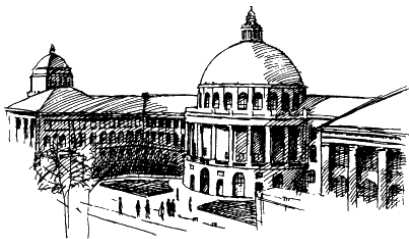


Policy Monitoring in First-order Temporal Logic

David Basin
ETH Zurich



In collaboration with



Felix Klaedtke



Samuel Müller



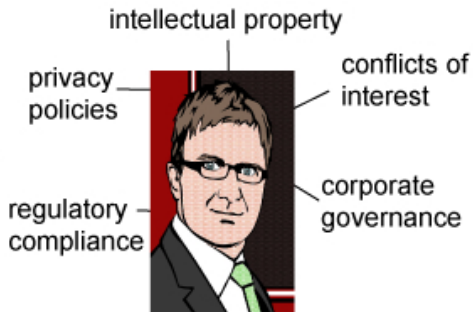
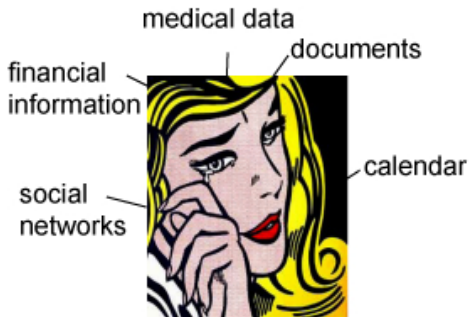
Matúš Harvan



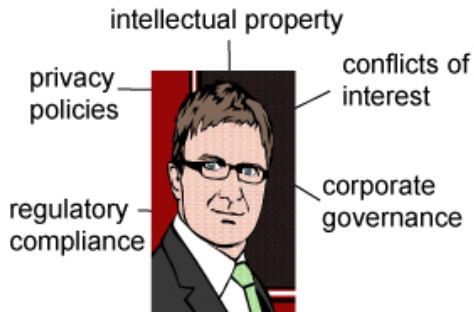
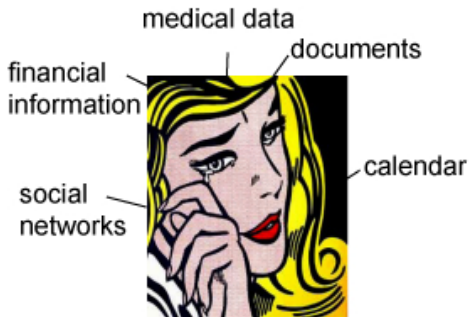
Eugen Zălinescu

NOKIA

Modern problems

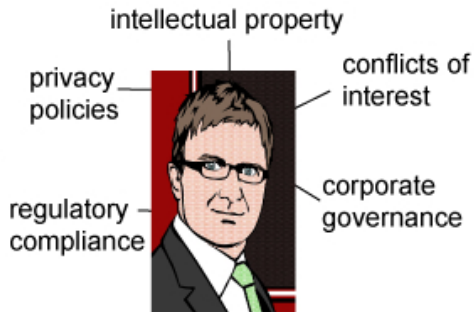
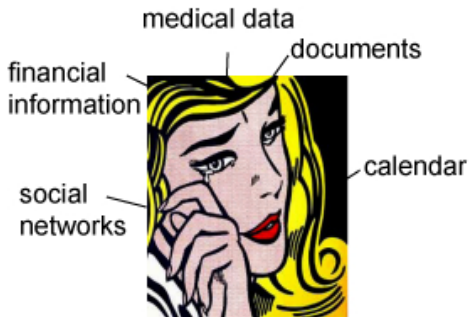


Modern problems



What do these topics have to do with each other?

Modern problems



**What do these topics have to do with each other?
Are they theoretically interesting?**

Technical issues

Processes to monitor and control processes

- ▶ Controlling access
My medical data should only be accessible to my care givers.
- ▶ Controlling usage
... and then used for intended purpose, e.g., improving healthcare
- ▶ Corporate governance and regulatory compliance
Implement controls to reduce risks.

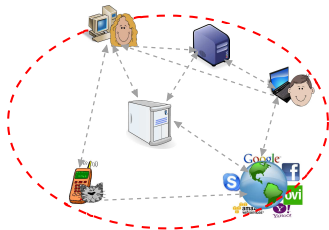
Technical issues

Processes to monitor and control processes

- ▶ Controlling access
My medical data should only be accessible to my care givers.
- ▶ Controlling usage
... and then used for intended purpose, e.g., improving healthcare
- ▶ Corporate governance and regulatory compliance
Implement controls to reduce risks.

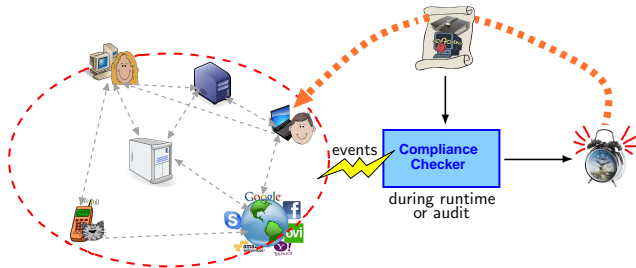
Core problems are theoretically interesting!
So are system aspects.

Focus



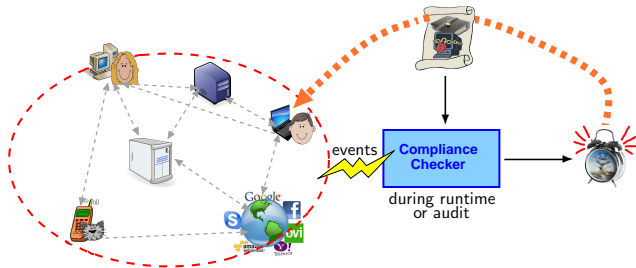
- ▶ Setting: security and compliance
 - Business processes
 - Policies regulating data and processes

Focus



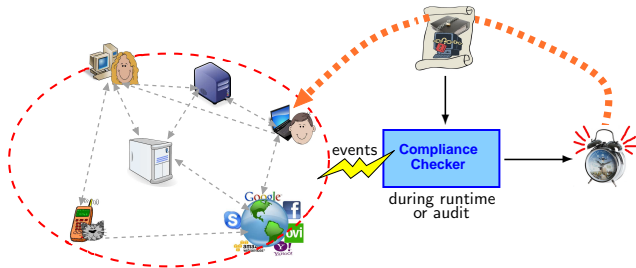
- ▶ Setting: security and compliance
 - Business processes
 - Policies regulating data and processes
- ▶ Monitoring (\neq enforcement)

Focus



- ▶ Setting: security and compliance
 - Business processes
 - Policies regulating data and processes
- ▶ Monitoring (\neq enforcement)
- ▶ General solution using metric first-order temporal logic and an associated monitoring algorithm

Focus



- ▶ Setting: security and compliance
 - Business processes
 - Policies regulating data and processes
- ▶ Monitoring (\neq enforcement)
- ▶ General solution using metric first-order temporal logic and an associated monitoring algorithm
- ▶ Practical experience within Nokia Data Collection Campaign

Road map

1. An example
2. Metric First-order Temporal Logic
3. Formalization examples
4. Monitoring
5. Case study
6. Conclusion

Road map

1. **An example**
2. Metric First-order Temporal Logic
3. Formalization examples
4. Monitoring
5. Case study
6. Conclusion

Example



▶ Consider a financial or research institute:

- Employees write and publish reports
- Reports may contain confidential data

▶ Report approval policy

- 1. Reports must be approved before they are published.*
- 2. Approvals must happen at most 10 days before publication.*
- 3. The employees' managers must approve the reports.*

▶ IT system logs events

```
2010-03-03    publish_report (Charlie, #234)
2010-03-04    archive_report (Alice, #104)
      ⋮                ⋮
2010-03-09    approve_report (Alice, #248)
2010-03-13    publish_report (Bob, #248)
      ⋮                ⋮
```

▶ Are executions policy conform? (answer online or at later audit)

Policy elements

- 1. Reports must be approved before they are published.*
- 2. Approvals must happen at most 10 days before publication.*
- 3. The employees' managers must approve the reports.*

Policy elements

Subjects

- ▶ reports and employees
- ▶ unbounded over time

1. Reports must be approved before they are published.
2. Approvals must happen at most 10 days before publication.
3. The employees' managers must approve the reports.

Policy elements

Subjects

- ▶ reports and employees
- ▶ unbounded over time

1. Reports must be approved before they are published.
2. Approvals must happen at most 10 days before publication.
3. The employees' managers must approve the reports.

Temporal aspects

- ▶ qualitative: before and always
- ▶ quantitative: at most 10 days

Policy elements

Subjects

- ▶ reports and employees
- ▶ unbounded over time

Event predicates

- ▶ approving and publishing a report
- ▶ happen at a time point
- ▶ logged with time stamps

1. Reports must be approved before they are published.
2. Approvals must happen at most 10 days before publication.
3. The employees' managers must approve the reports.

Temporal aspects

- ▶ qualitative: before and always
- ▶ quantitative: at most 10 days

Policy elements

Subjects

- ▶ reports and employees
- ▶ unbounded over time

Event predicates

- ▶ approving and publishing a report
- ▶ happen at a time point
- ▶ logged with time stamps

1. *Reports must be approved before they are published.*
2. *Approvals must happen at most 10 days before publication.*
3. *The employees' managers must approve the reports.*

Temporal aspects

- ▶ qualitative: before and always
- ▶ quantitative: at most 10 days

State predicates

- ▶ being someone's manager
- ▶ have a duration

These aspects can be formalized using MFOTL

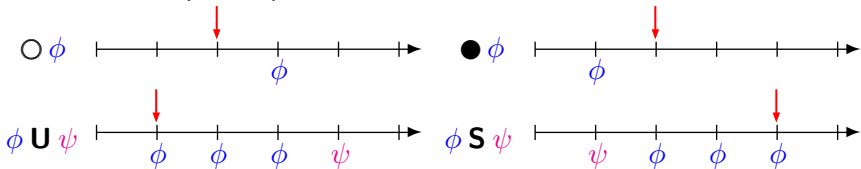
$$\square \forall e. \forall r. \text{publish_report}(e, r) \rightarrow \\ \blacklozenge_{[0,11)} \exists m. \underline{\text{manager}}(m, e) \wedge \text{approve_report}(m, r)$$

- ▶ **First-order** for expressing relations on system data.
- ▶ **Metric temporal operators** for expressing qualitative and quantitative timing information.
- ▶ Can represent both **event** and **state** predicates.

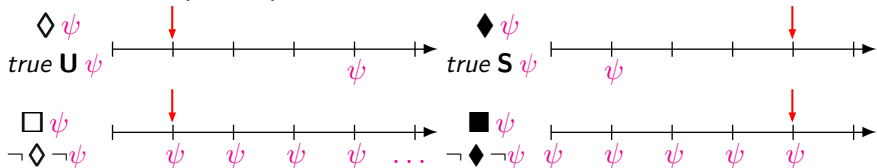
Let's look at this, starting with the temporal operators.

Standard linear temporal operators

▶ Primitive temporal operators

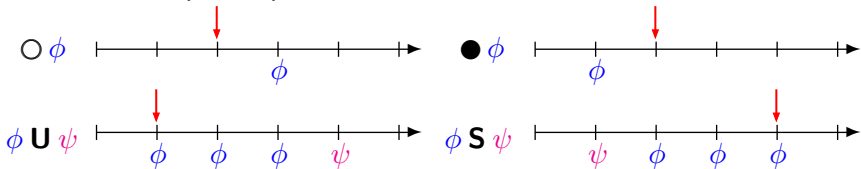


▶ Derived temporal operators

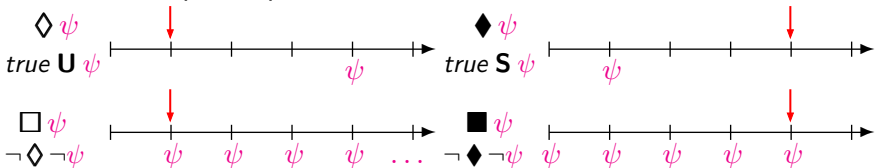


Metric temporal operators

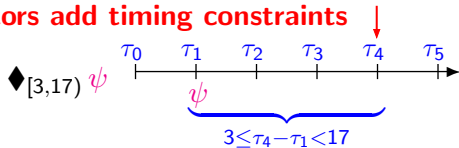
Primitive temporal operators



Derived temporal operators



Metric operators add timing constraints



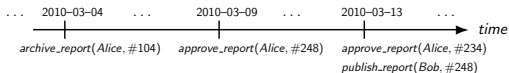
Policy revisited and simplified



1. Reports must be approved before they are published.
2. Approvals must happen at most 10 days before publication.
- ~~3. The employees' managers must approve the reports.~~

- ▶ Publishing and approving events are logged with time stamps

```
2010-03-04  archive_report (Alice, #104)
  ⋮
2010-03-09  approve_report (Alice, #248)
  ⋮
2010-03-13  approve_report (Alice, #234)
2010-03-13  publish_report (Bob, #248)
  ⋮
```



- ▶ Simplified policy in MFOTL:

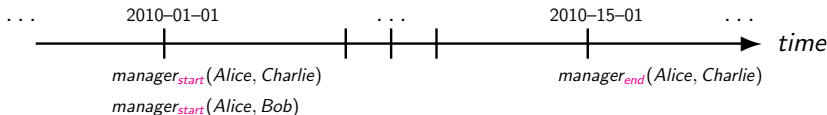
$$\square \forall e. \forall r. \text{publish_report}(e, r) \rightarrow \blacklozenge_{[0,11]} \exists m. \text{approve_report}(m, r)$$

Policy revisited



1. Reports must be approved before they are published.
2. Approvals must happen at most 10 days before publication.
3. The employees' managers must approve the reports.

- ▶ Being someone's manager is a **state property**, with a duration
 - Also log events that mark **start** and **end** points



- State predicate as syntactic sugar
$$\underline{manager}(m, e) := \neg manager_{end}(m, e) \mathbf{S} manager_{start}(m, e)$$

- ▶ Policy in MFOTL

$$\square \forall e. \forall r. \text{publish_report}(e, r) \rightarrow$$
$$\blacklozenge_{[0,11]} \exists m. \underline{manager}(m, e) \wedge \text{approve_report}(m, r)$$

Road map

1. An example
2. **Metric First-order Temporal Logic**
First-order variant of [Koymans 1990], [Alur/Henzinger 1990], ...
3. Formalization examples
4. Monitoring
5. Case study
6. Conclusion

Syntax

- ▶ A *signature* S is a tuple (C, R) .

C is a finite set of **constant symbols** and R is a finite set of **predicates**, each with an associated arity.

- ▶ (*MFOTL*) *formulas* over a signature S and set of variables V

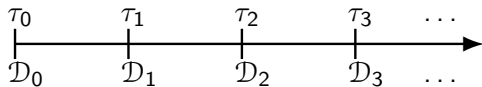
$$\phi ::= t_1 \approx t_2 \mid t_1 \prec t_2 \mid r(t_1, \dots, t_n) \mid \neg \phi \mid \phi \wedge \phi \mid \exists x. \phi \mid \\ \bullet_I \phi \mid \circ_I \phi \mid \phi \mathbf{S}_I \phi \mid \phi \mathbf{U}_I \phi,$$

where t_i range over $V \cup C$, r, x , range over R, V , and I is an *interval* of the form:

$$[b, b') := \{a \in \mathbb{N} \mid b \leq a < b'\}, \text{ for } b \in \mathbb{N}, b' \in \mathbb{N} \cup \{\infty\}, \text{ and } b < b'$$

- ▶ **Sugar** like $\blacksquare_I \phi := \neg(\text{true } \mathbf{S}_I \neg \phi)$ and $\square_I \phi := \neg(\text{true } \mathbf{U}_I \neg \phi)$.
Non-metric operators like $\square \phi := \square_{[0, \infty)} \phi$

Semantics (I)



- ▶ A *temporal (first-order) structure* (over S) is a pair $(\bar{\mathcal{D}}, \bar{\tau})$.
 - Sequence of *first-order structures* $\bar{\mathcal{D}} = (\mathcal{D}_0, \mathcal{D}_1, \dots)$.
Constant domains and *rigid interpretation* of constant symbols.
 - Sequence $\bar{\tau} = (\tau_0, \tau_1, \dots)$ of *time stamps*, $\tau_i \in \mathbb{N}$
Monotonically increasing and *progresses*.
- ▶ $(\bar{\mathcal{D}}, \bar{\tau}, v, i) \models \phi$ denotes *MFOTL satisfaction*
 $(\bar{\mathcal{D}}, \bar{\tau})$ is a temporal structure, v a valuation, $i \in \mathbb{N}$, and ϕ a formula.
- ▶ Standard semantics for first-order fragment.

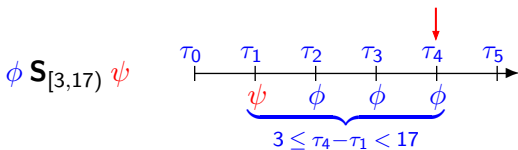
Semantics (II)

Metric temporal operators

A temporal formula is only satisfied at time point i if it is satisfied within the bounds given by interval I , relative to time stamp τ_i .

$(\bar{\mathcal{D}}, \bar{\tau}, v, i) \models \bigcirc_I \phi$	iff	$\tau_{i+1} - \tau_i \in I$ and $(\bar{\mathcal{D}}, \bar{\tau}, v, i+1) \models \phi$
$(\bar{\mathcal{D}}, \bar{\tau}, v, i) \models \bullet_I \phi$	iff	$i > 0$, $\tau_i - \tau_{i-1} \in I$, and $(\bar{\mathcal{D}}, \bar{\tau}, v, i-1) \models \phi$
$(\bar{\mathcal{D}}, \bar{\tau}, v, i) \models \phi \mathbf{U}_I \psi$	iff	for some $j \geq i$, $\tau_j - \tau_i \in I$, $(\bar{\mathcal{D}}, \bar{\tau}, v, j) \models \psi$, and $(\bar{\mathcal{D}}, \bar{\tau}, v, k) \models \phi$, for all $k \in [i, j)$
$(\bar{\mathcal{D}}, \bar{\tau}, v, i) \models \phi \mathbf{S}_I \psi$	iff	for some $j \leq i$, $\tau_i - \tau_j \in I$, $(\bar{\mathcal{D}}, \bar{\tau}, v, j) \models \psi$, and $(\bar{\mathcal{D}}, \bar{\tau}, v, k) \models \phi$, for all $k \in [j+1, i+1)$

Example



Road map

1. An example
2. Metric First-order Temporal Logic
3. **Formalization examples**
Three examples illustrating typical compliance policies and their formalization in MFOTL.
4. Monitoring
5. Case study
6. Conclusion

Transaction requirements (I)

Banking compliance à la **Bank Secrecy** or **USA Patriot Act**



- ▶ Requirements for monitoring, authorizing, and reporting **large** or **suspicious transactions**.
- ▶ Signature
 - Constant *th*: a threshold “large” amount.
 - *trans*(c, t, a): customer c carries out transaction t involving fund amount a .
 - *auth*(e, t): employee e authorizes t .
 - *report*(t): t is reported.
- ▶ In general, signature determined by monitoring requirements and events that system actually provides.

Transaction requirements (II)

- ▶ Transactions t of any customers c must be reported within 5 days when the transferred amount a exceeds a given threshold th :

$$\square \forall c. \forall t. \forall a. trans(c, t, a) \wedge th < a \rightarrow \diamond_{[0,6]} report(t)$$

- ▶ Transactions exceeding the threshold must be authorized by an employee (e.g., 2-20 days) before execution:

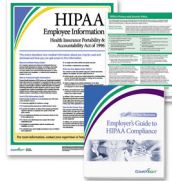
$$\square \forall c. \forall t. \forall a. trans(c, t, a) \wedge th < a \rightarrow \blacklozenge_{[2,21]} \exists e. auth(e, t)$$

- ▶ Each transaction t of a customer c , who has within the last 30 days been involved in a suspicious transaction t' , must be reported as suspicious within 2 days:

$$\square \forall t. \forall c. \forall a. trans(c, t, a) \wedge \\ (\blacklozenge_{[0,31]} \exists t'. \exists a'. trans(c, t', a') \wedge \diamond_{[0,6]} report(t')) \rightarrow \\ \diamond_{[0,3]} report(t)$$

Data retention requirements (I)

Health Insurance Portability and Accountability Act (HIPAA)



- ▶ Regulations address storage of health records.
 - Limited storage of sensitive records in the hospital's central database.
 - However, archiving is required for auditing and liability reasons.
- ▶ Signature
 - Constants *db* and *archive*: hospital's central and archive databases.
 - *hospitalize(p)* and *release(p)*: patient p is hospitalized and released.
 - *delete(d, p)*: patient p 's health record is deleted from the database d .
 - *copy(d, d', p)*: patient p 's health record is copied from database d to d' .

Data retention requirements (II)

- ▶ A patient's health record must be deleted from hospital's database within 14 days after the patient is released from the hospital, unless the patient is readmitted to the hospital within this time window:

$$\Box \forall p. \text{release}(p) \rightarrow \Diamond_{[0,15)} \text{delete}(db, p) \vee \text{hospitalize}(p).$$

- ▶ A health record is archived at most 7 days before it is deleted from the central database:

$$\Box \forall p. \text{delete}(db, p) \rightarrow \blacklozenge_{[0,8)} \text{copy}(db, \text{archive}, p)$$

- ▶ Archived data must be stored for at least 8 years:

$$\Box \forall p. \text{copy}(db, \text{archive}, p) \rightarrow \Box_{[0,9)} \neg \text{delete}(\text{archive}, p)$$

N.B. timestamps must distinguish time units, e.g., days versus years

Chinese Wall



- ▶ Policy to avoid conflict-of-interest situations

Subject s must not access object o when s has previously accessed another object in a different dataset than o and both datasets are in the same conflict-of-interest class

- ▶ A possible formalization (with timing constraints):

$$\square \forall s. \forall o. \forall d. \forall d'. \text{access}(s, o) \wedge \underline{\text{dataset}}(o, d) \wedge (\exists o'. (\blacklozenge_{[0,4]} \text{access}(s, o')) \wedge \underline{\text{dataset}}(o', d')) \rightarrow \neg \underline{\text{conflict}}(d, d')$$

Assume that:

- At each time point, conflict is irreflexive and symmetric
 - At each time point, dataset is a partial function from objects to datasets
- ▶ Different types of predicates:
 - Event predicate: **accessing** an object happens at a time point
 - State predicate: **being** in a dataset has a duration (start and finish)
 - Datasets and conflict-of-interest classes might change over time

Experience with formalization in practice

Limitations and problems

- ▶ Precision must precede formalization.
 - “... must be securely stored.”
- ▶ Not all requirements can be enforced by monitoring system traces.
 - “Information systems must be protected from intrusion.”
 - “A contingency plan should be in place for responding to emergencies.”
- ▶ Large gap between high-level policies and system information.
 - “Data should be use for statistical purposes only.”
 - “... must be deleted ...”

Overcoming these problems is nontrivial.

MFOTL is a good fit afterwards.

Road map

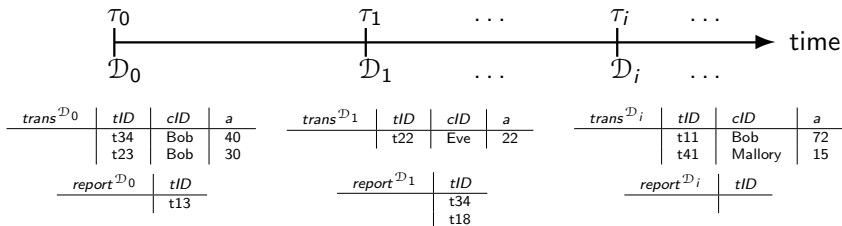
1. An example
2. Metric First-order Temporal Logic
3. Formalization examples
4. **Monitoring**
5. Case study
6. Conclusion

Monitoring objective

- ▶ Given a policy ϕ (example from transaction processing)

$$\square \forall t. \forall c. \forall a. \text{trans}(c, t, a) \wedge (\blacklozenge_{[0,31]} \exists t'. \exists a'. \text{trans}(c, t', a') \wedge \blacklozenge_{[0,6]} \text{report}(t')) \\ \rightarrow \blacklozenge_{[0,3]} \text{report}(t)$$

and a **timed temporal structure prefix** given by system events or logs

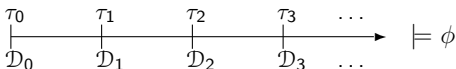


- ▶ monitor should **report all policy violations**

Ideas just sketched here. See papers for definitions and proofs.

Restrictions

Not all policies and log files can be effectively monitored



- ▶ MFOTL formula ϕ of form $\Box \phi'$, where ϕ' is bounded.
 - For all occurrences of operator $\mathbf{U}_{[c,d]}$ in ϕ' , $d \neq \infty$
 - So ϕ describes a safety property

- ▶ Structures $\bar{\mathcal{D}} = \mathcal{D}_0, \mathcal{D}_1, \dots$ Options:

1. Each structure \mathcal{D}_i is **automatic**

Roughly, each \mathcal{D}_i representable by a collection of finite automata.

See, e.g. [Khoussainov & Nerode 1995] and [Blumensath & Grädel 2004]

2. **or all relations** in \mathcal{D}_i are **finite** (Special case of 1.)

But then negation and quantification must be appropriately handled, e.g.,

$$r(x) \wedge \bigcirc \neg q(x) \quad \rightsquigarrow \quad r(x) \wedge \bigcirc (\neg q(x) \wedge \bullet r(x))$$

We have taken Option 2 in our implementation.

Preprocessing: negation and rewriting

► Input formula ϕ

$$\square \forall t. \forall c. \forall a. \text{trans}(c, t, a) \wedge (\blacklozenge_{[0,31]} \exists t'. \exists a'. \text{trans}(c, t', a') \wedge \blacklozenge_{[0,6]} \text{report}(t')) \\ \rightarrow \blacklozenge_{[0,3]} \text{report}(t)$$

Preprocessing: negation and rewriting

- ▶ Input formula ϕ

$$\square \forall t. \forall c. \forall a. \text{trans}(c, t, a) \wedge (\blacklozenge_{[0,31]} \exists t'. \exists a'. \text{trans}(c, t', a') \wedge \diamond_{[0,6]} \text{report}(t')) \\ \rightarrow \diamond_{[0,3]} \text{report}(t)$$

- ▶ Negate, rewrite, and drop outermost \diamond and \exists quantifiers, yielding ψ

$$\diamond \exists t. \exists c. \exists a. \text{trans}(c, t, a) \wedge (\blacklozenge_{[0,31]} \exists t'. \exists a'. \text{trans}(c, t', a') \wedge \diamond_{[0,6]} \text{report}(t')) \\ \wedge \square_{[0,3]} \neg \text{report}(t)$$

Preprocessing: negation and rewriting

- ▶ Input formula ϕ

$$\Box \forall t. \forall c. \forall a. \text{trans}(c, t, a) \wedge (\blacklozenge_{[0,31]} \exists t'. \exists a'. \text{trans}(c, t', a') \wedge \blacklozenge_{[0,6]} \text{report}(t')) \\ \rightarrow \blacklozenge_{[0,3]} \text{report}(t)$$

- ▶ Negate, rewrite, and drop outermost \blacklozenge and \exists quantifiers, yielding ψ

$$\blacklozenge \exists t. \exists c. \exists a. \text{trans}(c, t, a) \wedge (\blacklozenge_{[0,31]} \exists t'. \exists a'. \text{trans}(c, t', a') \wedge \blacklozenge_{[0,6]} \text{report}(t')) \\ \wedge \Box_{[0,3]} \neg \text{report}(t)$$

- ▶ To monitor: for each $i \in \mathbb{N}$, determine elements satisfying ψ :

$$\{\bar{a} \mid (\bar{\mathcal{D}}, \bar{\tau}, v[\bar{x}/\bar{a}], i) \models \psi\}$$

These are suspicious transactions that were not reported.

Preprocessing: reduction to first-order queries

- For each temporal subformula α in ψ , introduce an auxiliary predicate p_α

$$\text{trans}(c, t, a) \wedge \underbrace{\left(\underbrace{\diamond_{[0,31]} \exists t'. \exists a'. \text{trans}(c, t', a')}_{p_{\alpha_1}} \wedge \underbrace{\diamond_{[0,6]} \text{report}(t')}_{p_{\alpha_2}} \right)}_{p_{\alpha_3}} \wedge \underbrace{\square_{[0,3]} \neg \text{report}(t)}_{p_{\alpha_2}}$$

Preprocessing: reduction to first-order queries

- ▶ For each temporal subformula α in ψ , introduce an auxiliary predicate p_α

$$\text{trans}(c, t, a) \wedge \underbrace{\left(\underbrace{\diamond_{[0,31]} \exists t'. \exists a'. \text{trans}(c, t', a')}_{p_{\alpha_1}} \wedge \underbrace{\diamond_{[0,6]} \text{report}(t')}_{p_{\alpha_2}} \right)}_{p_{\alpha_3}} \wedge \underbrace{\square_{[0,3]} \neg \text{report}(t)}_{p_{\alpha_2}}$$

- ▶ Replace each α by a corresponding p_α , yielding first-order formula $\hat{\psi}$

$$\text{trans}(c, t, a) \wedge p_{\alpha_3}(c) \wedge p_{\alpha_2}(t)$$

Preprocessing: reduction to first-order queries

- ▶ For each temporal subformula α in ψ , introduce an auxiliary predicate p_α

$$trans(c, t, a) \wedge \underbrace{\left(\underbrace{\left(\underbrace{\diamond_{[0,31]} \exists t'. \exists a'. trans(c, t', a')}_{p_{\alpha_1}} \wedge \underbrace{\diamond_{[0,6]} report(t')}_{p_{\alpha_1}} \right)}_{p_{\alpha_3}} \wedge \underbrace{\square_{[0,3]} \neg report(t)}_{p_{\alpha_2}} \right)}_{p_{\alpha_3}}$$

- ▶ Replace each α by a corresponding p_α , yielding first-order formula $\hat{\psi}$

$$trans(c, t, a) \wedge p_{\alpha_3}(c) \wedge p_{\alpha_2}(t)$$

- ▶ **Monitoring:** for each $i \in \mathbb{N}$
 - Extend \mathcal{D}_i to $\hat{\mathcal{D}}_i$, where for each temporal subformula α

$$p_{\alpha}^{\hat{\mathcal{D}}_i} = \{ \bar{a} \mid (\bar{\mathcal{D}}, \bar{\tau}, v[\bar{x}/\bar{a}], i) \models \alpha \}$$

- Query extended first-order structure $\hat{\mathcal{D}}_i$

$$\{ \bar{a} \mid (\hat{\mathcal{D}}_i, v[\bar{x}/\bar{a}]) \models \hat{\psi} \}$$

Preprocessing: reduction to first-order queries

- ▶ For each temporal subformula α in ψ , introduce an auxiliary predicate p_α

$$trans(c, t, a) \wedge \underbrace{\left(\underbrace{\left(\underbrace{\diamond_{[0,31]} \exists t'. \exists a'. trans(c, t', a')}_{p_{\alpha_1}} \wedge \underbrace{\diamond_{[0,6]} report(t')}_{p_{\alpha_1}} \right)}_{p_{\alpha_3}} \wedge \underbrace{\square_{[0,3]} \neg report(t)}_{p_{\alpha_2}} \right)}_{p_{\alpha_3}}$$

- ▶ Replace each α by a corresponding p_α , yielding first-order formula $\hat{\psi}$

$$trans(c, t, a) \wedge p_{\alpha_3}(c) \wedge p_{\alpha_2}(t)$$

- ▶ **Monitoring:** for each $i \in \mathbb{N}$
 - Extend \mathcal{D}_i to $\hat{\mathcal{D}}_i$, where for each temporal subformula α

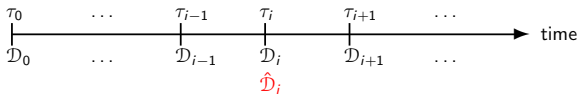
$$p_{\alpha}^{\hat{\mathcal{D}}_i} = \{ \bar{a} \mid (\bar{\mathcal{D}}, \bar{\tau}, v[\bar{x}/\bar{a}], i) \models \alpha \}$$

- Query extended first-order structure $\hat{\mathcal{D}}_i$

$$\{ \bar{a} \mid (\hat{\mathcal{D}}_i, v[\bar{x}/\bar{a}]) \models \hat{\psi} \}$$

Next: how to **incrementally** build the auxiliary relations $p_{\alpha}^{\hat{\mathcal{D}}_i}$ for each $\hat{\mathcal{D}}_i$

Building the auxiliary relations



- ▶ Build auxiliary relations $p_\alpha^{\hat{\mathcal{D}}_i}$ in $\hat{\mathcal{D}}_i$ inductively over α 's formula structure and using relations from both **previous** and **subsequent** structures.

- ▶ Example for

$$\alpha := \bullet_I \beta$$

$$p_\alpha^{\hat{\mathcal{D}}_i} := \begin{cases} \hat{\beta}^{\hat{\mathcal{D}}_{i-1}} & \text{if } i > 0 \text{ and } \tau_i - \tau_{i-1} \in I \\ \emptyset & \text{otherwise} \end{cases}$$

- ▶ Example for

$$\alpha := \circ_I \beta$$

$$p_\alpha^{\hat{\mathcal{D}}_i} := \begin{cases} \hat{\beta}^{\hat{\mathcal{D}}_{i+1}} & \text{if } \tau_{i+1} - \tau_i \in I \\ \emptyset & \text{otherwise} \end{cases}$$

Depends on the relations in \mathcal{D}_{i+1} and auxiliary relations in $\hat{\mathcal{D}}_{i+1}$.

Hence monitor instantiates $p_\alpha^{\hat{\mathcal{D}}_i}$ with a delay of at least one time step.

- ▶ Idea similar for other temporal operators.

Additional complexity: ensuring an **incremental** construction.

Monitor $\mathcal{M}(\psi)$

```
1:  $i \leftarrow 0$  % lookahead index in sequence  $(\mathcal{D}_0, \tau_0), (\mathcal{D}_1, \tau_1), \dots$ 
2:  $q \leftarrow 0$  % index of next query evaluation in sequence  $(\mathcal{D}_0, \tau_0), (\mathcal{D}_1, \tau_1), \dots$ 
3:  $Q \leftarrow \{(\alpha, 0, \text{waitfor}(\alpha)) \mid \alpha \text{ temporal subformula of } \psi\}$ 
4: loop
5: Carry over constants and relations of  $\mathcal{D}_i$  to  $\hat{\mathcal{D}}_i$ .
6: for all  $(\alpha, j, \emptyset) \in Q$  do % can build relation for  $\alpha$  in  $\hat{\mathcal{D}}_j$ 
7:   Build auxiliary relation for  $\alpha$  in  $\hat{\mathcal{D}}_j$ .
8:   Discard auxiliary relation for  $\alpha$  in  $\hat{\mathcal{D}}_{j-1}$  if  $j - 1 \geq 0$ .
9:   Discard relations  $p_\delta^{\hat{\mathcal{D}}_j}$ , where  $\delta$  is a temporal subformula of  $\alpha$ .
10: while all relations  $p_\alpha^{\hat{\mathcal{D}}_q}$  are built for  $\alpha \in \text{tsub}(\psi)$  do
11:   Output violations  $\hat{\psi}^{\hat{\mathcal{D}}_q}$  and time stamp  $\tau_q$ .
12:   Discard structure  $\hat{\mathcal{D}}_{q-1}$  if  $q > 0$ .
13:    $q \leftarrow q + 1$ 
14:  $Q \leftarrow \{(\alpha, i + 1, \text{waitfor}(\alpha)) \mid \alpha \text{ temporal subformula of } \psi\} \cup$   

    $\{(\alpha, j, \bigcup_{\alpha' \in \text{update}(S, \tau_{i+1} - \tau_i)} \text{waitfor}(\alpha')) \mid (\alpha, j, S) \in Q \text{ and } S \neq \emptyset\}$ 
15:  $i \leftarrow i + 1$  % process next element in input sequence  $(\mathcal{D}_{i+1}, \tau_{i+1})$ 
16: end loop
```

Counters q (query) and i (lookahead) into input sequence

Monitor $\mathcal{M}(\psi)$

```

1:  $i \leftarrow 0$                                 % lookahead index in sequence  $(\mathcal{D}_0, \tau_0), (\mathcal{D}_1, \tau_1), \dots$ 
2:  $q \leftarrow 0$                             % index of next query evaluation in sequence  $(\mathcal{D}_0, \tau_0), (\mathcal{D}_1, \tau_1), \dots$ 
3:  $Q \leftarrow \{(\alpha, 0, \text{waitfor}(\alpha)) \mid \alpha \text{ temporal subformula of } \psi\}$ 
4: loop
5:   Carry over constants and relations of  $\mathcal{D}_i$  to  $\hat{\mathcal{D}}_i$ .
6:   for all  $(\alpha, j, \emptyset) \in Q$  do                                % can build relation for  $\alpha$  in  $\hat{\mathcal{D}}_j$ 
7:     Build auxiliary relation for  $\alpha$  in  $\hat{\mathcal{D}}_j$ .
8:     Discard auxiliary relation for  $\alpha$  in  $\hat{\mathcal{D}}_{j-1}$  if  $j - 1 \geq 0$ .
9:     Discard relations  $p_\delta^{\hat{\mathcal{D}}_j}$ , where  $\delta$  is a temporal subformula of  $\alpha$ .
10:  while all relations  $p_\alpha^{\hat{\mathcal{D}}_q}$  are built for  $\alpha \in \text{tsub}(\psi)$  do
11:    Output violations  $\hat{\psi}^{\hat{\mathcal{D}}_q}$  and time stamp  $\tau_q$ .
12:    Discard structure  $\hat{\mathcal{D}}_{q-1}$  if  $q > 0$ .
13:     $q \leftarrow q + 1$ 
14:   $Q \leftarrow \{(\alpha, i + 1, \text{waitfor}(\alpha)) \mid \alpha \text{ temporal subformula of } \psi\} \cup$ 
       $\{(\alpha, j, \bigcup_{\alpha' \in \text{update}(S, \tau_{i+1} - \tau_i)} \text{waitfor}(\alpha')) \mid (\alpha, j, S) \in Q \text{ and } S \neq \emptyset\}$ 
15:   $i \leftarrow i + 1$                                 % process next element in input sequence  $(\mathcal{D}_{i+1}, \tau_{i+1})$ 
16: end loop

```

Q maintains list of unevaluated subformula (α, j, S) for past time points

Monitor $\mathcal{M}(\psi)$

```

1:  $i \leftarrow 0$                                 % lookahead index in sequence  $(\mathcal{D}_0, \tau_0), (\mathcal{D}_1, \tau_1), \dots$ 
2:  $q \leftarrow 0$                                 % index of next query evaluation in sequence  $(\mathcal{D}_0, \tau_0), (\mathcal{D}_1, \tau_1), \dots$ 
3:  $Q \leftarrow \{(\alpha, 0, \text{waitfor}(\alpha)) \mid \alpha \text{ temporal subformula of } \psi\}$ 
4: loop
5:   Carry over constants and relations of  $\mathcal{D}_i$  to  $\hat{\mathcal{D}}_i$ .
6:   for all  $(\alpha, j, \emptyset) \in Q$  do                                % can build relation for  $\alpha$  in  $\hat{\mathcal{D}}_j$ 
7:     Build auxiliary relation for  $\alpha$  in  $\hat{\mathcal{D}}_j$ .
8:     Discard auxiliary relation for  $\alpha$  in  $\hat{\mathcal{D}}_{j-1}$  if  $j - 1 \geq 0$ .
9:     Discard relations  $p_\delta^{\hat{\mathcal{D}}_j}$ , where  $\delta$  is a temporal subformula of  $\alpha$ .
10:  while all relations  $p_\alpha^{\hat{\mathcal{D}}_q}$  are built for  $\alpha \in \text{tsub}(\psi)$  do
11:    Output violations  $\hat{\psi}^{\hat{\mathcal{D}}_q}$  and time stamp  $\tau_q$ .
12:    Discard structure  $\hat{\mathcal{D}}_{q-1}$  if  $q > 0$ .
13:     $q \leftarrow q + 1$ 
14:   $Q \leftarrow \{(\alpha, i + 1, \text{waitfor}(\alpha)) \mid \alpha \text{ temporal subformula of } \psi\} \cup$ 
       $\{(\alpha, j, \bigcup_{\alpha' \in \text{update}(S, \tau_{i+1} - \tau_i)} \text{waitfor}(\alpha')) \mid (\alpha, j, S) \in Q \text{ and } S \neq \emptyset\}$ 
15:   $i \leftarrow i + 1$                                 % process next element in input sequence  $(\mathcal{D}_{i+1}, \tau_{i+1})$ 
16: end loop

```

Given relations for all temporal subformulas, output policy violations

Analysis of space consumption of $\mathcal{M}(\psi)$

► Assumptions

- Relations are finite and ψ is monitorable
- Number of equal time stamps is bounded

- **Theorem:** At each time point, space $\mathcal{M}(\psi)$ needs to store auxiliary relations is **polynomially bounded** by cardinality of the active domain.

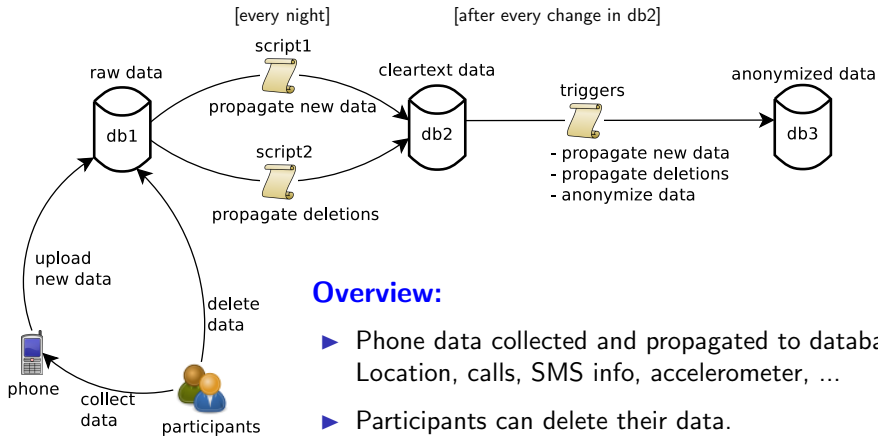
N.B. the **active domain** be the set of data elements occurring in the relations in a prefix of a timed temporal structure.

- On simulations and case studies, space requirements often modest. Only a **relevant** part of history is required (and must be saved) at any time, with an associated, smaller **relevant active domain**.

Road map

1. An example
2. Metric First-order Temporal Logic
3. Formalization examples
4. Monitoring
5. **Case study**
6. Conclusion

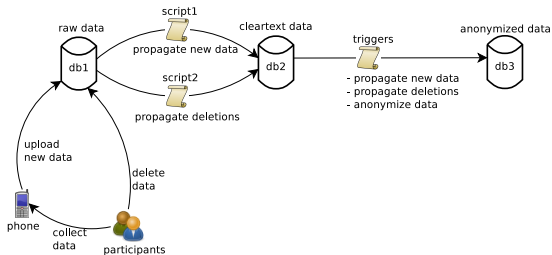
Nokia's Data Collection Campaign



Overview:

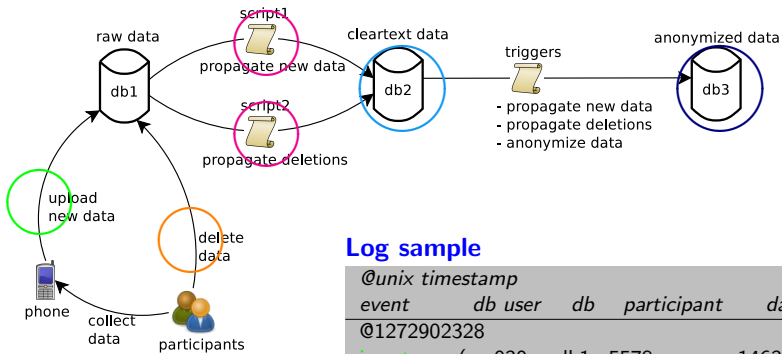
- ▶ Phone data collected and propagated to databases. Location, calls, SMS info, accelerometer, ...
- ▶ Participants can delete their data.
- ▶ Cleartext data used for personalized applications, e.g., location-history maps
- ▶ Anonymized data is used for research.

Policies (sample)



1. Access control rules
restrict who accesses and modifies data in databases, e.g.,
 - Only user *script2* may delete data from *db2*. (A)
2. Data changes are propagated between databases, e.g.,
 - Data deleted from *db2* is deleted from *db3* within 60 seconds. (B)
 - New data inserted into *db1* is, within 30 hours, either inserted into *db2* or deleted from *db1*. (C)
 - Data deleted from *db1* is deleted from *db2* within 30 hours, provided it has already been inserted into *db2*. (D)
3. Database is accessed by a script account only while script is running.
4. Scripts run for at most 8 hours.
5. Scripts are the latest SVN version when executed.

Logging

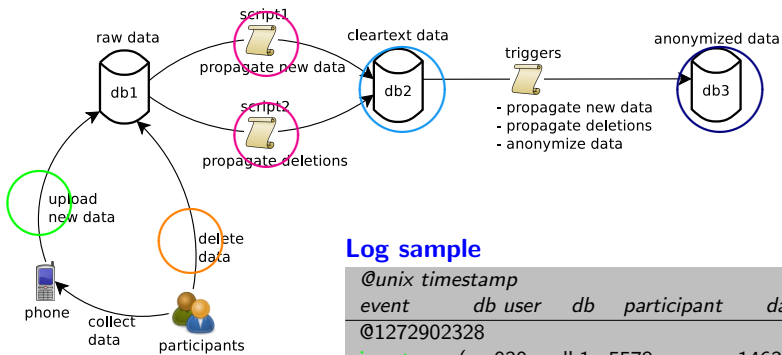


Log sample

@unix timestamp

event	db user	db	participant	data id
@1272902328				
insert	(eu.030,	db1,	5578,	146368038)
delete	(eu.031,	db1,	5480,	122368122)
@1272902355				
delete	(script2,	db2,	5580,	108031209)
select	(res.012,	db3,	[unknown],	146368038)
@1273158243				
script_end	(script1)			

Logging



Log sample

@unix timestamp

event	db user	db	participant	data id
@1272902328				
insert	(eu.030,	db1,	5578,	146368038)
delete	(eu.031,	db1,	5480,	122368122)
@1272902355				
delete	(script2,	db2,	5580,	108031209)
select	(res.012,	db3,	[unknown],	146368038)
@1273158243				
script_end	(script1)			

- ▶ need to merge logs produced by different parts of the system

Results: resource usage

- ▶ **Example:** 24-hour log with 1.5 million entries, 180 participants

policy	analyzed policy formula	running time	max storage
(A)	$\text{delete}(user, db2, x, y) \rightarrow user = \text{script2}$	10 s	10.5 MB
(B)	$(\text{delete}(x, db2, y, data) \wedge data \neq \text{unknown}) \rightarrow$ $\Diamond_{[0,60s]} \exists u, v. \text{delete}(u, db3, v, data)$	60 s	13.8 MB
(C)	$(\text{insert}(x, db1, y, data) \wedge data \neq \text{unknown}) \rightarrow$ $\Diamond_{[0,30h]} \exists u, v. (\text{insert}(u, db2, v, data) \vee \text{delete}(u, db1, v, data))$	3 h 5 min	564 MB
(D)	$(\text{delete}(x, db1, y, data) \wedge data \neq \text{unknown}) \rightarrow$ $(((((\neg \exists u, v. \text{insert}(u, db2, v, data)) \mathbf{S}_{[0,30h]} \exists u, v. \text{insert}(u, db1, v, data)))$ $\wedge \square_{[0,30h]} \neg \exists u, v. \text{insert}(u, db2, v, data))$ $\vee \Diamond_{[0,30h]} \exists u, v. \text{delete}(u, db2, v, data))$	1 h 37 min	610 MB

- ▶ **Processing times reasonable:** could process (on ordinary PC) log entries faster than they were produced.
- ▶ **Space requirements manageable.**
 - However, this is a potential bottleneck for larger case studies.
 - Raises research questions on how to best scale approach.

Results: compliance checking

- ▶ Monitor uncovered a number of (potential) violations.
- ▶ **Attempted violations**: system users carry out unauthorized actions.
 - **Reason**: we initially monitored just attempted actions, not their effects.
- ▶ **Actual violations**
 - Revealed some bugs, e.g., in scripts.
 - Due to testing, debugging, and other improvement activities, which were taking place while data was being collected.
 - **Reason**: system is an experimental system under development, rather than a production system
- ▶ Case study shows the value of policy monitoring.
 - Useful even in a benevolent environment where the enterprise is committed to policy compliance.
 - Helpful to debug and sharpen controls.
 - Can be used to support audits, both internal and external.

Road map

1. An example
2. Metric First-order Temporal Logic
3. Formalization examples
4. Monitoring
5. Case study
6. **Conclusion**

Conclusion

- ▶ Policy monitoring is an exciting, challenging, and relevant topic!
- ▶ MFOTL is well-suited for formalizing and monitoring policies.
Very expressive, yet practically feasible.
- ▶ No silver bullet
 - Not every policy can be formalized in MFOTL.
 - Space consumption is still an issue.

Conclusion

- ▶ Policy monitoring is an exciting, challenging, and relevant topic!
- ▶ MFOTL is well-suited for formalizing and monitoring policies.
Very expressive, yet practically feasible.
- ▶ No silver bullet
 - Not every policy can be formalized in MFOTL.
 - Space consumption is still an issue.
- ▶ Current and future work
 - Larger case studies
 - Implementation using automatic structures
 - Enforcement rather than audit (in general, undecidable)

Bibliography

► **Monitoring foundations**

- D.B., Felix Klaedtke, Samuel Müller: Policy Monitoring in First-order Temporal Logic, CAV 2010.
- D.B., Felix Klaedtke, Samuel Müller, Birgit Pfitzmann: Runtime Monitoring of Metric First-order Temporal Properties. FSTTCS 2008.

► **Applications and enforcement**

- D.B., Felix Klaedtke, Samuel Müller: Monitoring security policies with metric first-order temporal logic. SACMAT 2010.
- Alex Pretschner, Manuel Hilty, D.B., Christian Schaefer, Thomas Walter: Mechanisms for usage control. ASIACCS 2008.
- Alex Pretschner, Manuel Hilty, D.B., Distributed usage control. Commun. ACM 49(9), 2006.
- Manuel Hilty, D.B., Alex Pretschner: On Obligations. ESORICS 2005.