isr institute for SOFTWARE RESEARCH

Towards Rapid Re-Certification Using Formal Analysis

Daniel Smullen

Travis Breaux

Carnegie Mellon University

Outline

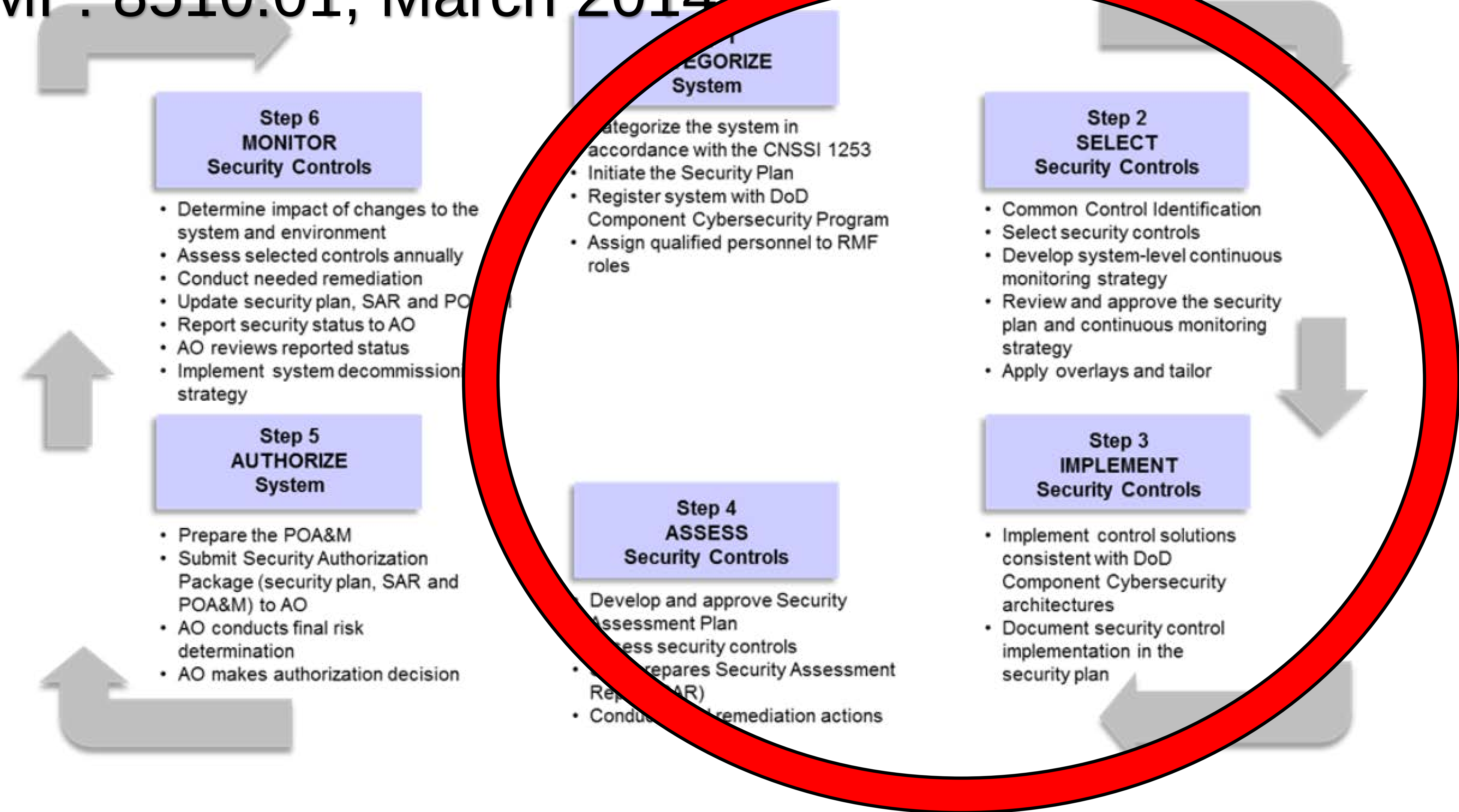
1. Problem Overview
   - Why is software (re)certification hard?
   - What's the risk?
2. What kind of solution is needed?
3. Technical Background
4. Approach, Running Example
   - Conflict Detection, Reconciliation
5. Recertification Triggers
6. Does it scale?
7. Future Work

Why is software (re)certification hard?

- Systems change, requirements evolve.
- As changes occur, how do we determine how the changes affect security?
  - Review, review, then review some more.

- DIACAP, -RMF for IS and PIT systems mandates continuous review process…
- Reviews require **time**, **expertise**, **manpower**, **money**.

RMF: 8510.01, March 2014



EGORIZE
System

- ategorize the system in accordance with the CNSSI 1253
- Initiate the Security Plan
- Register system with DoD Component Cybersecurity Program
- Assign qualified personnel to RMF roles

Step 2
SELECT
Security Controls

- Common Control Identification
- Select security controls
- Develop system-level continuous monitoring strategy
- Review and approve the security plan and continuous monitoring strategy
- Apply overlays and tailor

Step 3
IMPLEMENT
Security Controls

- Implement control solutions consistent with DoD Component Cybersecurity architectures
- Document security control implementation in the security plan

Step 4
ASSESS
Security Controls

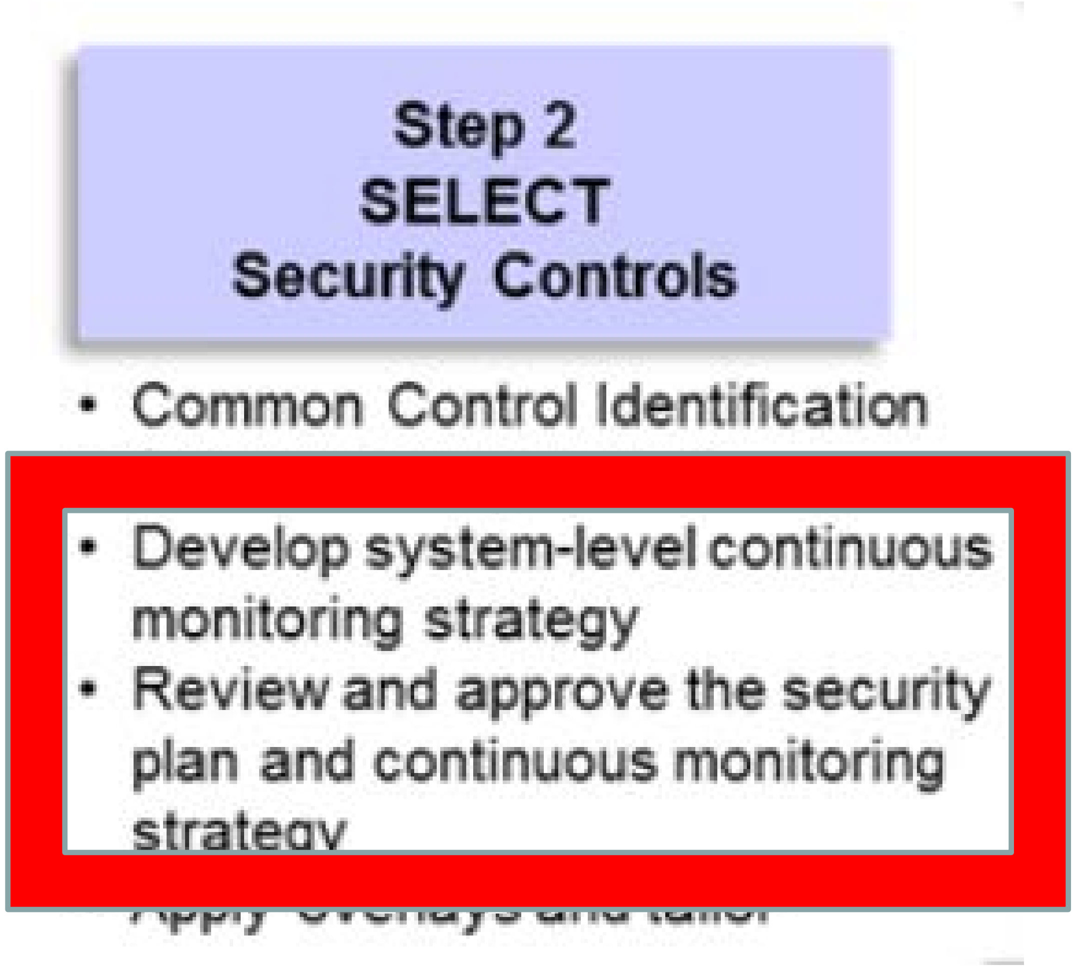
- Develop and approve Security Assessment Plan
- ssess security controls
- repares Security Assessment Rep AR)
- Condu remediation actions

Step 5
AUTHORIZE
System

- Prepare the POA&M
- Submit Security Authorization Package (security plan, SAR and POA&M) to AO
- AO conducts final risk determination
- AO makes authorization decision

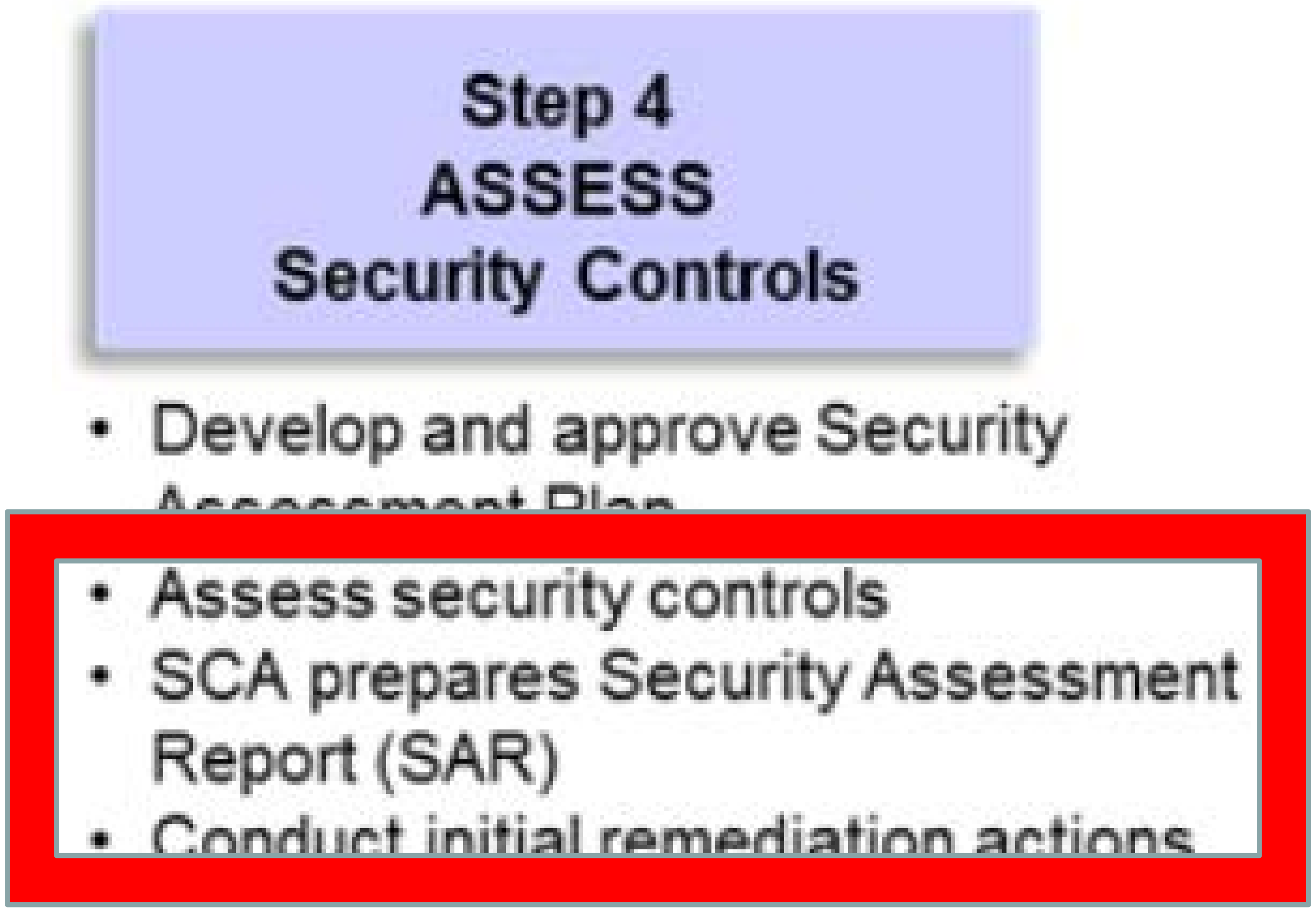
Step 6
MONITOR
Security Controls

- Determine impact of changes to the system and environment
- Assess selected controls annually
- Conduct needed remediation
- Update security plan, SAR and PO
- Report security status to AO
- AO reviews reported status
- Implement system decommission strategy



Step 2
SELECT
Security Controls

- Common Control Identification
- Develop system-level continuous monitoring strategy
- Review and approve the security plan and continuous monitoring strategy
- Apply overlays and tailor



Step 4
ASSESS
Security Controls

- Develop and approve Security Assessment Plan
- Assess security controls
- SCA prepares Security Assessment Report (SAR)
- Conduct initial remediation actions

Assess, review, remediate… rinse, repeat…

- Good in theory, but in practice? Everything is done manually; i.e. slowly.

- Cannot scale as complexity increases.

- Mobile? Cloud-based platforms?
- Constant change.
- Constantly increasing complexity.

What’s the risk?

- Fast and loose: **data spills**.
  - Quick and dirty, miss critical faults.
- Slow and steady: **lose agility**.
  - Must avoid review “backlog mission impossible”.
  - Adversaries will roll out new systems faster than us.
- Can’t just throw more experts at the problem…
  - Brooks’ Law.
  - Too many cooks! Increases accidental complexity.
  - “9 women can’t make a baby in 1 month!”

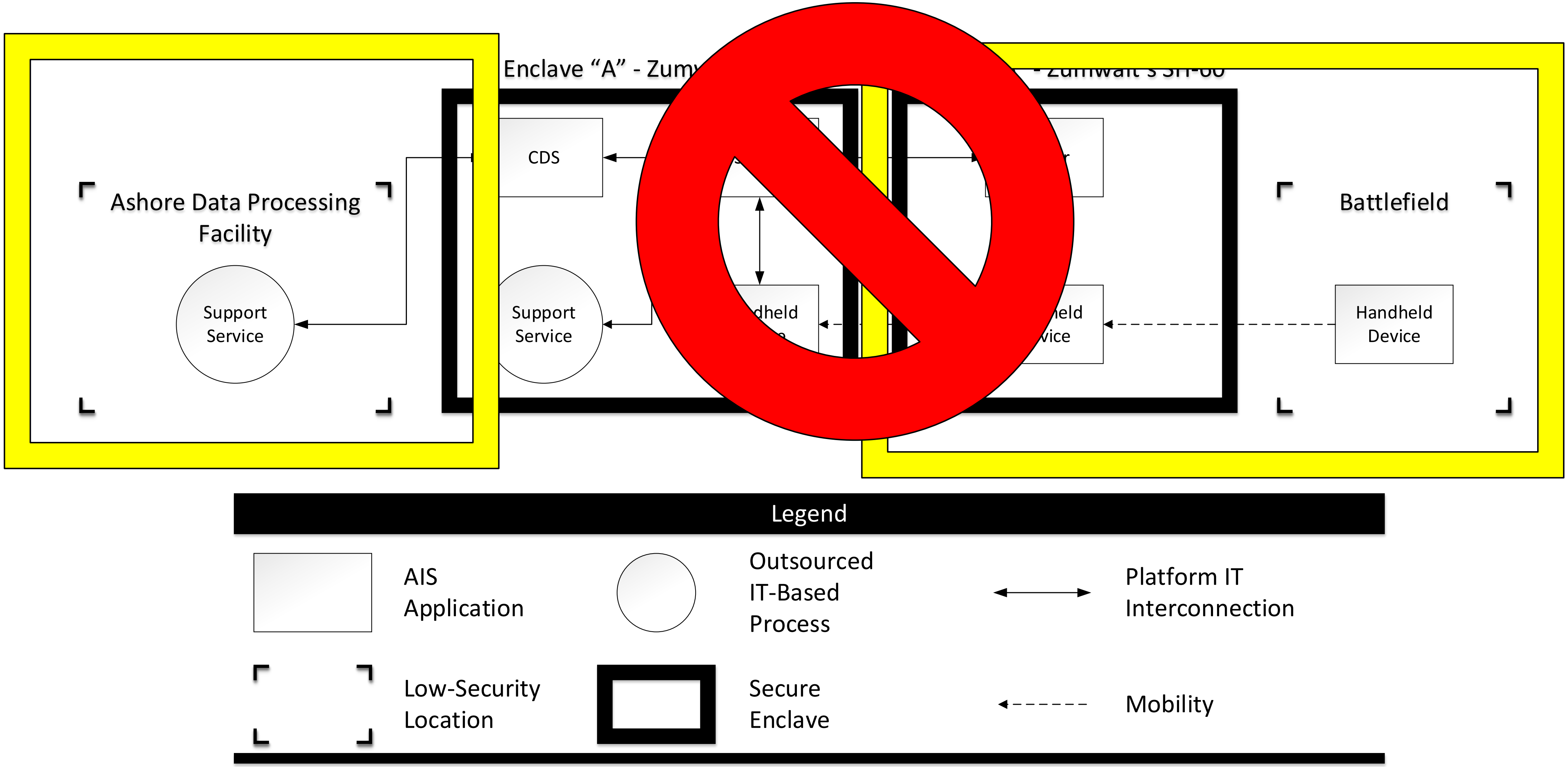
What kind of solution is needed?

- Use automation.
- Scale with evolving architectural assumptions.
- Do analysis computationally.
- Focus on adding new features, let the analysis determine the impact.

- **Result: Rapid analysis at recertification (or design) time.**

- Focus on the parts that commensurate with risk:
  - Data.
  - Secure enclave boundaries.
  - Changes.

What parts do we focus on?



Enclave “A” - Zum

- Zumwalt’s SH-60

Ashore Data Processing Facility

Support Service

CDS

Support Service

dheld

ield vice

Battlefield

Handheld Device

Legend

AIS Application

Outsourced IT-Based Process

Platform IT Interconnection

Low-Security Location

Secure Enclave

Mobility

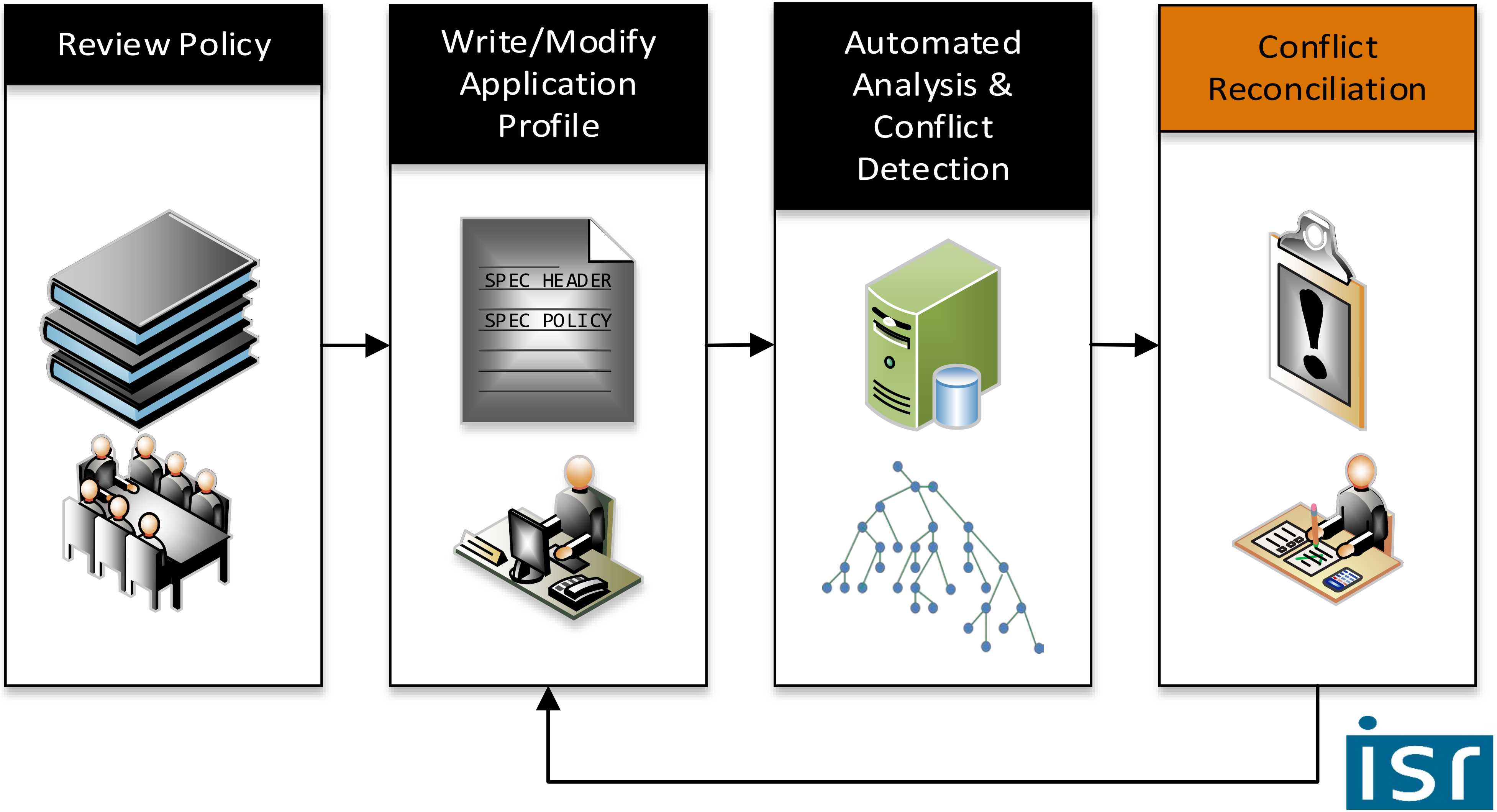
Technical Background

- Application Profile Language, model-checking.
- Semantic parameterization (Breaux et al., 2008)
  - Actions on data; actors, objects, purposes, source, destination.

- Bell-LaPadula: high-, low-confidentiality.
- Characterize the *purpose*; security level.
- Express compositions; logical subsumption.
  - Containment
  - Disjointness

- **This forms the basis for our application profile language.**

Technical Background



Review Policy

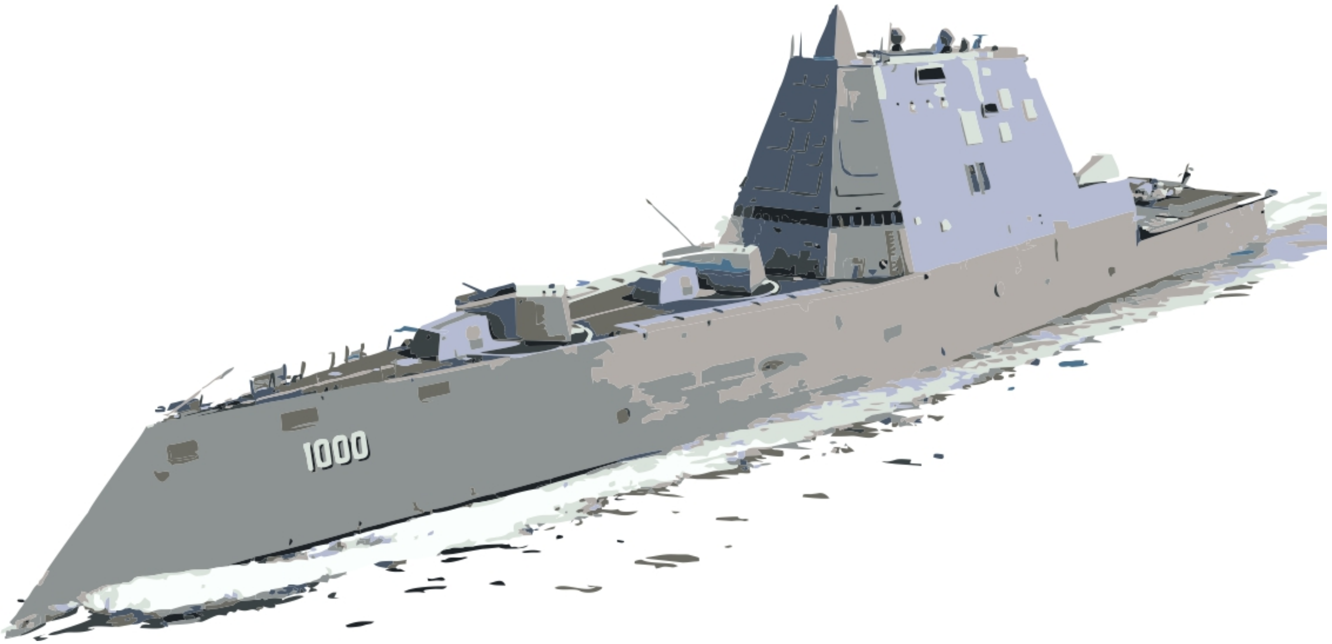
Write/Modify Application Profile

Automated Analysis & Conflict Detection

Conflict Reconciliation

Running Example

- Public accounts of real-world ship.
- Zumwalt-class destroyer.
- TSCE Infrastructure
- 6 MLOC
- Focus on software requirements:
  - Sensory and information sharing capabilities.



02-92-0-C
FR92-103
OP1
Surface Dominance - Undersea Warfare
Land Attack Warfare
Air Dominance

Approach

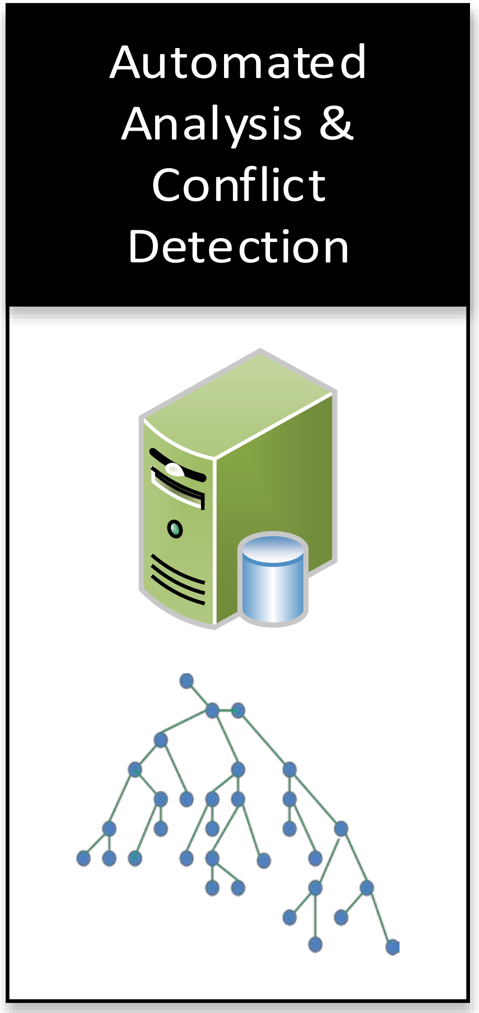
- Application profiles
  - Actions on data:
    - Collection
    - Use
    - Transfer
  - Traces:
    - Collection-Use
    - Collection-Transfer
    - Vice-versa

Write/Modify Application Profile

SPEC HEADER
SPEC POLICY

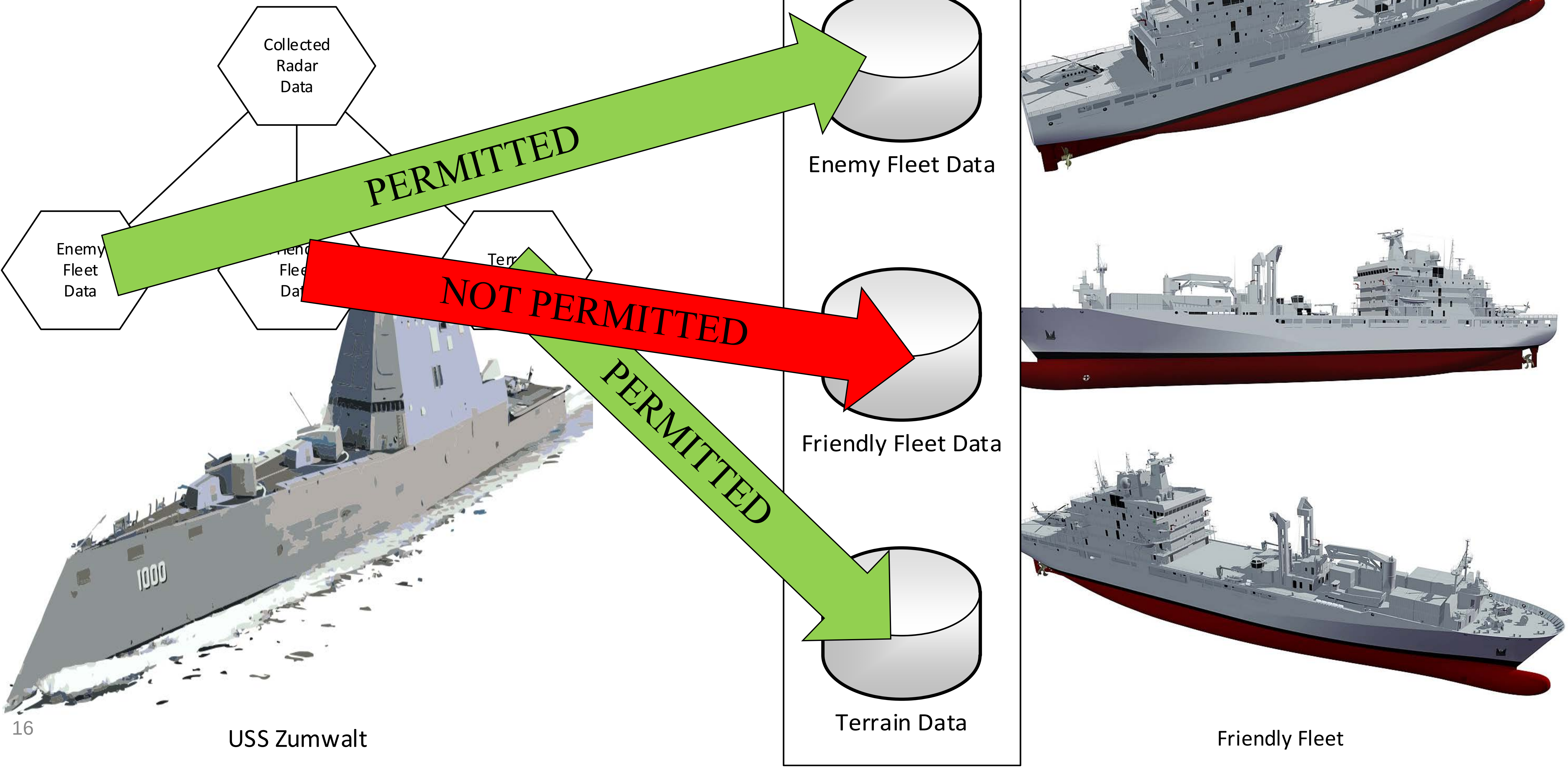
Approach

- Conflict Detection
    - Policy may specify a prohibition and a right on the same data, for the same purpose.
    - Leads to conflict.



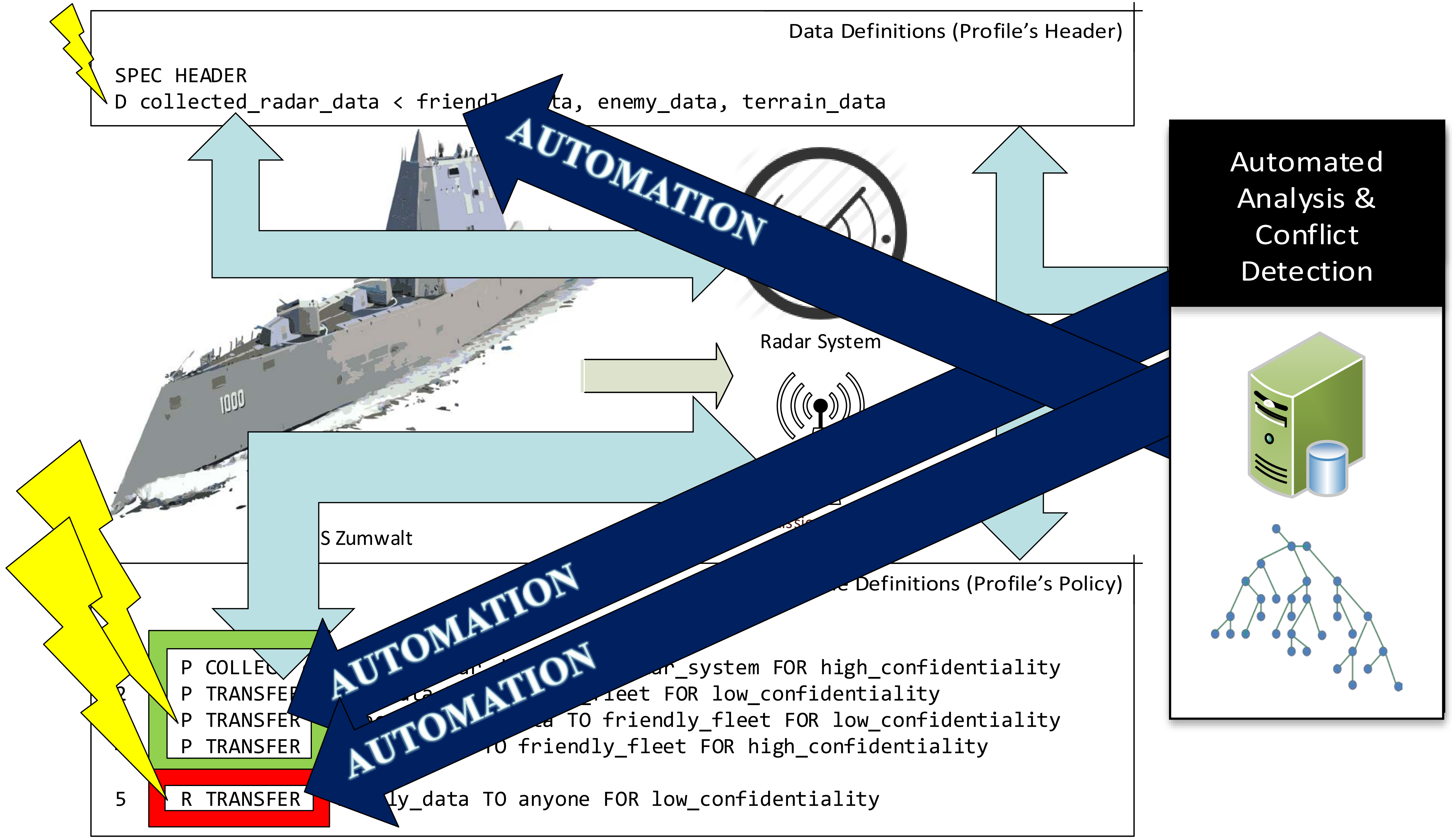
Automated Analysis & Conflict Detection

```
D collected_radar_data <
friendly_data, enemy_data,
terrain_data
```



USS Zumwalt

Friendly Fleet



Data Definitions (Profile’s Header)

```
SPEC HEADER
D collected_radar_data < friendl  ta, enemy_data, terrain_data
```

e Definitions (Profile’s Policy)

```
  P COLLEC                    ar_system FOR high_confidentiality
2 P TRANSFER                 _fleet FOR low_confidentiality
  P TRANSFER           a TO friendly_fleet FOR low_confidentiality
  P TRANSFER          TO friendly_fleet FOR high_confidentiality

5 R TRANSFER     ly_data TO anyone FOR low_confidentiality
```

1. Permit collection of collected radar data from Zumwalt's radar system, designating it as high-confidentiality data.

Application Profile Language	Formalization in Description Logic
`P COLLECT collected_radar_data FROM radar_system FOR high_confidentiality`	$T \models p_0 \equiv \mathit{COLLECT} \sqcap \exists \text{has}\mathit{Object}.\ \text{collected\_radar\_data} \sqcap \exists \mathit{hasSource}.\ \text{radar\_system} \sqcap \exists \mathit{hasPurpose}.\ \text{high\_confidentiality}$

2. Permit transfer of data about enemy vessels to friendly fleet members for general, low-confidentiality purposes.

Application Profile Language	Formalization in Description Logic
`P TRANSFER enemy_data TO friendly_fleet FOR low_confidentiality`	$T \models p_1 \equiv \text{TRANSFER} \sqcap \exists \text{has}\mathit{Object}.\ \text{enemy\_data} \sqcap \exists \mathit{has}\text{Target}.\ \text{radar\_system} \sqcap \exists \mathit{hasPurpose}.\ \text{low\_confidentiality}$

3. Permit transfer of all collected radar data to friendly fleet members for general, low confidentiality purposes. *This rule generates a conflict, which is explained below.*

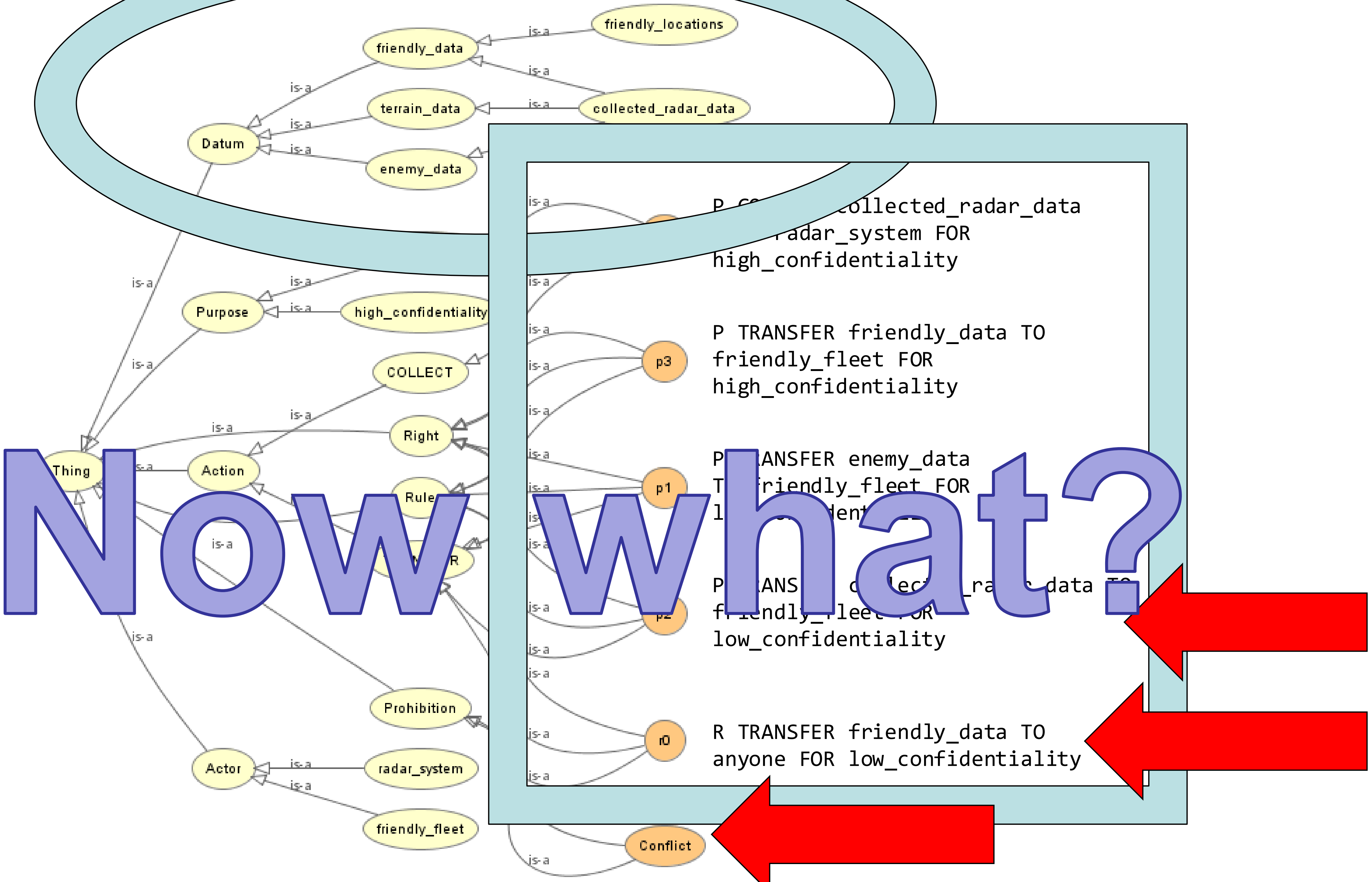
Application Profile Language	Formalization in Description Logic
`P TRANSFER collected_radar_data TO friendly_fleet FOR low_confidentiality`	$T \models p_2 \equiv \text{TRANSFER} \sqcap \exists \text{has}\mathit{Object}.\ \text{collected\_radar\_data} \sqcap \exists \mathit{has}\text{Target}.\ \text{friendly\_fleet} \sqcap \exists \mathit{hasPurpose}.\ \text{low\_confidentiality}$

4. Permit transfer of data about friendly vessels to friendly fleet members for specific, high-confidentiality purposes.

Application Profile Language	Formalization in Description Logic
`P TRANSFER friendly_data TO friendly_fleet FOR high_confidentiality`	$T \models p_3 \equiv \text{TRANSFER} \sqcap \exists \text{has}\mathit{Object}.\ \text{friendly\_data} \sqcap \exists \mathit{has}\text{Target}.\ \text{friendly\_fleet} \sqcap \exists \mathit{hasPurpose}.\ \text{high\_confidentiality}$

5. Prohibit transfer of friendly fleet data to anyone for general, low confidentiality purposes. *This rule conflicts with Rule 3, explained below.*

Application Profile Language	Formalization in Description Logic
`R TRANSFER friendly_data TO anyone FOR low_confidentiality`	$T \models \text{r}_0 \equiv \text{TRANSFER} \sqcap \exists \text{has}\mathit{Object}.\ \text{collected\_radar\_data} \sqcap \exists \mathit{has}\text{Target}.\ \text{Actor} \sqcap \exists \mathit{hasPurpose}.\ \text{low\_confidentiality}$



P COLLECT collected_radar_data FROM radar_system FOR high_confidentiality

P TRANSFER friendly_data TO friendly_fleet FOR high_confidentiality

P TRANSFER enemy_data TO friendly_fleet FOR low_confidentiality

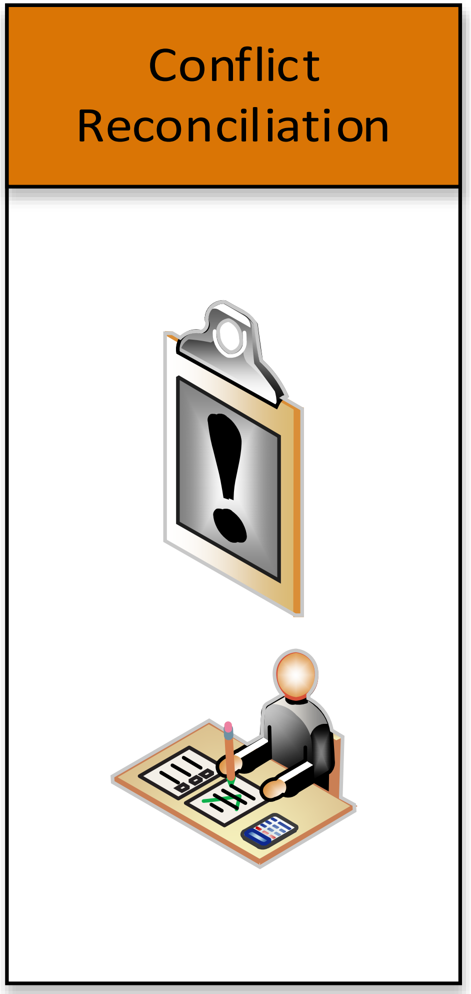
P TRANSFER collected_radar_data TO friendly_fleet FOR low_confidentiality

R TRANSFER friendly_data TO anyone FOR low_confidentiality

Now what?

Reconciliation

- Two reconciliation approaches identified:
  - Redaction
  - Generalization
- One approach that defeats these measures:
  - Merging

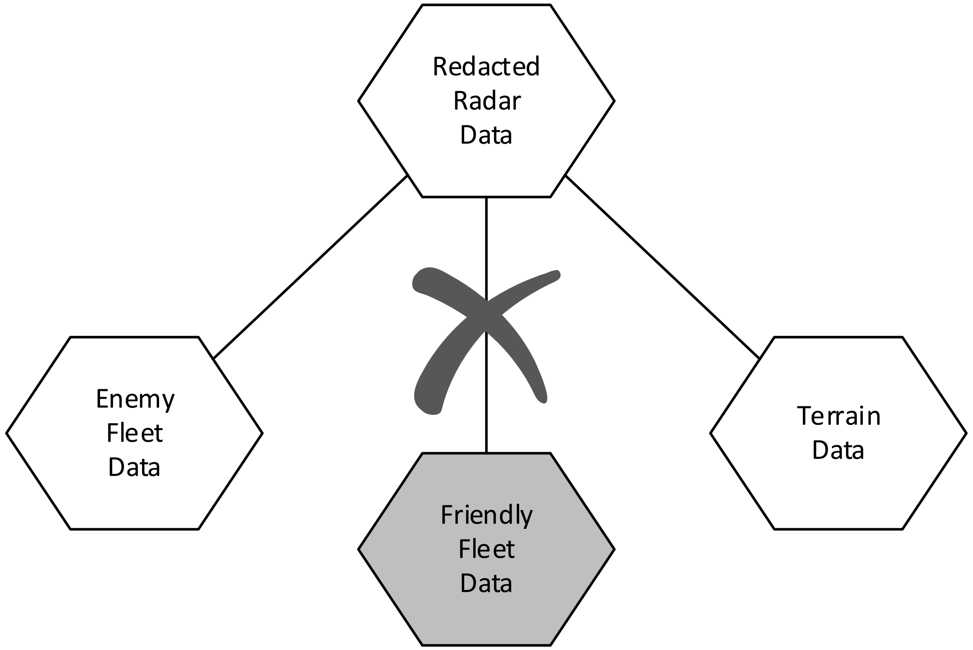


Conflict Reconciliation

Redaction

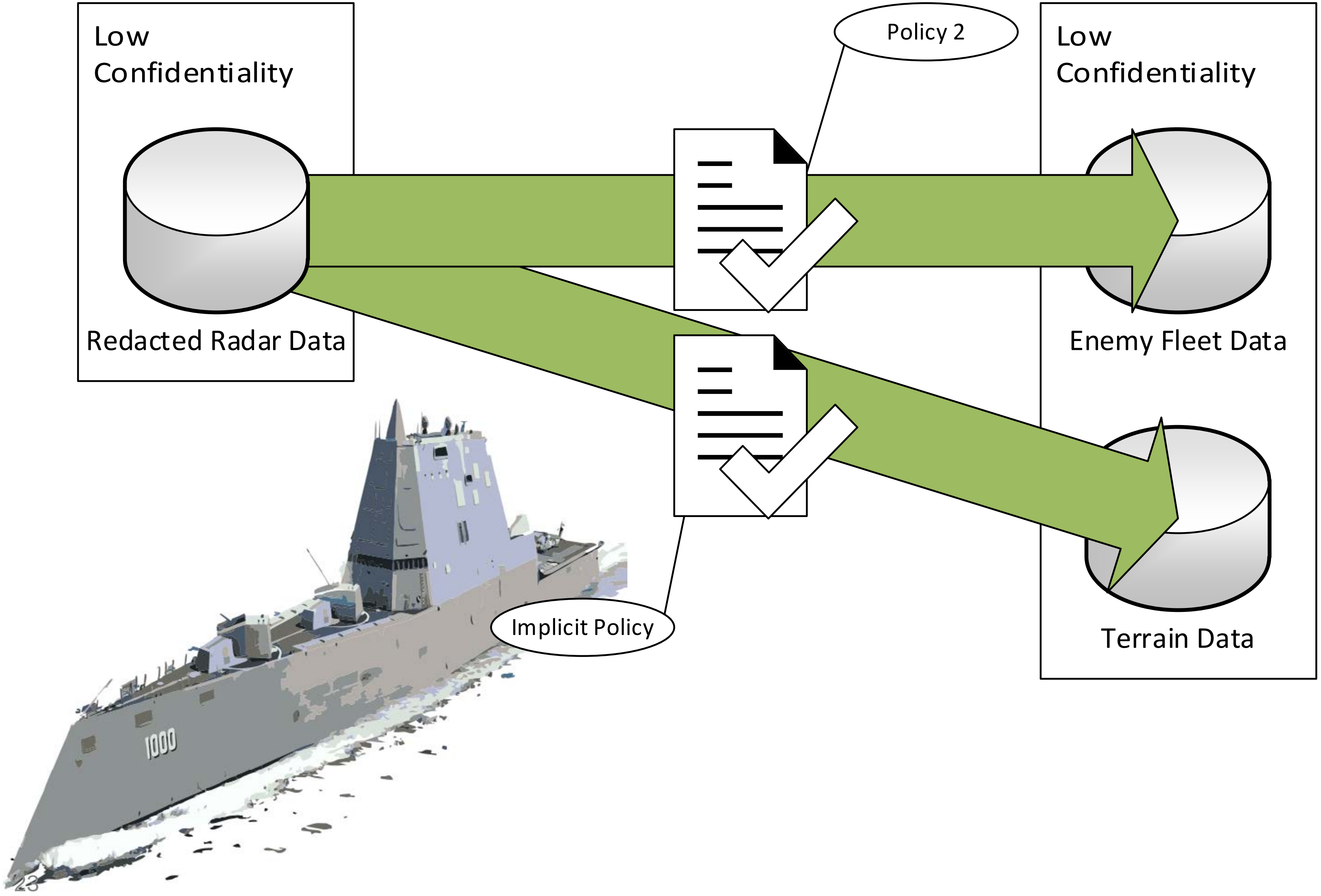
- Eliminate a subsumption relationship within a collection.
- Permits the new (redacted) collection to be used for low-confidentiality purposes.

```
D redacted_radar_data <
enemy_fleet_data, terrain_data
```

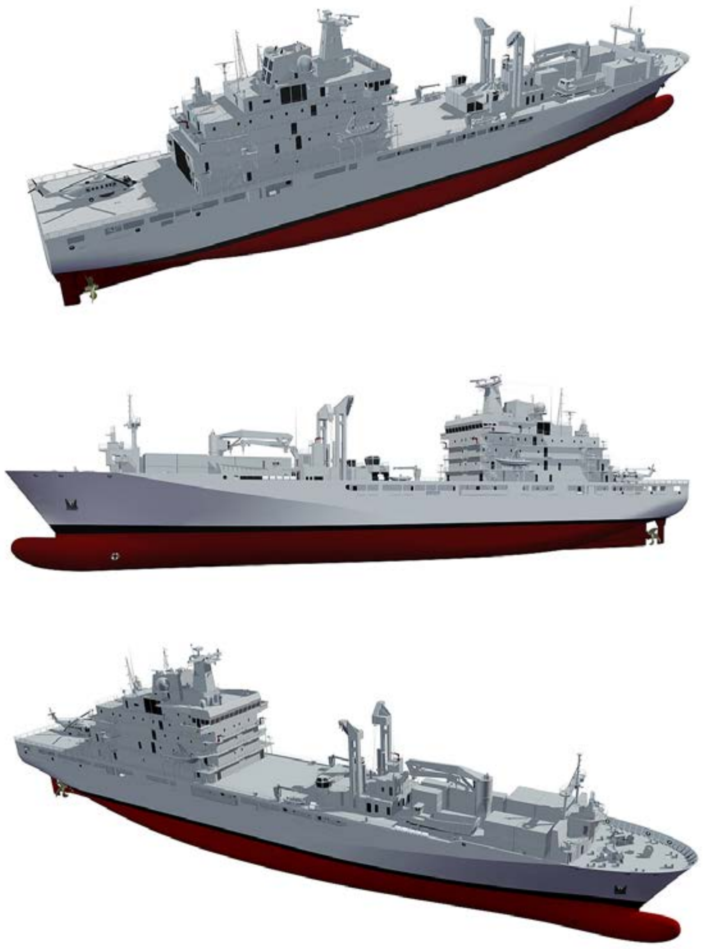


Redaction

```
        SPEC POLICY
1       P COLLECT collected_radar_data FROM radar_system FOR high_confidentiality
2       P TRANSFER enemy_data TO friendly_fleet FOR low_confidentiality

        REDACT(collected_radar_data -> redacted_radar_data, friendly_data,
        low_confidentiality)

3       P TRANSFER redacted_radar_data TO friendly_fleet FOR low_confidentiality
4       P TRANSFER friendly_data TO friendly_fleet FOR high_confidentiality

5       R TRANSFER friendly_data TO anyone FOR low_confidentiality
```



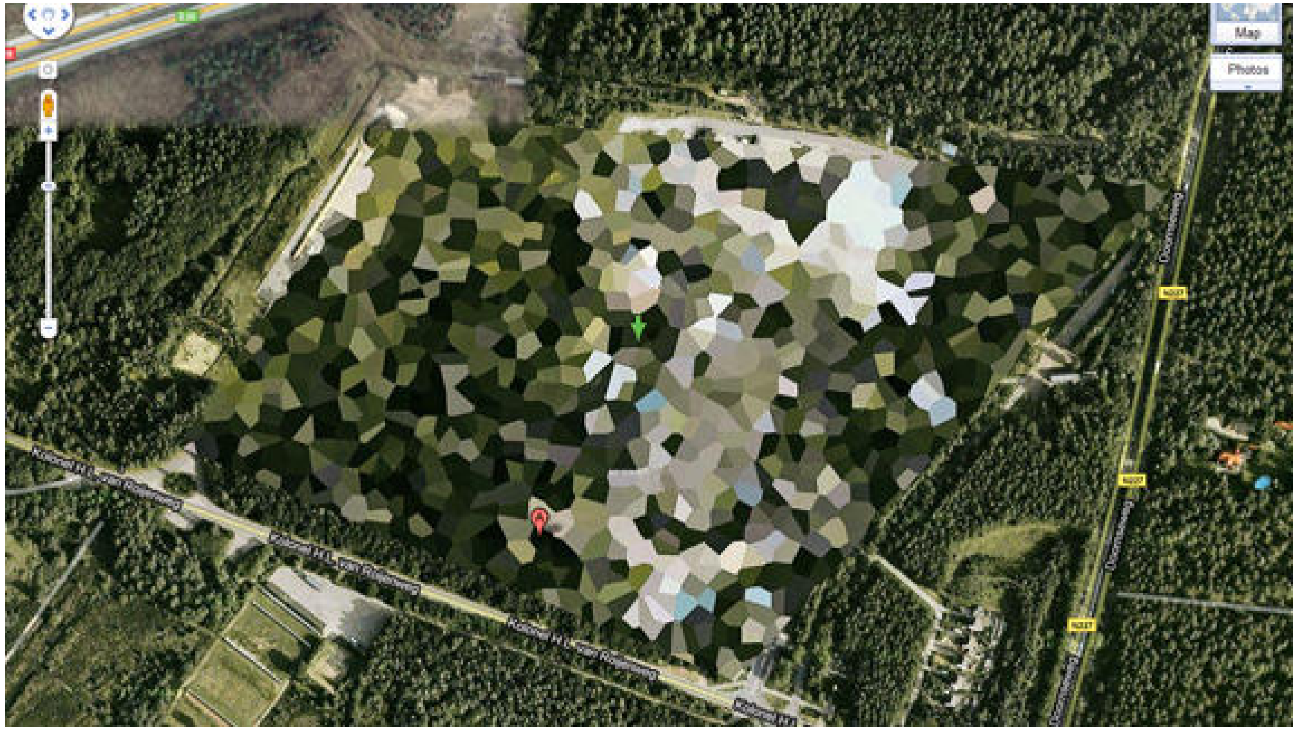
USS Zumwalt



Friendly Fleet

Generalization

- Some types of data can be **fuzzified**.
  - Add noise, decrease fidelity.
- Numerical data:
  - Coordinates, time…
- All collections’ members must be generalized.



Merging

- Combine redacted data with un-redacted to recreate original.
- Combine generalized data with **de-noised** data to recreate original.

Distinguishing the Merging Risk

Policy Violation

1. Collect data for **high-confidentiality** purpose.
2. Collect other data for **low-confidentiality** purpose.
3. Repurpose high-confidentiality data, violate policy.

Merging

1. Collect data for **low-confidentiality** purpose.
   - Data is subset of redacted superset.
2. Collect related data for **low-confidentiality** purpose.
   - Data is negation of superset and redacted superset.
3. Merge two disjoint collections.

Similarly purposed data flows may be merged.

Merging Risk Mitigation

- Can catch merging risks as a result of conflict analysis.
  - Check subsumed purposes.
  - Trace data flows, transfer only what data is needed.

- Mitigates human error due to missed interpretations.

Recertification Triggers

How do you know when to run the analysis?

- Reconcile a conflict? Rerun, recheck.
- Add a new feature? Rerun, recheck.
- Modify the policy? Rerun, recheck.

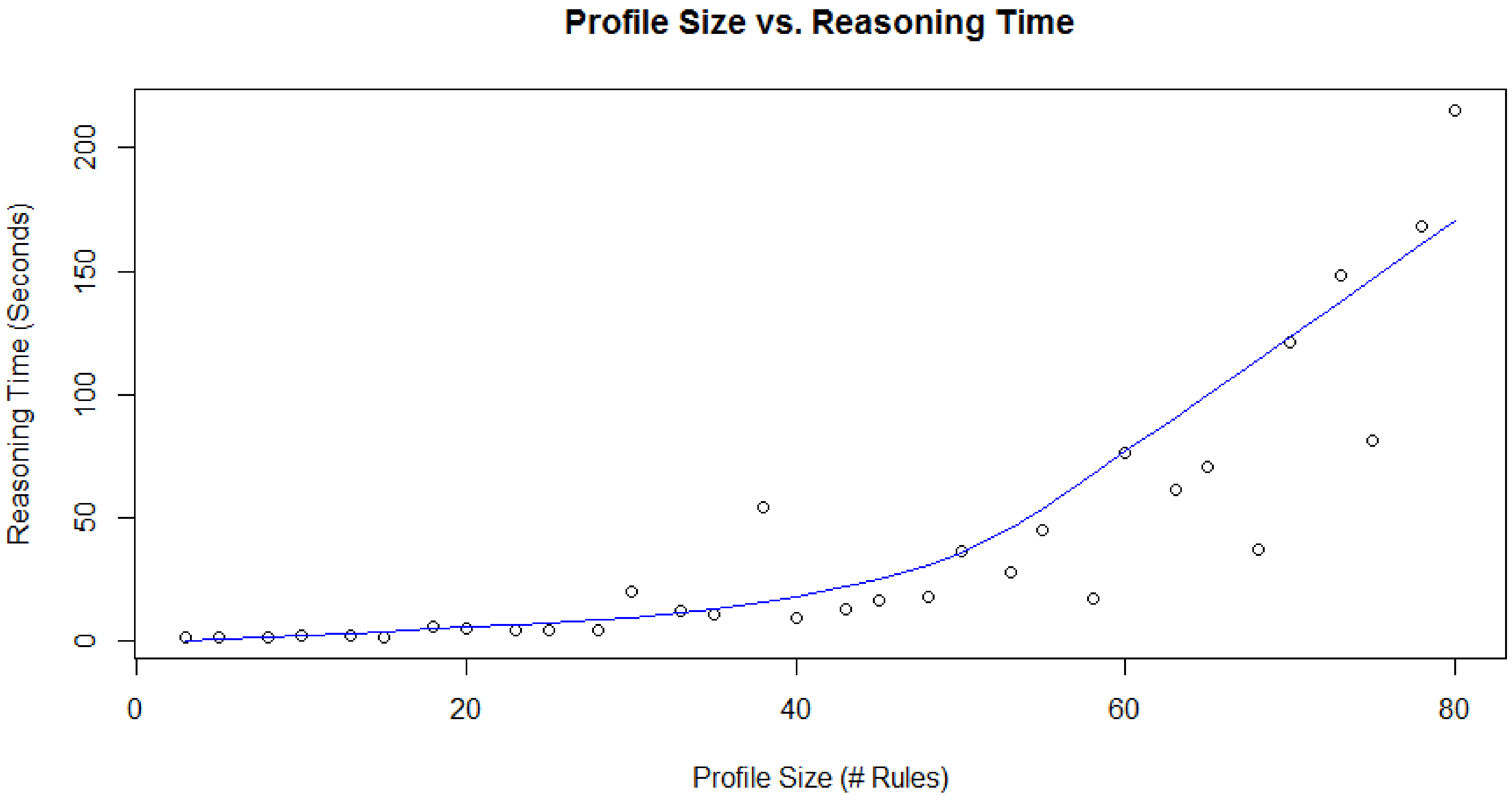
- Rapid analysis means recertification is rapid.

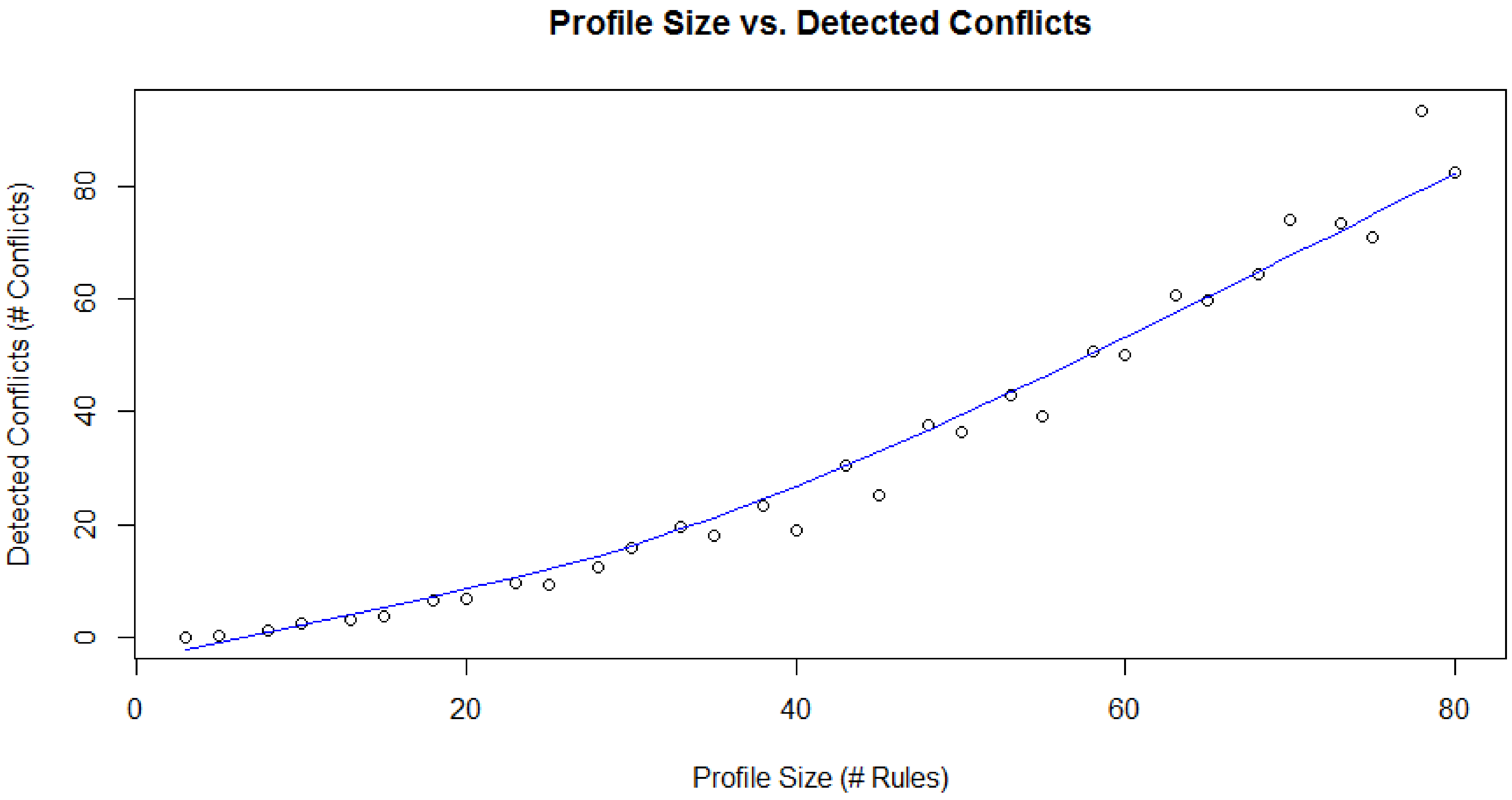
Does it scale?

- How fast can we do analysis? Is it fast enough to let us rerun whenever we want?

- Simulations; 27 repetitions, increasing number of rules [0-80], 1.13 conflicts per increasing rule.

No objective basis for comparison.


Profile Size vs. Reasoning Time
Reasoning Time (Seconds)
0
50
100
150
200
0
20
40
60
80
Profile Size (# Rules)


Profile Size vs. Detected Conflicts
Detected Conflicts (# Conflicts)
0
20
40
60
80
0
20
40
60
80
Profile Size (# Rules)

Does it scale?

- No statistically significant relationship between performance and number of conflicts.
  $\{\underline{r}(874) = .36, \underline{p} > .05\}$

Average Profile Parsing Time	**<1 second**
Largest Profile Size	80 rules
Longest Profile Processing Time	**400 seconds**
Average Conflicts per Statement	1.13

Conclusions

- Yes, it scales:
  - Analysis can scale in quasilinear time.
- Simulations show that even huge profiles can be analyzed in roughly 7 minutes.

- What do we mean by huge profiles?
  - Hundreds of data flows.
  - Hundreds of rule combinations.
  - Hundreds of conflicts.

Future Work

- Extend automation to provide “hints” to analysts.
  - Profile development environment.
  - Automate reconciliation strategies.

- Characterize performance gain against manual processes.

Carnegie Mellon University

Questions?

- Daniel Smullen

Graduate Research Assistant, Carnegie Mellon University

dsmullen@cs.cmu.edu

- Travis Breaux

Assistant Professor, Carnegie Mellon University

breaux@cs.cmu.edu

isr institute for SOFTWARE RESEARCH