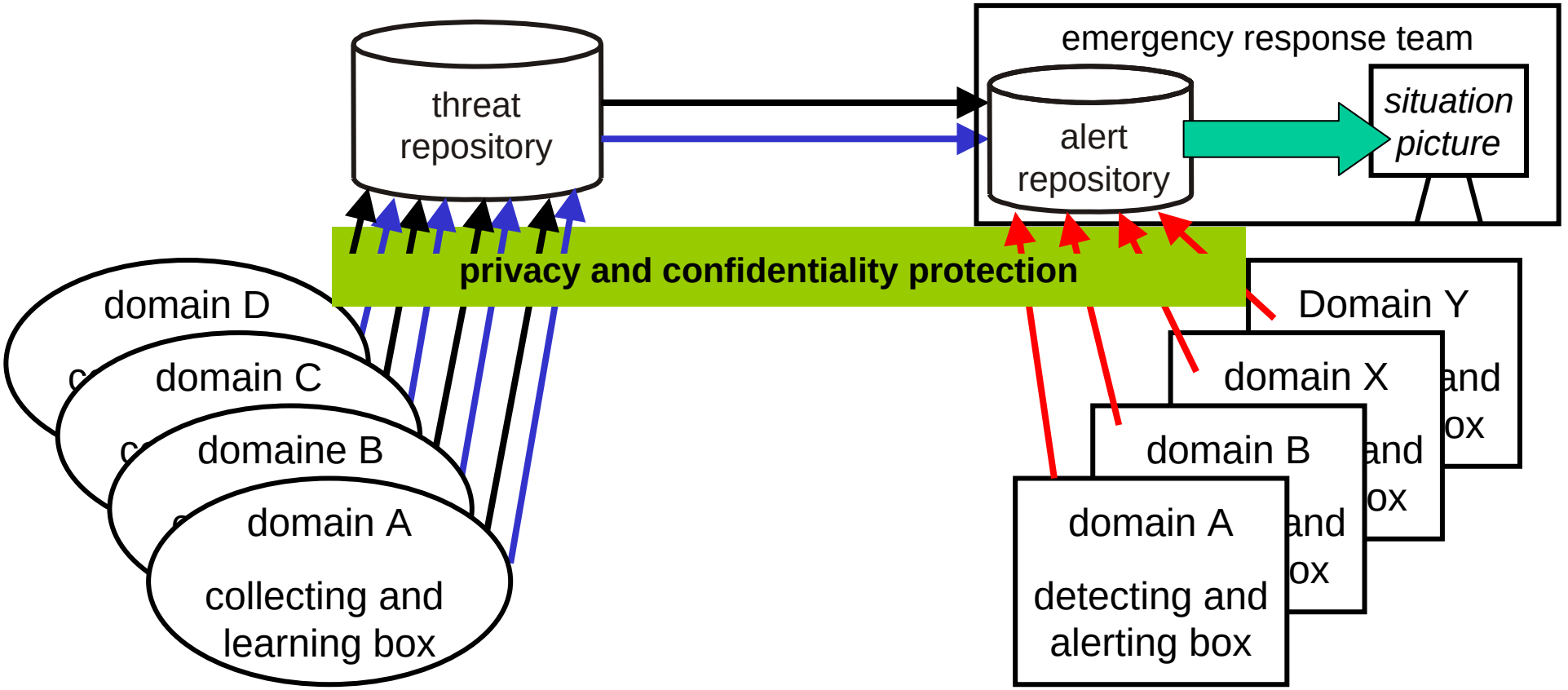
- integration of existing behavior detection systems requires compatible feature domains
  - features extracted using CWSandbox and used for signature generation and features observed/monitored by the detection system need to be compatible
  - signature transformations need to be realized

- new detection systems are developed
  - based jSAM – Java Signature Analysis Module
    - optimized multi-step-signature matching engine
    - expressive signature language EDL (Event Description Language)
    - full support of the behavior features used by EWS supplied signatures



Deployment Scenario

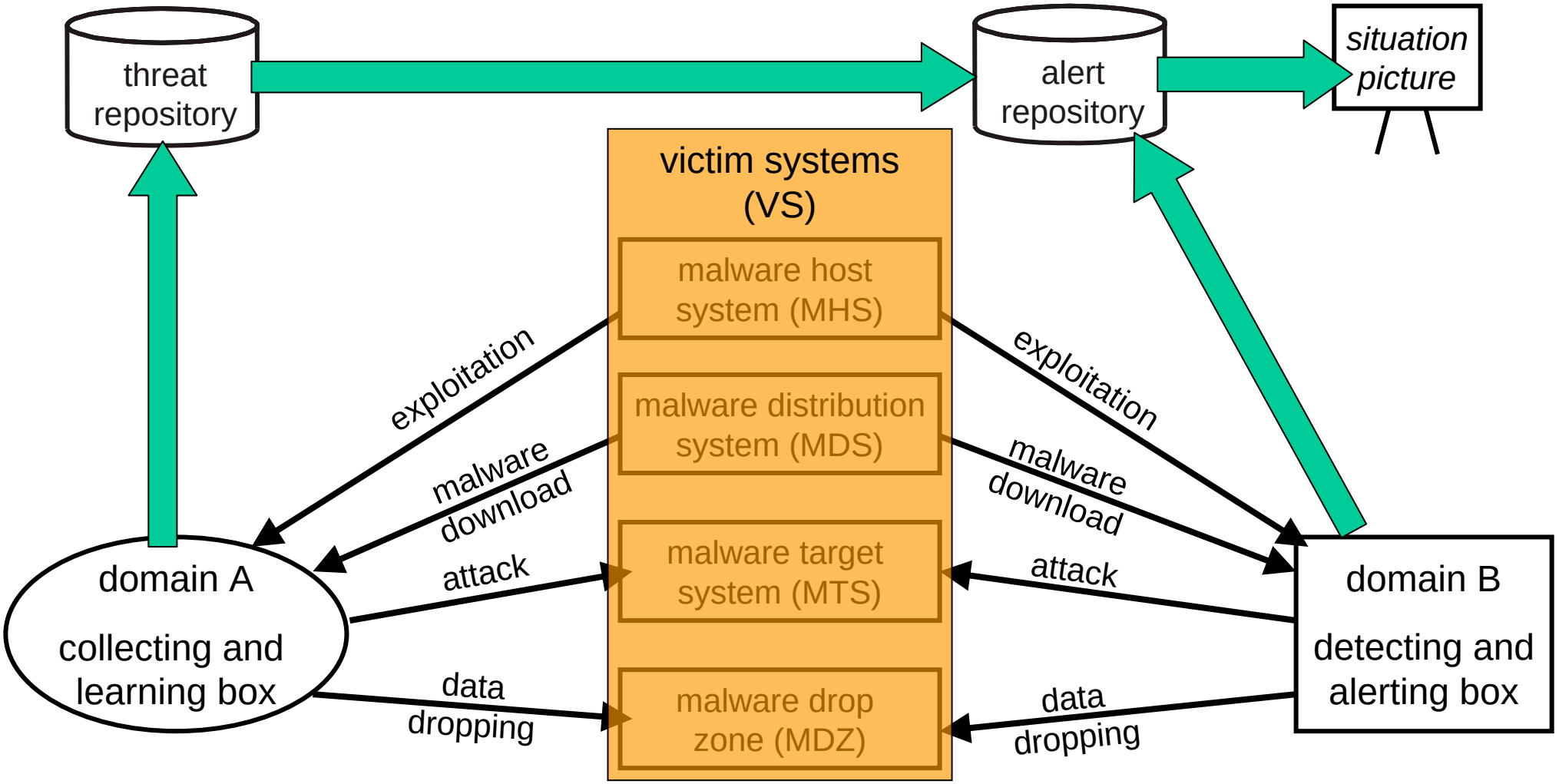
- information exchange
  - private and confidential information
  - allows outsiders (competitors, customers) insights into security incidents



Cooperation Enablement

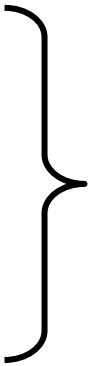
- consideration of conflicting confidentiality and availability interests of participating and involved parities
- resolution of conflicts by use of information reductions, e.g. pseudonymization
- detailed study of
  - flow of information inside the EWS
  - participating and involved parties and their interest wrt. to particular information
  - ⇨ two classes of EWS functionality
    - **analysis**: requires linkability of information
      - are two ip addresses equal?
    - **reaction**: requires disclosure of original information
      - block this ip address

Information Flow



Exchanged Information

- timestamp
- alert signature name
- sending endpoint of MHS
- receiving endpoint of MTS
- download endpoint of MDS
- receiving endpoint of MDZ
- vulnerability module name
- receiving endpoint of CL box
- observing endpoint of DA box
- malware exploit payload
- malware sample payload



personal data of victim systems

Participating and Involved Parties

- collecting and learning box
- detecting and learning box
- threat repository
- alert repository
- victim systems
  - malware host system
  - malware distribution system
  - malware target system
  - malware drop zone

Conflicting Interests (Examples)

- victim systems
  - want to keep their endpoints confidential
- collecting and learning boxes
  - want to keep their existence confidential
- threat repository
  - needs to disclose endpoints of MDZ and MDS for blacklisting sites involved in an malware outbreak
- alert repository
  - need to link all data to create a situation picture

⇨ defining a suitable balance between conflicting interests

- in some cases a given interest is only supported for repositories and not for box owners
- confidentiality interest of VS is only partially supported – box and repository owners can link and disclose information in most cases

⇨ pseudonymization techniques are used for tailoring linkable or disclosable pseudonyms

Summary

- architecture of an automatic EWS
- existing approaches are used for malware collection and analysis
- focus of our ongoing research
  - clustering of malware behavior
  - generating behavior signatures
  - balancing conflicting availability and confidentiality requirements

Thank You!

Optimization

- clustering and signature generation are time-consuming
- for each new malware behavior
  - check if existing signature matches
  - if no signature matches
    - determine cluster closest to the new behavior
    - add new behavior to closest cluster
    - generate new signature for this cluster
- complete re-clustering is performed periodically