Technische Universität München

Distributed Data Usage Control

Alexander Pretschner
TU München

FOSAD Summer School, September 2014

jww E. Lovat, F. Kelbert, P. Kumari, M. Büchler, P. Birnstill, C. Bier,
T. Wüchner, D. Holling, M. Lörscher, J. Peschla, P. Wenz

Problem

Technische Universität München

Don‘t copy my data

Delete my data
if I say so

A. Pretschner: Usage Control. Bertinoro 2014 2

Step 1: Object Clarifications

Technische Universität München

Copy data

Delete data

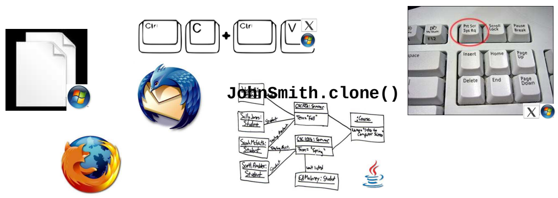
A. Pretschner: Usage Control. Bertinoro 2014 3

Step 2: Action Clarifications

Technische Universität München

Copy data

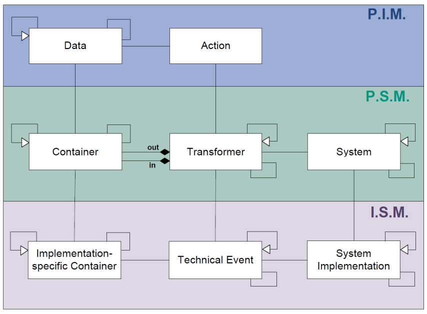
Delete data



A. Pretschner: Usage Control. Bertinoro 2014 4

Technische Universität München

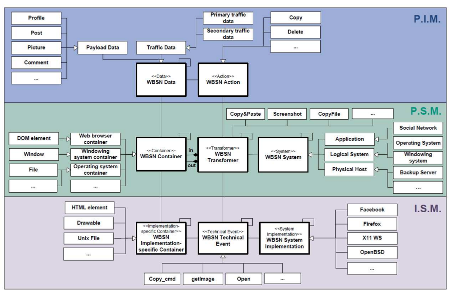
Domain Meta Model



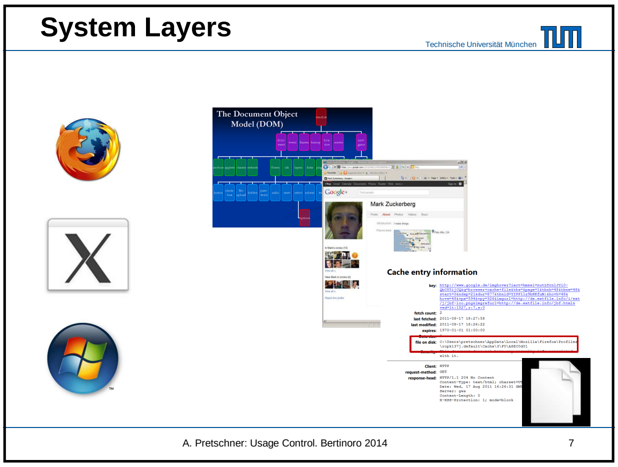
A. Pretschner: Usage Control. Bertinoro 2014 5

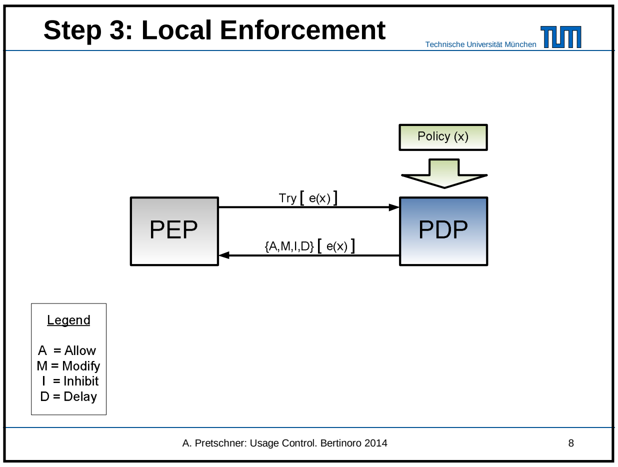
Technische Universität München

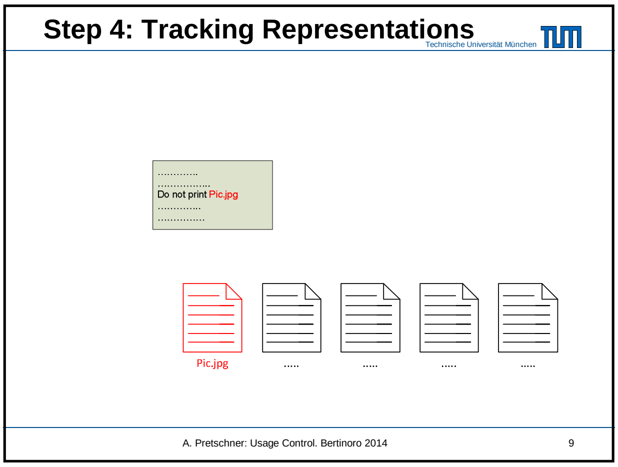
WBSN Domain Model

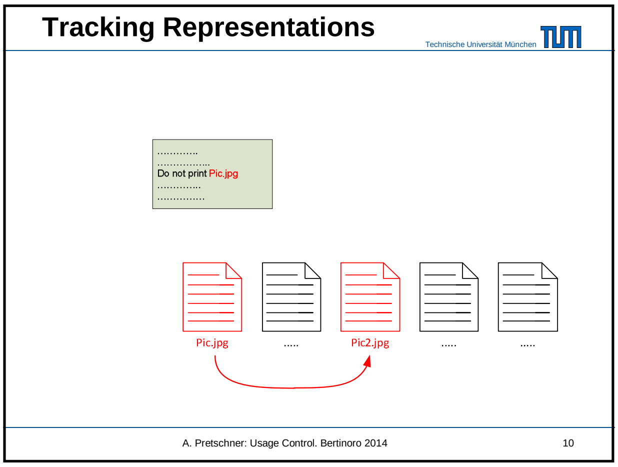


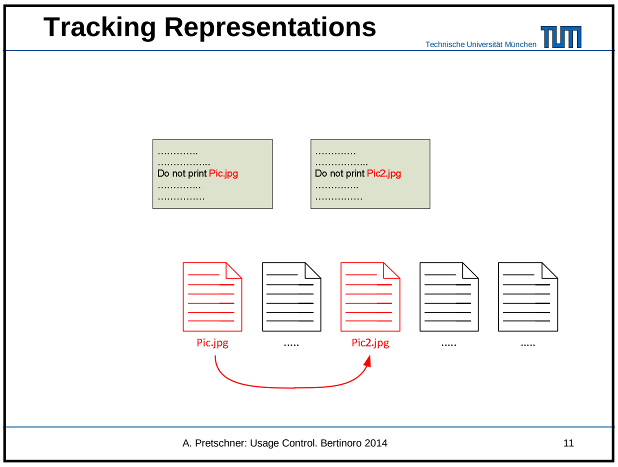
A. Pretschner: Usage Control. Bertinoro 2014 6

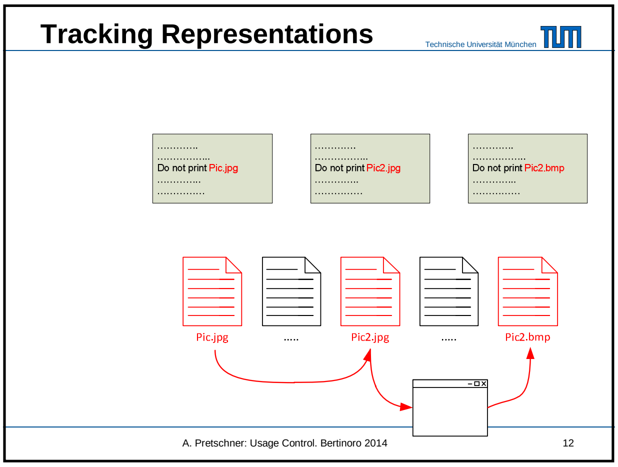

System Layers
Technische Universität München
The Document Object Model (DOM)
Mark Zuckerberg
Cache entry information
A. Pretschner: Usage Control. Bertinoro 2014
7


Step 3: Local Enforcement
Technische Universität München
Policy (x)
Try [e(x)]
PEP
PDP
{A,M,I,D} [e(x)]
Legend
A = Allow
M = Modify
I = Inhibit
D = Delay
A. Pretschner: Usage Control. Bertinoro 2014
8

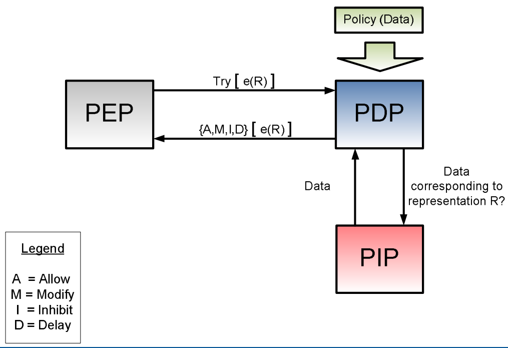

Step 4: Tracking Representations
Technische Universität München
Do not print Pic.jpg
Pic.jpg
.....
.....
.....
.....
A. Pretschner: Usage Control. Bertinoro 2014
9


Tracking Representations
Technische Universität München
Do not print Pic.jpg
Pic.jpg
.....
Pic2.jpg
.....
.....
A. Pretschner: Usage Control. Bertinoro 2014
10

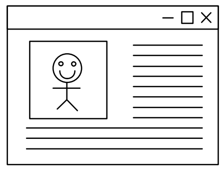

Tracking Representations
Technische Universität München
Do not print Pic.jpg
Do not print Pic2.jpg
Pic.jpg
.....
Pic2.jpg
.....
.....
A. Pretschner: Usage Control. Bertinoro 2014
11


Tracking Representations
Technische Universität München
Do not print Pic.jpg
Do not print Pic2.jpg
Do not print Pic2.bmp
Pic.jpg
.....
Pic2.jpg
.....
Pic2.bmp
A. Pretschner: Usage Control. Bertinoro 2014
12

Implementation: PEP+PDP+PIP

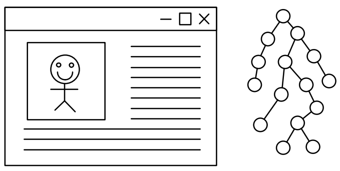


Step 5: Local Multi-Layer Enforcement



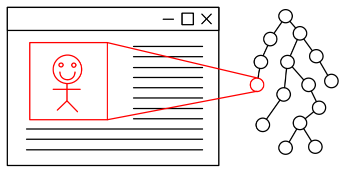
Local Multi-Layer Enforcement

- Web Browser



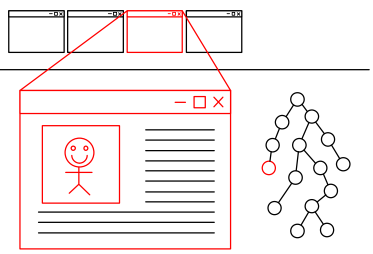
Local Multi-Layer Enforcement

- Web Browser



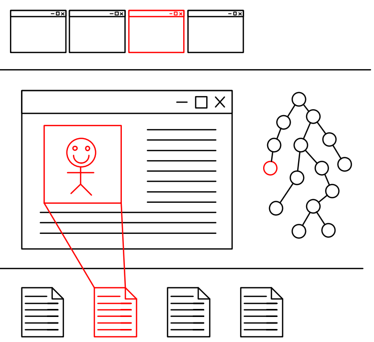
Local Multi-Layer Enforcement

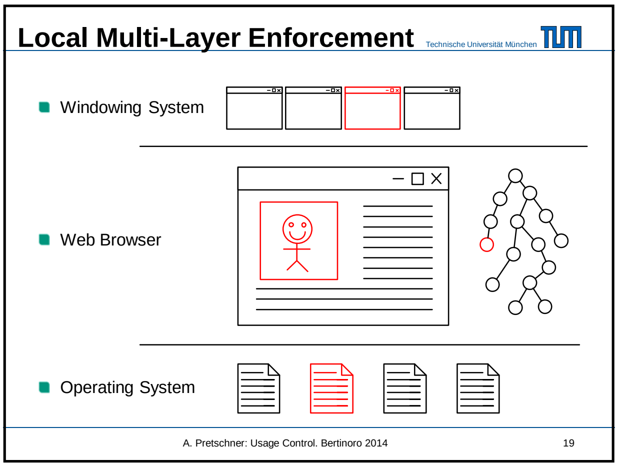
- Windowing System
- Web Browser

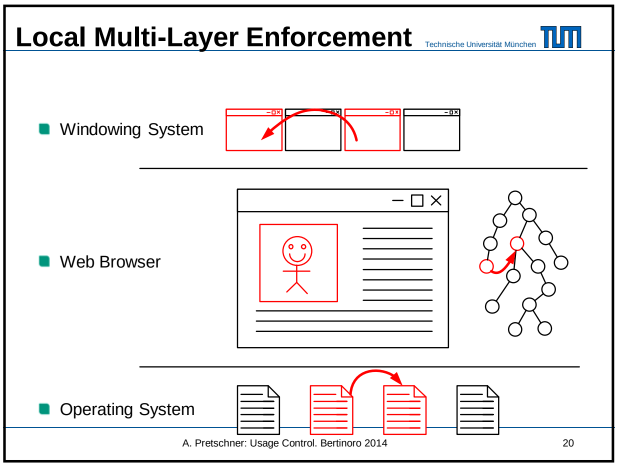


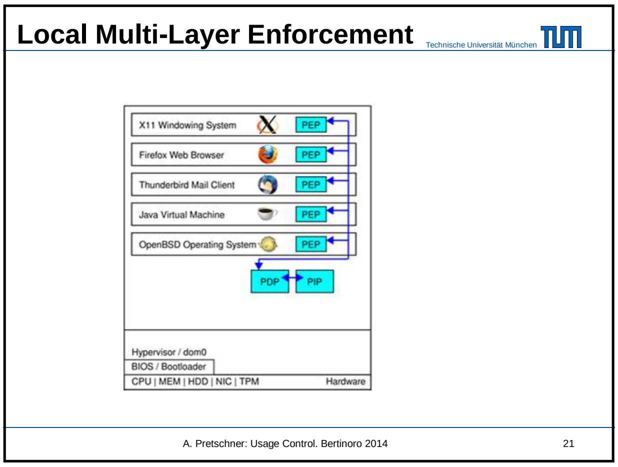
Local Multi-Layer Enforcement

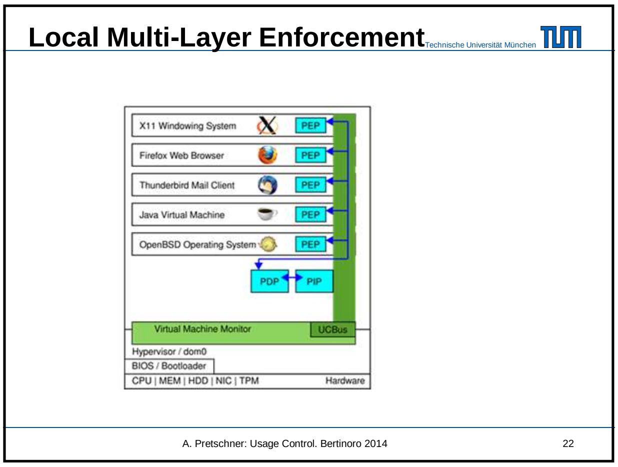
- Windowing System
- Web Browser
- Operating System

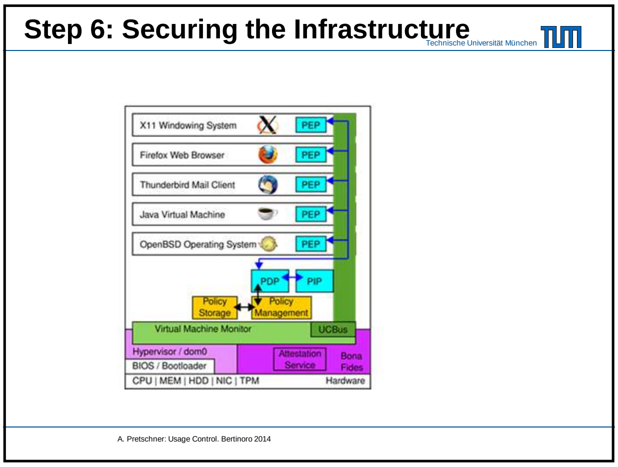


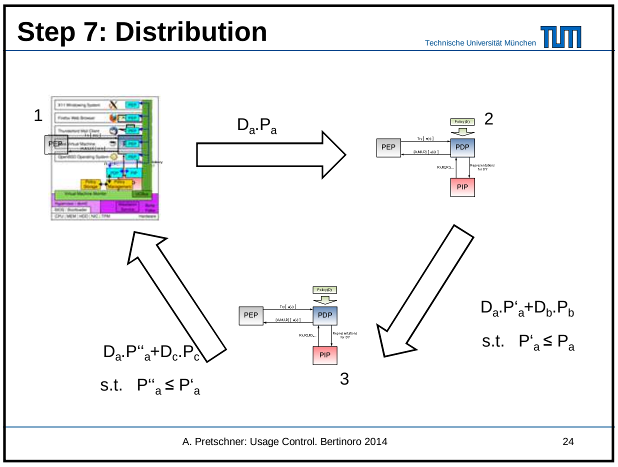

Local Multi-Layer Enforcement
Technische Universität München
Windowing System
Web Browser
Operating System
A. Pretschner: Usage Control. Bertinoro 2014
19


Local Multi-Layer Enforcement
Technische Universität München
Windowing System
Web Browser
Operating System
A. Pretschner: Usage Control. Bertinoro 2014
20


Local Multi-Layer Enforcement
Technische Universität München
X11 Windowing System
PEP
Firefox Web Browser
PEP
Thunderbird Mail Client
PEP
Java Virtual Machine
PEP
OpenBSD Operating System
PEP
PDP
PIP
Hypervisor / dom0
BIOS / Bootloader
CPU | MEM | HDD | NIC | TPM
Hardware
A. Pretschner: Usage Control. Bertinoro 2014
21

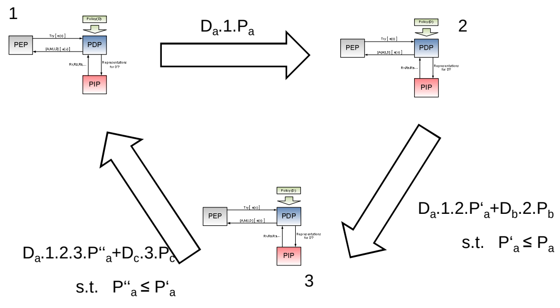

Local Multi-Layer Enforcement
Technische Universität München
X11 Windowing System
PEP
Firefox Web Browser
PEP
Thunderbird Mail Client
PEP
Java Virtual Machine
PEP
OpenBSD Operating System
PEP
PDP
PIP
Virtual Machine Monitor
UCBus
Hypervisor / dom0
BIOS / Bootloader
CPU | MEM | HDD | NIC | TPM
Hardware
A. Pretschner: Usage Control. Bertinoro 2014
22


Step 6: Securing the Infrastructure
Technische Universität München
X11 Windowing System
PEP
Firefox Web Browser
PEP
Thunderbird Mail Client
PEP
Java Virtual Machine
PEP
OpenBSD Operating System
PEP
PDP
PIP
Policy Storage
Policy Management
Virtual Machine Monitor
UCBus
Hypervisor / dom0
Attestation Service
Bona Fides
BIOS / Bootloader
CPU | MEM | HDD | NIC | TPM
Hardware
A. Pretschner: Usage Control. Bertinoro 2014


Step 7: Distribution
Technische Universität München
1
$D_a.P_a$
2
PEP
PDP
PIP
$D_a.P'_a+D_b.P_b$
s.t. $P'_a \le P_a$
3
$D_a.P''_a+D_c.P_c$
s.t. $P''_a \le P'_a$
A. Pretschner: Usage Control. Bertinoro 2014
24

Step 8: Provenance Tracking

Technische Universität München



A. Pretschner: Usage Control. Bertinoro 2014 25

Usage Control

Technische Universität München

- Access control defines and enforces access conditions
- Usage control additionally defines and enforces what must and what must not happen after distribution of data
  - Rights and Duties
  - Modification and execution possible decisions

Personal Data
- Master and billing, connection and sensor data
- Loyalty cards
- Patient and tax records
- SN profiles
- Photos and videos

Intellectual Property
- Trade secrets (e.g., the cloud)
- Copyrighted content (e.g., multimedia)

Business Data
- All kinds of data relevant to auditors
- Financial statements
- Access logs

Administrative Secrets
- Military data
- Intelligence data
- Official secrets

A. Pretschner: Usage Control. Bertinoro 2014 26

Distributed Usage Control

Technische Universität München

Teaser Demo: Data-Driven Cross-Layer Usage Control

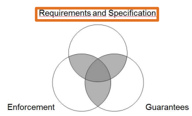
Video 1: http://www22.in.tum.de/fileadmin/demos/uc/uc_new_mp4.mp4

A. Pretschner: Usage Control. Bertinoro 2014 27

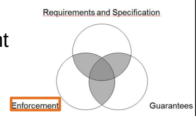
Technische Universität München

Agenda

- Part I: Introduction
- Part II: Event-Based Usage Control
- Part III: Data-Centric Usage Control
- Part IV: Quantitative Usage Control



- Part V: Local Single-Layer Enforcement
- Part VI: Distributed Enforcement
- Part VII: Cross-Layer Enforcement
- Part VIII: Policy Derivation



A. Pretschner: Usage Control. Bertinoro 2014 28

Technische Universität München

Agenda

- Part I: Introduction
- **Part II: Event-Based Usage Control**
- Part III: Data-Centric Usage Control
- Part IV: Quantitative Usage Control
- Part V: Local Single-Layer Enforcement
- Part VI: Distributed Enforcement
- Part VII: Cross-Layer Enforcement
- Part VIII: Policy Derivation
- Part IX: Discussion

A. Pretschner: Usage Control. Bertinoro 2014 29

Technische Universität München

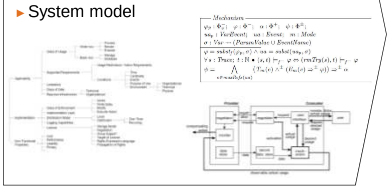
Event-Based Usage Control

- **II.1 Requirements**
- II.2 Events, refinements, traces
- II.3 Specification-level policies
- II.4 Implementation-level policies
- II.5 Video examples: context-aware UC on Android; UC for camera surveillance
- II.6 Analysis problems

A. Pretschner: Usage Control. Bertinoro 2014 30

Requirements

- Interviews
- Regulations
- Literature
- Taxonomy of enforcement mechanisms
- System model



- Restrictions and necessary actions
  - Permission and duty
- Conditions
  Time, cardinality, events, purpose, environment
- Examples
  - No distribution
  - Deletion after 30 days
  - At most one copy
  - Notification upon access
  - For statistical purposes only
  - Firewalls certified w.r.t. Common Criteria

Event-Based Usage Control

- II.1 Requirements
- **II.2 Events, refinements, traces**
- II.3 Specification-level policies
- II.4 Implementation-level policies
- II.5 Video examples
- II.6 Analysis problems

Policies

- Specification-level policies specify the what
  - Expressed as first order future time temporal logic formulas
- Implementation-level policies specify the how
  - Expressed as Event-Condition-Action rules:
    - Conditions first order past time temporal logic formulas
    - Actions describe inhibition, modification, execution

Background

- Events $\mathcal{E} \subseteq EName \times \mathbb{P}(PName \times PValue)$
- Variable events
  $Var = VName \to VVal \qquad VVal = \mathbb{P}(PValue \cup EName)$
  $\mathcal{VE} \subseteq (EName \cup VName) \times \mathbb{P}(PName \times (PValue \cup VName))$
- Refinement
  $\forall e_1, e_2 \in \mathcal{E} : e_1\ refinesEv\ e_2 \Leftrightarrow e_1.n = e_2.n \wedge e_1.p \supseteq e_2.p$
  $maxRefEv = \mathcal{E} \setminus \{e \in \mathcal{E} \mid \exists e' \in \mathcal{E} :\ e' \neq e \wedge e'\ refinesEv\ e\}$
- System events $\mathcal{S} \subseteq maxRefEv \times \{intended, actual\}$
- Traces $Trace : \mathbb{N} \to \mathbb{P}(\mathcal{S})$

Policies: first order part

$$\Gamma = \ \mathcal{VE} \mid \mathbb{N} \mid String \mid \Gamma\ op\ \Gamma \mid \ldots$$
$$\Psi = \ (\Psi) \mid \underline{false} \mid \Psi\ \underline{implies}\ \Psi \mid E(\mathcal{VE}) \mid I(\mathcal{VE})$$
$$\mid eval(\Gamma)$$
$$\mid \underline{forall}\ VName\ \underline{in}\ VVal : \Psi$$

Plus shortcuts

$\underline{true}$ for $\underline{false}\ \underline{implies}\ \underline{false}$, $\underline{not}(\psi)$ for $\psi\ \underline{implies}\ \underline{false}$
$\psi_1\ \underline{or}\ \psi_2$ for $(\underline{not}\ \psi_1)\ \underline{implies}\ \psi_2$,
$\psi_1\ \underline{and}\ \psi_2$ for $\underline{not}(\psi_1\ \underline{implies}\ \underline{not}\ \psi_2)$
$\underline{exists}\ vn\ \underline{in}\ VS : \psi$ for $\underline{not}(\underline{forall}\ vn \in VS : \underline{not}\psi)$

Will use $\to, \neg, \wedge, \vee, \forall, \exists, \in$

Macros for permissions omitted here

Examples: quantification, variables

$$\forall vn_1 \in SNGNAME : \forall vn_2 \in TIME :$$
$$E(play \mapsto \{obj \mapsto vn_1, time \mapsto vn_2\}) \ \to$$
$$E(log \mapsto \{obj \mapsto vn_1, time \mapsto vn_2\})$$

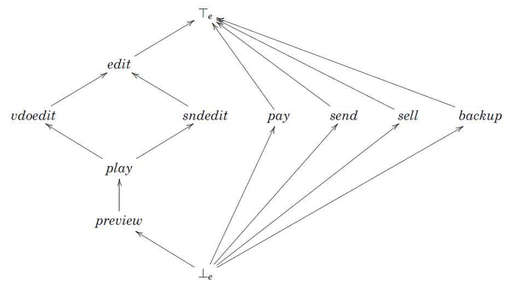
$$\forall vn_1 \in \{0, \ldots, 100\} : \forall vn_2 \in \{EU, US, JP, AU\} :$$
$$E(play \mapsto \{obj \mapsto sng.mp3, quality \mapsto vn_1, location \mapsto vn_2\})$$
$$\to (eval(vn_2 \neq EU) \ \to \ eval(vn_1 \leq 50))$$

$[\![eval(\gamma)]\!]_{eval}$ left open (possibly undecidable!)

Ordering event parameters: $\leq_p, \perp_p, \top_p$

- Strengthening only upon re-distribution
- How to strengthen „pay 10€" or „play with 50% quality"?
- Specify rights as left-open intervals and duties as right-open intervals
  - „Play with at most 75% quality"
    $\forall v \in \{\perp_p, \ldots, \top_p\} : E(play \mapsto \{quality \mapsto v\}) \rightarrow eval(\perp_p \leq_p v \leq_p 75)$
  - Strengthened to „play with at most 50% quality"
    $\forall v \in \{\perp_p, \ldots, \top_p\} : E(play \mapsto \{quality \mapsto v\}) \rightarrow eval(\perp_p \leq_p v \leq_p 50)$
  - „Pay at least €10"
    $\forall v \in \{\perp_p, \ldots, \top_p\} : E(pay \mapsto \{obj \mapsto o, amount \mapsto v\}) \rightarrow eval(10 \leq_p v \leq_p \top_p)$
  - Strengthened to „pay at least €15"
    $\forall v \in \{\perp_p, \ldots, \top_p\} : E(pay \mapsto \{obj \mapsto o, amount \mapsto v\}) \rightarrow eval(15 \leq_p v \leq_p \top_p$

Ordering event names: $\leq_e, \perp_e, \top_e$



- Expressed as disjunction

Semantics of events $\models_e \subseteq \mathcal{S} \times \Psi$

- Substitutions: for a variable event e;
  $v \in Var$, $vn \in \mathrm{dom}(v)$, $x \in Var(vn)$
  $e[vn \mapsto x]$ is the result of simultaneously replacing all occurrences of vn by x in e
- VarsIn(e) is the set of variables in a variable event e
- $Inst_\varepsilon : \mathcal{VE} \rightarrow \mathbb{P}(\mathcal{E})$ generates all ground substitutions of an event

$$VarsIn(e) = \{vn_1 \mapsto VS_1, \ldots, vn_k \mapsto VS_k\}$$
$$\Rightarrow Inst_\varepsilon(e) = \{e[vn_1 \mapsto vv_1, \ldots, vn_k \mapsto vv_k] : \textstyle\bigwedge_{i=1}^{k} vv_i \in VS_i\}$$

- Semantics of events

$$\forall e' \in maxRefEv;\ e \in \mathcal{VE}\ \exists e'' \in \mathcal{E}$$
$$(e', actual) \models_e E(e) \Leftrightarrow e'\ refinesEv\ e'' \wedge e'' \in Inst_\varepsilon(e)$$
$$\wedge\ (e', intended) \models_e I(e) \Leftrightarrow e'\ refinesEv\ e'' \wedge e'' \in Inst_\varepsilon(e)$$

Event-Based Usage Control

- II.1 Requirements
- II.2 Events, refinements, traces
- **II.3 Specification-level policies**
- II.4 Implementation-level policies
- II.5 Video examples
- II.6 Analysis problems

Specification-level policies

$$\Phi ::= (\Phi) \mid \Psi \mid \underline{false} \mid \Phi\ \underline{implies}\ \Phi \mid \underline{forall}\ VName\ \underline{in}\ VVal : \Phi \mid$$
$$\Phi\ \underline{until}\ \Phi \mid \Phi\ \underline{after}\ \mathbb{N} \mid \underline{replim}(\mathbb{N}, \mathbb{N}, \mathbb{N}, \Psi) \mid \underline{repuntil}(\mathbb{N}, \Psi, \Phi)$$

- Plus the usual macros
- Will use usual symbols instead

Semantics $\models_f \subseteq (Trace \times \mathbb{N}) \times \Phi$

$$\forall s \in Trace;\ t \in \mathbb{N};\ \varphi \in \Phi \bullet (s,t) \models_f \varphi \Leftrightarrow \varphi \neq \underline{false} \wedge$$
$$\boxed{\exists e \in \mathcal{VE} \bullet (\varphi = E(e) \vee \varphi = I(e)) \wedge \exists e' \in s(t) : e' \models_e \varphi}$$
$$\boxed{\vee \exists \psi, \chi : \Phi \bullet \varphi = \psi\ implies\ \chi \wedge \neg((s,t) \models_f \psi) \vee (s,t) \models_f \chi}$$
$$\boxed{\vee \exists \gamma \in \Gamma \bullet \varphi = eval(\gamma) \wedge [\![\varphi]\!]_{eval} = true}$$
$$\boxed{\begin{array}{l} \vee \exists vn \in VName;\ vs \in VVal;\ \psi \in \Phi \bullet \\ \quad \varphi = (\underline{forall}\ \ vn\ in\ vs : \psi) \wedge \forall vv \in vs \bullet (s,t) \models_f \psi[vn \mapsto vv] \end{array}}$$
$$\boxed{\begin{array}{l} \vee \exists \psi, \chi \in \Phi \bullet \varphi = \psi\ \underline{until}\ \chi \wedge (\forall v \in \mathbb{N} \bullet t \leq v \Rightarrow (s,v) \models_f \psi \\ \quad \vee \exists u \in \mathbb{N} \bullet t < u \wedge (s,u) \models_f \chi \wedge \forall v \in \mathbb{N} \bullet t \leq v < u \Rightarrow (s,v) \models_f \psi) \\ \vee \exists i \in \mathbb{N};\ \psi \in \Phi \bullet \varphi = \psi\ after\ i \wedge (s, t+i) \models_f \psi \end{array}}$$
$$\boxed{\begin{array}{l} \vee \exists i \in \mathbb{N}_1;\ m,n \in \mathbb{N};\ \psi \in \Psi \bullet \\ \quad \varphi = \underline{replim}(i,m,n,\psi) \wedge \\ \qquad m \leq \sum_{j=1}^{i} \#\{ie \in \mathcal{S} \mid ie \in s(t+j) \wedge ie \models_f \psi\} \leq n \\ \vee \exists n \in \mathbb{N};\ \psi \in \Psi, \chi \in \Phi \bullet \varphi = \underline{repuntil}(n, \psi, \chi) \\ \quad \wedge (\exists u \in \mathbb{N}_1 \bullet (s, t+u) \models_f \chi \wedge (\forall v \in \mathbb{N}_1 \bullet v < u \Rightarrow \neg((s,t+v) \models_f \chi)) \\ \quad\ \wedge (\sum_{j=1}^{u} \#\{ie : \mathcal{S} \mid ie \in s(t+j) \wedge ie \models_e \psi\}) \leq n) \\ \quad\ \vee \sum_{j=1}^{\infty} \#\{ie : \mathcal{S} \mid ie \in s(t+j) \wedge ie \models_e \psi\}) \leq n) \end{array}}$$

Three Examples

Quality of playing sng.mp3
must not exceed 50% before paying €10.

Non-anonymized data must not leave system without notifying admin.

Accounts are suspended after three consecutive failed login attempts. They can be reactivated within a week upon a signed request, sent over email by their owner. After one week without receiving a reactivation email, suspended accounts will be closed.

Remarks

- No liveness
  - But can be expressed
- Decidability
  - Depends on eval
- Complexity
  - Terrible – but intended to be used for runtime monitoring
- Expressivity
  - Use templates instead

Event-Based Usage Control

- II.1 Requirements
- II.2 Events, refinements, traces
- II.3 Specification-level policies
- **II.4 Implementation-level policies**
- II.5 Video examples
- II.6 Analysis problems

Implementation-Level Policies

- ILPs are Event-Condition-Action rules
- Conditions

$$\begin{aligned}\Phi^- ::= \; & (\Phi^-) \mid \Psi \mid \underline{false}^- \mid \Phi^- \; \underline{implies}^- \; \Phi^- \mid \underline{forall}\; VName\; \underline{in}\; VVal : \Phi^- \mid \\ & \Phi^- \; \underline{since}^- \; \Phi^- \mid \boxdot\, \Phi^- \mid \Phi^- \; \underline{before}^- \; \mathbb{N} \mid \\ & \underline{replim}^-(\mathbb{N}, \mathbb{N}, \mathbb{N}, \Psi) \mid \underline{repsince}^-(\mathbb{N}, \Psi, \Phi^-)\end{aligned}$$

- Actions
  - Inhibition, modification, execution

Semantics of Conditions $\models_{f^-} \subseteq (Trace \times \mathbb{N}) \times \Phi$

$$\begin{aligned}
&\forall s \in Trace;\; t \in \mathbb{N};\; \pi \in \Phi^- \bullet \wedge (s,t) \models_{f^-} \pi \Leftrightarrow (\pi \neq \overline{false}^-) \wedge \\
&\quad (\exists e \in \mathcal{VE} \bullet (\pi = E(e) \vee \pi = I(e)) \wedge \exists e' \in s(t) : e' \models_e \pi \\
&\vee\; \exists \gamma \in \Gamma \bullet \pi = eval(\gamma) \wedge [\![\pi]\!]_{eval} = true \\
&\vee\; \exists vn \in VName;\; vs \in VVal;\; \psi \in \Phi^- \bullet \\
&\qquad \pi = (\underline{forall}\;\; vn\; \underline{in}\; vs : \psi) \wedge \forall vv \in vs \bullet (s,t) \models_{f^-} \psi[vn \mapsto vv] \\
&\vee\; \exists \psi, \chi \in \Phi^- \bullet \pi = \psi\; \underline{implies}^-\; \chi \wedge \neg((s,t) \models_{f^-} \psi) \vee (s,t) \models_{f^-} \chi \\
&\vee\; \exists \psi \in \Phi^- \bullet \pi = \boxdot\, \psi \wedge \forall u \in \mathbb{N} \bullet u \leq t \Rightarrow (s,u) \models_{f^-} \psi \\
&\vee\; \exists i \in \mathbb{N};\; \psi \in \Phi^- \bullet \pi = \psi\; \underline{before}^-\; i \wedge t \geq i \wedge (s, t-i) \models_{f^-} \psi \\
&\vee\; \exists i,m,n \in \mathbb{N};\; \psi \in \Psi;\; e \in \mathcal{E} \bullet \varphi = \underline{replim}^-(i,m,n,\psi) \\
&\quad \wedge m \leq (\textstyle\sum_{j=0}^{\min\{i,t\}} \left|\{ie \in \mathcal{S} \mid ie \in s(t-j) \wedge ie \models_e \psi\}\right| \leq n \\
&\vee\; \exists n \in \mathbb{N};\; \psi \in \Psi;\; \chi \in \Phi;\; e \in \mathcal{E} \bullet \varphi = \underline{repsince}^-(n, \psi, \chi) \\
&\quad \wedge \big((\exists u \in \mathbb{N}_1 \bullet t \geq u \wedge (s, t-u) \models_f \chi \wedge (\forall v \in \mathbb{N} \bullet v < u \Rightarrow \neg((s,t-v) \models_f \chi)) \\
&\quad\; \wedge (\textstyle\sum_{j=0}^{u} \#\{ie \in \mathcal{S} \mid ie \in s(t-j) \wedge ie \models_e \psi\} \leq n)) \\
&\quad \vee (\textstyle\sum_{j=0}^{t} \#\{ie \in \mathcal{S} \mid ie \in s(t-j) \wedge ie \models_e \psi\} \leq n)\big)
\end{aligned}$$

ECA rules

- $t \subseteq \mathcal{VE} \times \Phi^-$ describes trigger event and condition:

  $t(ve, \varphi) \Leftrightarrow \big(I(ve) \wedge (E(ve) \rightarrow \varphi)\big)$

- Effect

$$\begin{aligned}
m_{inh}(ve, \varphi) \quad & \Leftrightarrow \forall\, VarsIn(ve) :\; t(ve, \varphi) \rightarrow \neg E(ve) \\
m_{mod}(ve, \varphi, Mod) \quad & \Leftrightarrow \forall\, VarsIn(ve) :\; t(ve, \varphi) \rightarrow (\neg E(ve) \wedge m_{exc}(ve, \varphi, Mod)) \\
m_{exc}(ve, \varphi, Exc) \quad & \Leftrightarrow \forall\, VarsIn(ve) :\; t(ve, \varphi) \rightarrow \textstyle\bigwedge_{x_i \in Exc} I(x_i).
\end{aligned}$$

- Composition of ILPs

$$M \leftrightarrow \bigwedge_{i=1}^{n_1} m_{inh}(ve_i^{inh}, \varphi_i^{inh}) \wedge \bigwedge_{i=1}^{n_2} m_{mod}(ve_i^{mod}, \varphi_i^{mod}, Mod_i) \wedge \bigwedge_{i=1}^{n_3} m_{exc}(ve_i^{exc}, \varphi_i^{exc}, Exc_i)$$

Allow default

$$M_{default} \leftrightarrow \bigwedge_{e \in maxRefEv} I(e) \rightarrow \Big(E(e) \vee \bigvee_{\substack{(ve,\varphi):\ M \rightarrow m_{inh}(ve,\varphi) \\ \vee\ M \rightarrow m_{mod}(ve,\varphi,Mod)}} \exists\, VarsIn(ve) : e\ refinesEv\ ve \wedge \varphi\Big)$$

- Semantics of a set of ILPs: $\square(M_{complete})$ with $M_{complete} \leftrightarrow M \wedge M_{default}$

Example 1

Quality of playing sng.mp3 must not exceed 50% before paying €10

Example 2

Non-anonymized data must not leave system without notifying admin

Example 3

Accounts are suspended after three consecutive failed login attempts.
They can be reactivated within a week upon a signed request, sent over email by their owner.
After one week without receiving a reactivation email, suspended accounts will be closed.

Event-Based Usage Control

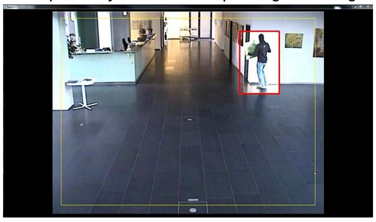
- II.1 Requirements
- II.2 Events, refinements, traces
- II.3 Specification-level policies
- II.4 Implementation-level policies
- **II.5 Video examples event-based UC:**
  - **Video 2: Context-aware UC on Android**
    http://www22.in.tum.de/fileadmin/demos/uc/uc4android/Demo3.wmv
  - **Video 3: UC for camera surveillance**
    http://www22.in.tum.de/fileadmin/demos/uc/Demo4-UC4NEST-Demo-v3.mp4
- II.6 Analysis problems

Example: UC4NEST scenario (I)

- Video surveillance on main floor of Fraunhofer IOSB building
  - People can authenticate as members of some groups using mobile devices and MC-MXT marker

Example: UC4NEST scenario (II)

- Video surveillance on main floor of Fraunhofer IOSB building
  - Intrusion detection service (IDS) observes the immediate proximity of a valuable painting of an ongoing art exhibition



Example: UC4NEST policies (III)

- (Group specific) privacy policies:
  - Show and track unknown persons
  - Show security personnel, but do not track
  - Neither track nor show staff members
- Alarm policy:
  - People causing an IDS alarm are tracked and visualized on the map
  - The video stream of the corresponding camera is temporarily shown to the operator
  - After the alarm has been handled, group specific policies are reactivated

Event-Based Usage Control

- II.1 Requirements
- II.2 Events, refinements, traces
- II.3 Specification-level policies
- II.4 Implementation-level policies
- II.5 Video examples event-based UC:
  - context-aware UC on Android
  - UC for camera surveillance
- **II.6 Analysis problems**

Analysis Problems

- Satisfiability
- Satisfiability of composed ILPs
- Strengthening policies for distribution
- Entailment of specification-level policies by ILPs
- Configuration of ILPs with free variables

Analysis Problems via $\models_{f^\pm} \subseteq \Phi^\pm$

- Use $\Phi^\pm = \Phi \mid \Phi^- \mid \Phi^- \ \underline{implies}^\pm \ \Phi$ and $\models_{f^\pm}$ with

$$\begin{aligned}&\forall \varphi \in \Phi^\pm;\ s \in Trace;\ t \in \mathbb{N} : \varphi \in \Phi \Rightarrow ((t,n) \models_{f^\pm} \varphi \leftrightarrow (t,n) \models_f \varphi)\\ &\wedge \varphi \in \Phi^- \Rightarrow ((t,n) \models_{f^\pm} \varphi \Leftrightarrow (t,n) \models_{f^-} \varphi)\\ &\wedge (\exists \psi \in \Phi^-;\ \chi \in \Phi : \varphi = \psi \ \underline{implies}^\pm \ \chi) \Rightarrow (\neg((t,n) \models_{f^-} \psi) \vee (t,n) \models_f \chi)\end{aligned}$$

for consistency:

$$\forall \varphi \in \Phi^\pm : \ \models_{f^\pm} \varphi \Leftrightarrow \exists t \in Trace : (t,0) \models_{f^\pm} \varphi$$

- Plus a chaotic model $\mathcal{M}$, a transformation τ_{OSL}, an encoding of the analysis problems in φ , a model checker and

$$\mathcal{M} \models \tau_{OSL}(\varphi) \quad \Leftrightarrow \quad \models_{f^\pm} \varphi$$

Model Checking

- Sanity constraint $C = \Box(\bigwedge_{e \in \mathcal{S}} E(e) \rightarrow I(e))$
- Entailment $\mathcal{M} \models \tau_{OSL}(C \rightarrow (\varphi_1 \rightarrow \varphi_2))$
- Checking capabilities: $\mathcal{M} \models \tau_{OSL}(C \wedge \Box(M_{complete}) \rightarrow \varphi)$
- Configure ILPs with free variables by enumeration and
  $\mathcal{M} \models \tau_{OSL}(\Box(C \wedge M_{complete}[X \mapsto x_i]) \rightarrow \varphi)$

Examples I: Strengthening

$$
\begin{aligned}
\varphi_1 &\leftrightarrow \Box(E(download \mapsto \{class \mapsto song\}) \rightarrow E(play \mapsto \{obj \mapsto ad\})\ \underline{within}\ 3)\\
\varphi_2 &\leftrightarrow \underline{repuntil}(3,\ E(display \mapsto \{obj \mapsto mov\}),\\
&\qquad \exists v \in \{5, \ldots, \top^p\} : E(pay \mapsto \{receiver \mapsto Bob, amount \mapsto v\}))
\end{aligned}
$$

$$
\begin{aligned}
\psi_1 &\leftrightarrow \Box(E(download \mapsto \{class \mapsto song\}) \rightarrow E(play \mapsto \{obj \mapsto ad\})\ \underline{within}\ 1)\\
\psi_2 &\leftrightarrow \underline{repuntil}(2,\ E(display \mapsto \{obj \mapsto mov\}),\\
&\qquad \exists v \in \{15, \ldots, \top^p\} : E(pay \mapsto \{receiver \mapsto Bob, amount \mapsto v\}))
\end{aligned}
$$

Examples II: Capabilities and Configuration

Specification-level policy

$$\varphi_3 = \neg E(play \mapsto \{object \mapsto cd\})\ \underline{until}\ E(pay \mapsto \{amount \mapsto 30\})$$

enforced by ILP

$$
\begin{aligned}
M^1 = m_{inh}(\ & play \mapsto \{object \mapsto cd\},\\
& \forall v \in \{30, \ldots, \top^p\} : \boxdot(\neg E(pay \mapsto \{amount \mapsto v\}))\ \underline{before}^-\ 1))
\end{aligned}
$$

and a configurable ILP with free variable X

$$
\begin{aligned}
M^2 = m_{inh}(\ & play \mapsto \{object \mapsto cd\},\\
& \boxdot(\neg E(pay \mapsto \{amount \mapsto X\}))\ \underline{before}^-\ 1))
\end{aligned}
$$

Translation

- Straightforward except for eval and cardinality:

$$
\begin{aligned}
&\forall \varphi, \psi, \varepsilon : \Phi;\ i, m, n : \mathbb{N} \bullet\\
&\quad \ldots \wedge\\
&\wedge (\varepsilon = \underline{repuntil}(0, \varphi, \psi) \implies \tau_{OSL}(\varepsilon) = \mathtt{X}(\tau_{OSL}(\psi)\ \mathtt{V}\ \tau_{OSL}(\neg \varphi)))\\
&\wedge ((n > 0 \wedge \varepsilon = \underline{repuntil}(n, \varphi, \psi)) \implies \tau_{OSL}(\varepsilon) =\\
&\quad \tau_{OSL}(\underline{after}(1, \psi \vee (\varphi \wedge \underline{repuntil}(n-1, \varphi, \psi)))\\
&\quad \mid \mathtt{X}(\tau_{OSL}((\psi \vee (\underline{after}(1, \varphi) \wedge (\underline{after}(1, \psi) \vee repuntil(n-1, \varphi, \psi))))) \ \mathtt{V}\ \tau_{OSL}(\neg\varphi)))\\
&\wedge ((\varepsilon = \underline{replim}(i, m, n, \varphi) \wedge (i = 0 \vee m > n \vee (i > 0 \wedge i < m))) \implies \tau_{OSL}(\varepsilon) = 0)\\
&\wedge ((\varepsilon = \underline{replim}(i, 0, 0, \varphi) \wedge i > 1) \implies \tau_{OSL}(\varepsilon) = \tau_{OSL}(during(i, \neg\varphi)))\\
&\wedge ((\varepsilon = \underline{replim}(1, 0, n, \varphi)) \implies \tau_{OSL}(\varepsilon) = 1)\\
&\wedge ((\varepsilon = \underline{replim}(1, 1, n, \varphi) \wedge n \geq 1) \implies \tau_{OSL}(\varepsilon) = \mathtt{X}\tau_{OSL}(\varphi))\\
&\wedge ((\varepsilon = \underline{replim}(i, m, n, \varphi) \wedge (i > 1 \wedge i \geq m \wedge n \geq m \wedge m > 0)) \implies \tau_{OSL}(\varepsilon) =\\
&\quad \mathtt{X}((\tau_{OSL}(\varphi \wedge \underline{replim}(i-1, m-1, n-1, \varphi))) \mid (!\tau_{OSL}(\varphi \wedge \underline{replim}(i-1, m, n, \varphi)))))\\
&\wedge ((\varepsilon = \underline{replim}(i, 0, n, \varphi) \wedge (i > 1 \wedge n > 0)) \implies \tau_{OSL}(\varepsilon) =\\
&\quad \mathtt{X}((\tau_{OSL}(\varphi \wedge \underline{replim}(i-1, 0, n-1, \varphi))) \mid (\tau_{OSL}(\neg \varphi \wedge \underline{replim}(i-1, 0, n, \varphi)))).
\end{aligned}
$$

Discussion

- Said before: Complexity prohibitive in principle; implementation with NuSMV not that bad
- eval function possibly undecidable
- Simple (manual) abstractions help

Agenda

- Part I: Introduction
- Part II: Event-Based Usage Control
- **Part III: Data-Centric Usage Control slides by Enrico Lovat**
- Part IV: Quantitative Usage Control
- Part V: Local Single-Layer Enforcement
- Part VI: Distributed Enforcement
- Part VII: Cross-Layer Enforcement
- Part VIII: Policy Derivation
- Part IX: Discussion

Data-Centric Usage Control

- **III.1 Motivation and intuition**
- III.2 Formalization
- III.3 Demo video

Usage Control: Data not Representations!

Event Processing, Runtime Verification, … **EVENTS**

⇩

Technical

"Don't COPY" or "Do DELETE"

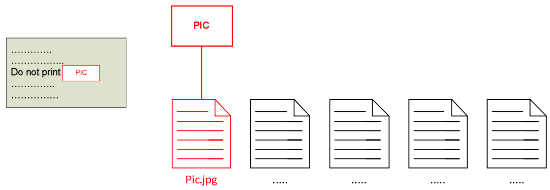
DATA

⇩

Intuitive

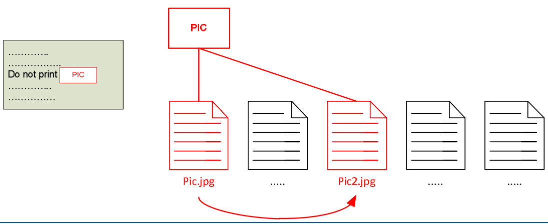
Usage Control with Data Flow detection

- Example



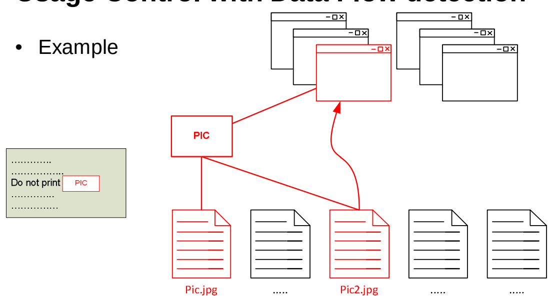
Usage Control with Data Flow detection

- Example



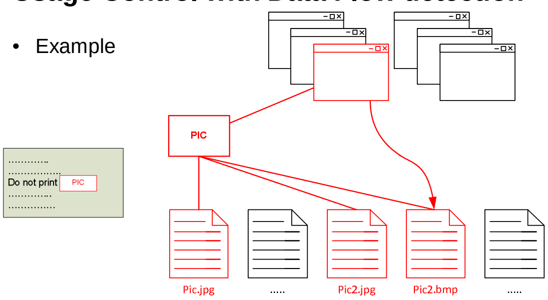
Usage Control with Data Flow detection

- Example

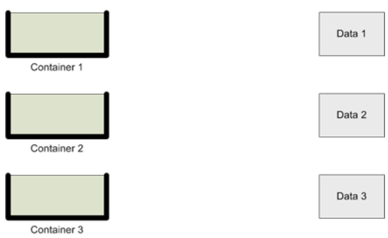


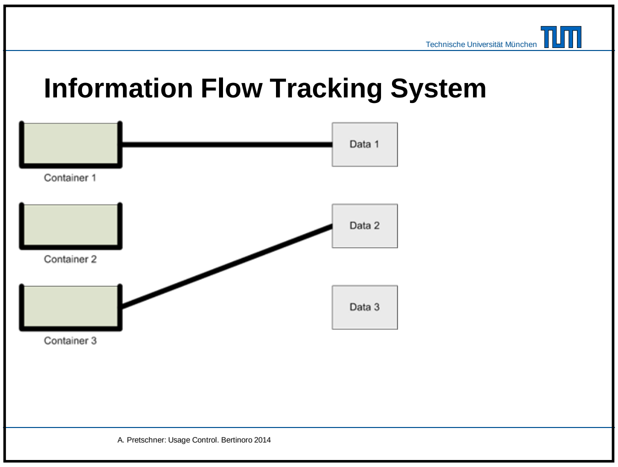
Usage Control with Data Flow detection

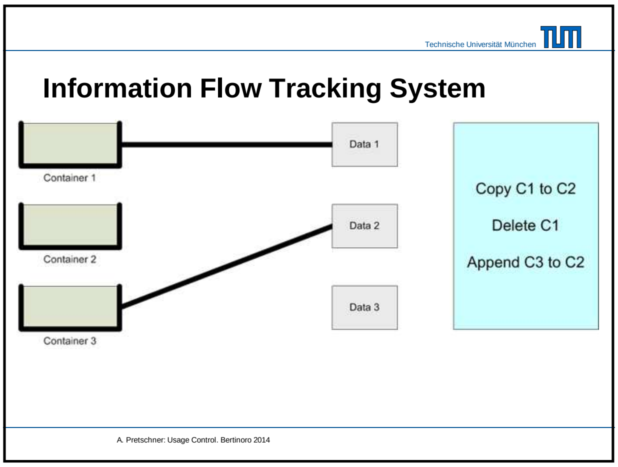
- Example

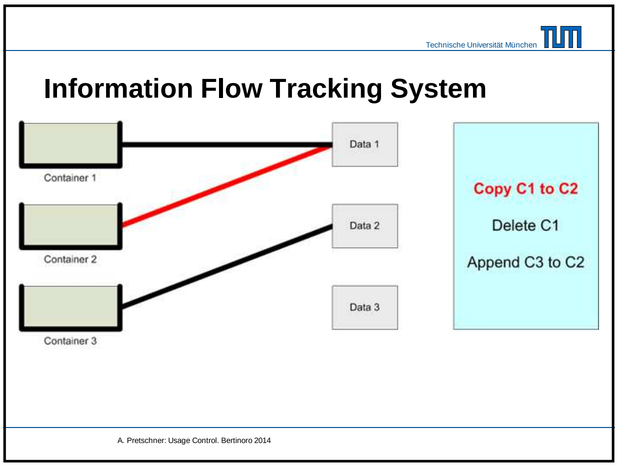


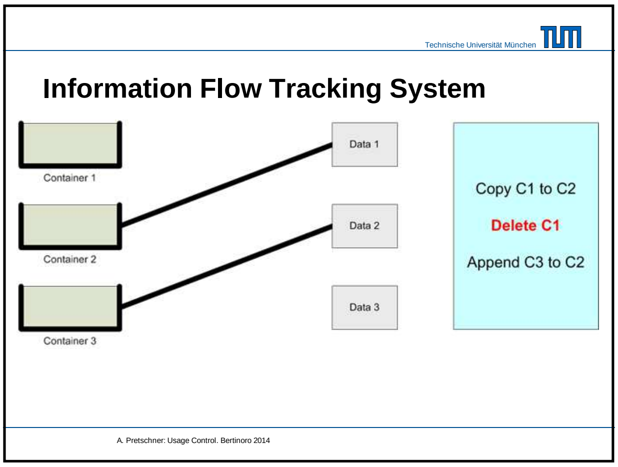
Information Flow Tracking System

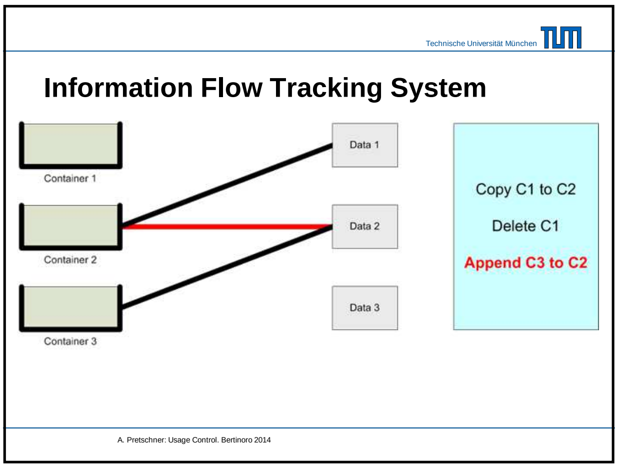


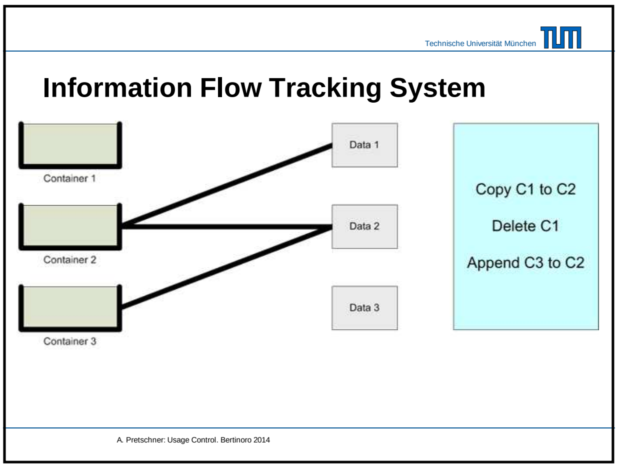

Technische Universität München
Information Flow Tracking System
Container 1
Container 2
Container 3
Data 1
Data 2
Data 3
A. Pretschner: Usage Control. Bertinoro 2014


Technische Universität München
Information Flow Tracking System
Container 1
Container 2
Container 3
Data 1
Data 2
Data 3
Copy C1 to C2
Delete C1
Append C3 to C2
A. Pretschner: Usage Control. Bertinoro 2014

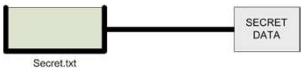

Technische Universität München
Information Flow Tracking System
Container 1
Container 2
Container 3
Data 1
Data 2
Data 3
Copy C1 to C2
Delete C1
Append C3 to C2
A. Pretschner: Usage Control. Bertinoro 2014


Technische Universität München
Information Flow Tracking System
Container 1
Container 2
Container 3
Data 1
Data 2
Data 3
Copy C1 to C2
Delete C1
Append C3 to C2
A. Pretschner: Usage Control. Bertinoro 2014

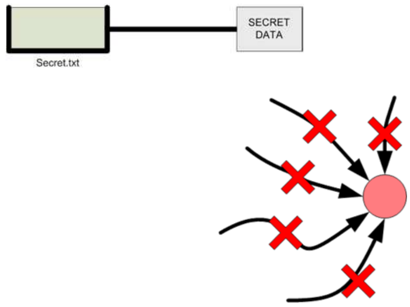

Technische Universität München
Information Flow Tracking System
Container 1
Container 2
Container 3
Data 1
Data 2
Data 3
Copy C1 to C2
Delete C1
Append C3 to C2
A. Pretschner: Usage Control. Bertinoro 2014


Technische Universität München
Information Flow Tracking System
Container 1
Container 2
Container 3
Data 1
Data 2
Data 3
Copy C1 to C2
Delete C1
Append C3 to C2
A. Pretschner: Usage Control. Bertinoro 2014

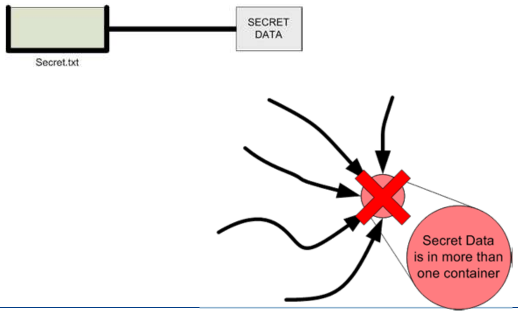
Information Flow Tracking System



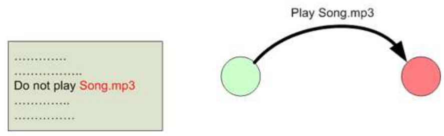
Information Flow Tracking System



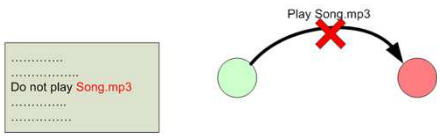
Information Flow Tracking System



Information Flow Tracking System

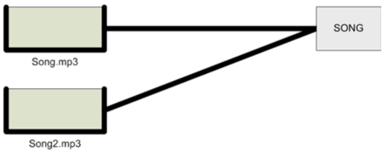


Information Flow Tracking System

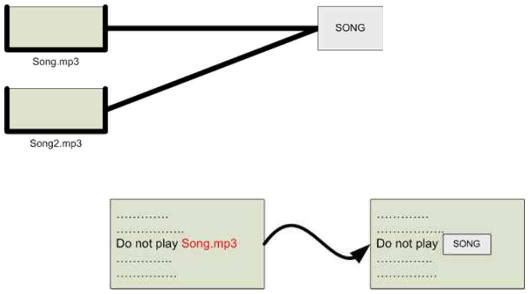


Information Flow Tracking System

Information Flow Tracking System



Information Flow Tracking System



Data-Centric Usage Control

- III.1 Motivation and intuition
- **III.2 Formalization**
- III.3 Demo video

Four ingredients

- Data state
- Transition relation capturing data flows
- Adjusted refinement and semantics of events
- New operators

Data State

$$\Sigma = \underbrace{(C \rightarrow 2^D)}_{\text{STORAGE}} \times \underbrace{(C \rightarrow 2^C)}_{\text{ALIAS}} \times \underbrace{(F \rightarrow C)}_{\text{NAMING}}$$

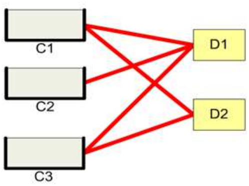
D : Set of **Data**
C : Set of data-**Containers**
F : Set of **Names** for containers

A : Set of **Actions**

Σ : Set of Information Flow **States**

Data State: Storage

$$\Sigma = \underbrace{(C \rightarrow 2^D)}_{\text{STORAGE}} \times \underbrace{(C \rightarrow 2^C)}_{\text{ALIAS}} \times \underbrace{(F \rightarrow C)}_{\text{NAMING}}$$



Four ingredients

- Data state
- **Transition relation capturing data flows**
- Adjusted refinement and semantics of events
- New operators

Transition Relation

$$\mathcal{R} \subseteq \Sigma \times \mathbb{P}(\mathcal{S}) \rightarrow \Sigma$$

Ordering of events does not matter:

$$\forall \sigma \in \Sigma : \mathcal{R}(\sigma, \varnothing) = \sigma$$
$$\forall \sigma \in \Sigma;\ Es \subseteq \mathcal{S};\ e \in \mathcal{S} : e \in Es \implies \mathcal{R}(\sigma, Es) = \mathcal{R}(\mathcal{R}(\sigma, \{e\}), Es \setminus \{e\})$$

uniquely defines R

Transition Relation determines Data State

$$states : (Trace \times \mathbb{N}) \rightarrow \Sigma$$

$$states(t, 0) = \sigma_i$$
$$n > 0 \implies states(t, n) = \mathcal{R}(states(t, n-1), t(n-1))$$

Four ingredients

- Data state
- Transition relation capturing data flows
- **Adjusted refinement and semantics of events**
- New operators

Adjusted refinement

POLICY
....
....
always (not (E_{fst} (OPEN { (obj, song.mp3) }))) e_1
....
....

Policy

TRACE
....
....
OPEN {(obj, song.mp3), (quality, HIGH), (from, 00:00), ... } e_2
....
....

1 2 3 4 5 ... ... t

POLICY
....
....
always (not (E_{fst} (OPEN { $\boxed{\text{(obj, song.mp3)}}$ }))) e_1
....
....

Policy

TRACE
....
....
OPEN { $\boxed{\text{(obj, song.mp3)}}$ (quality, HIGH), (from, 00:00), ... } e_2
....
....

1 2 3 4 5 ... ... t

e_2 refinesEv e_1 iff they have the **same event name** and either:

-e1 and e2 are of the same event type
-the set of parameters and values in e1 is a subset of e2

POLICY

……………

always (not (E_{fst} (OPEN {(obj, SONG}))) e_1

……………

TRACE

……………

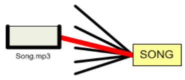
OPEN {(obj, song.mp3) (quality, HIGH), (from, 00:00), … } e_2

……………

e_2 refinesEv e_1 iff they have the **same event name** and either:

-e1 and e2 are of the same event type
-the set of parameters and values in e1 is a subset of e2

OR



-e1 is a **dataUsage** event and e2 is a **containerUsage** event
-e1 refers to data D, e2 to container C and D is stored (also) in C
-the set of parameters names and values in e1 is a subset of e2 (except for the obj parameter)

Adjusted refinement, formally

$$
\begin{aligned}
&refinesEv_i \subseteq (\mathcal{E} \times \Sigma) \times \mathcal{E}\\
&\forall e_1, e_2 \in \mathcal{E}\, \forall \sigma \in \Sigma : (e_2, \sigma)\, refinesEv_i\, e_1 \iff\\
&\quad (getclass(e_1) = getclass(e_2) \wedge e_2\, refinesEv\, e_1) \vee\\
&\quad ((getclass(e_1) = dataUsage \wedge getclass(e_2) = containerUsage \wedge\\
&\qquad \exists d \in \mathcal{D}\, \exists c \in \mathcal{C} :\ d \in \sigma.s(c) \wedge\\
&\qquad obj \mapsto d \in e_1.p \wedge obj \mapsto c \in e_2.p \wedge e_1.p \setminus \{obj \mapsto d\} \subseteq e_2.p \setminus \{obj \mapsto c\}))
\end{aligned}
$$

Semantics of events $\models_{e,i} \subseteq (\mathcal{S} \times \Sigma) \times \Psi$

$$
\begin{aligned}
&\forall e' \in maxRefEv\, \forall e \in \mathcal{VE}\, \forall \sigma \in \Sigma\ \exists e'' \in \mathcal{E} :\\
&\quad ((e', actual), \sigma) \models_{e,i} E(e) \iff (e', \sigma)\ refinesEv_i\ e'' \wedge e'' \in Inst_\varepsilon(e)\\
&\wedge\ ((e', intended), \sigma) \models_{e,i} I(e) \iff (e', \sigma)\ refinesEv_i\ e'' \wedge e'' \in Inst_\varepsilon(e)
\end{aligned}
$$

Four ingredients

- Data state
- Transition relation capturing data flows
- Adjusted refinement and semantics of events
- **New operators**

State-Based Operators

$\Phi_s ::= \underline{isNotIn}(\mathcal{D}, \mathbb{P}(\mathcal{C})) \mid \underline{isCombinedWith}(\mathcal{D}, \mathcal{D})$

$\Phi_i ::= \Phi \mid \Phi_s$

(plus $\underline{isOnlyIn}(d, Cs) \Leftrightarrow \underline{isNotIn}(d, \mathcal{C} \setminus Cs)$)

Semantics $\models_s \subseteq (Trace \times \mathbb{N}) \times \Phi_s$

$$
\begin{aligned}
&\forall t \in Trace;\ n \in \mathbb{N};\ \varphi \in \Phi_s;\ \sigma \in \Sigma :\ (t, n) \models_s \varphi \iff \sigma = states(t, n) \wedge\\
&\quad \exists d \in \mathcal{D}, Cs \subseteq \mathcal{C} : \varphi = \underline{isNotIn}(d, Cs) \wedge \forall c' \in \mathcal{C} :\\
&\qquad d \in \sigma.s(c') \implies c' \notin Cs\\
&\quad \vee \exists d_1, d_2 \in \mathcal{D} : \varphi = \underline{isCombinedWith}(d_1, d_2) \wedge \exists c' \in \mathcal{C} :\\
&\qquad d_1 \in \sigma.s(c') \wedge d_2 \in \sigma.s(c').
\end{aligned}
$$

(plus semantics for Φ_i and lifting to past, Φ_i^-)

Example Transition Relation: OS

$$
\begin{aligned}
&\forall s : \mathcal{C} \to \mathbb{P}(\mathcal{D}), \forall l : \mathcal{C} \to \mathbb{P}(\mathcal{C}), \forall f : \mathcal{P} \times F \to \mathcal{C}, \forall p \in \mathcal{P}, \forall fn \in F_{fName}, \forall fh \in F_{fHandle}:\\
&\quad \big((s, l, f), CreateFile\,(p, fh, fn), (s, l, f\,[(p, fh) \leftarrow f(p, fn)]\,)\big) \in \mathcal{R}.
\end{aligned}
$$

$$
\begin{aligned}
&\forall s : \mathcal{C} \to \mathbb{P}(\mathcal{D}), \forall l : \mathcal{C} \to \mathbb{P}(\mathcal{C}), \forall f : \mathcal{P} \times F \to \mathcal{C}), \forall p \in \mathcal{P}, \forall fh \in F_{fHandle}:\\
&\big((s, l, f), ReadFile\,(p, fh), (s[t \leftarrow s(f(p, fh)) \cup s(t)]_{t \in l^*(m_p)}, l, f)\big) \in \mathcal{R}
\end{aligned}
$$

$$
\begin{aligned}
&\forall s : \mathcal{C} \to \mathbb{P}(\mathcal{D}), \forall l : \mathcal{C} \to \mathbb{P}(\mathcal{C}), \forall f : \mathcal{P} \times F \to \mathcal{C}, \forall p \in \mathcal{P}, \forall fh \in F_{fHandle}:\\
&\big((s, l, f), WriteFile\,(p, fh), (s[f(p, fn) \leftarrow s(f(p, fn)) \cup s(m_p)], l, f)\big) \in \mathcal{R}
\end{aligned}
$$

… and so on

Technische Universität München

Data-Centric Usage Control

- III.1 Motivation and intuition
- III.2 Formalization
- **III.3 Demo video**
  - **Demo 4: Data-centric UC on Android via TaintDroid**
    http://www22.in.tum.de/fileadmin/demos/uc/uc4android/Demo2.wmv

A. Pretschner: Usage Control. Bertinoro 2014 — 103

Technische Universität München

So far …

- Generalization of access control to the future; permissions and duties
- Specification-level policies (SLP): what
- Implementation-level policies (ILP): how
  - Inhibition, modification, execution
- Event-based usage control
  - Semantic model: Traces of intended and actual events
  - First order future time temporal logic for SLPs
  - Event-condition-action rules with condition in first order past time temporal logic for ILPs

A. Pretschner: Usage Control. Bertinoro 2014 — 104

Technische Universität München

So far …

- Data-centric usage control
  - Data vs. representations/containers
  - Track data flow in-between representations
  - Policies extended to data usages
    - Data usage = any usage of a container that (potentially) contains the data
    - Operationally, if a system event involves a specific representation, we check which data items this representation potentially contains and if there are any applicable data usage policies

A. Pretschner: Usage Control. Bertinoro 2014 — 105

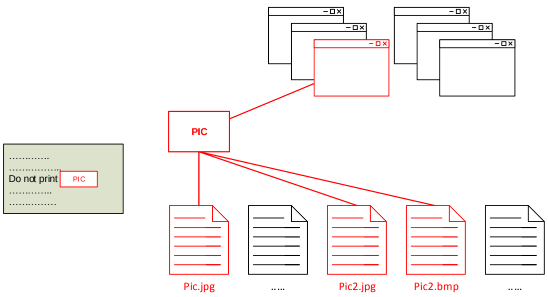
Technische Universität München

Agenda

- Part I: Introduction
- Part II: Event-Based Usage Control
- Part III: Data-Centric Usage Control
- **Part IV: Quantitative Usage Control slides by Enrico Lovat**
- Part V: Local Single-Layer Enforcement
- Part VI: Distributed Enforcement
- Part VII: Cross-Layer Enforcement
- Part VIII: Policy Derivation
- Part IX: Discussion

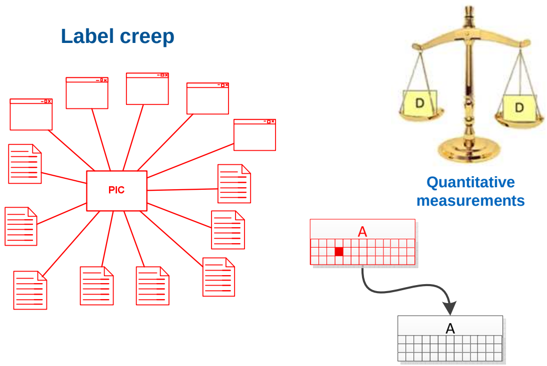
A. Pretschner: Usage Control. Bertinoro 2014 — 106

Technische Universität München



A. Pretschner: Usage Control. Bertinoro 2014 — 107

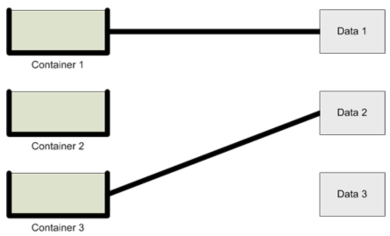
Technische Universität München



A. Pretschner: Usage Control. Bertinoro 2014 — 108

Possibilistic DFT

- Yes, data is (possibly) stored in this representation
- No, it is not



Quantitative DFT

- Yes, data is (possibly) stored in this representation
- No, it is not

(At most) x different units of data are stored in this representation

Quantitative Data Flow Tracking

- **Goal:** Model **how much** data is stored where
  - Where = in which representation
- Declassification criteria
  - Little data = No data ?
  - Avoid merging of declassified content
- Quantitative policies
  - Enforceable preventively or detectively
- Acceptable exception
  - A-posteriori policies (e.g. for auditing)
  - Security in practice

"If a file contains less than 10KB of data treat it as public (i.e. non sensitive)"

"No more than 10KB of data can leave the system"

"10KB of data have been leaked. This is acceptable"

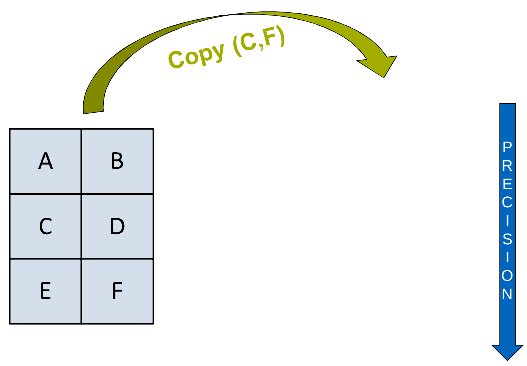
Quantitative Data Flow Tracking

Terms:

- ***UNIT of data***: smallest part of a representation that can be addressed by an action (e.g. OS → Blocks/Bytes, DB→Records, Window manager → Pixels)
- ***SIZE :*** number of units.

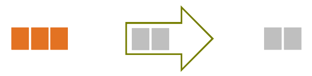
Assumptions:

- Each action of the system has a size.
- **We can observe how many units of representation are transferred by an action, but not which ones**

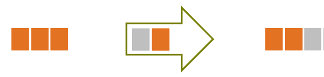


Idea

- Assumes a size for each action in the system



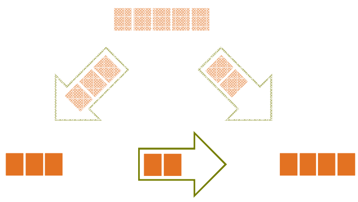
Size of action → Upper bound



Amount of different data in source → Upper bound

Technische Universität München

Data Provenance



Result depends on the "history" of data transfer

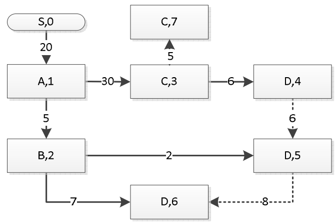
Provenance graph

A. Pretschner: Usage Control. Bertinoro 2014 115

Technische Universität München

Provenance Graph

- $G = (N, E)$
- $N = (C \times \mathbb{N})$



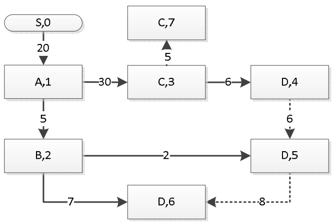
A node represents a container at a specific moment in time

A. Pretschner: Usage Control. Bertinoro 2014 116

Technische Universität München

Provenance Graph

- $G = (N, E)$
- $N = (C \times \mathbb{N})$
- $E = (N \times \mathbb{N} \times N)$



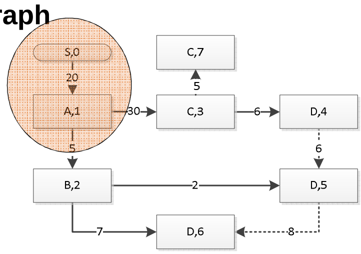
An edge represents an action that adds data to or removes data from a container

A. Pretschner: Usage Control. Bertinoro 2014 117

Technische Universität München

Provenance Graph

- $G = (N, E)$
- $N = (C \times \mathbb{N})$
- $E = (N \times \mathbb{N} \times N)$



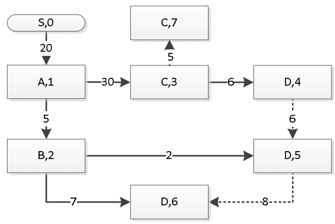
- $Init\ (c, m)$
- $Transfer\ (c_1, c_2, m)$
- $Truncate\ (c, m)$

A. Pretschner: Usage Control. Bertinoro 2014 118

Technische Universität München

Provenance Graph

- $G = (N, E)$
- $N = (C \times \mathbb{N})$
- $E = (N \times \mathbb{N} \times N)$



- $Init\ (c, m)$
- $Transfer\ (c_1, c_2, m)$
- $Truncate\ (c, m)$

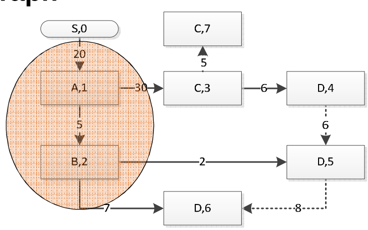
$1{:}\ Init\ (A, 20)$

A. Pretschner: Usage Control. Bertinoro 2014 119

Technische Universität München

Provenance Graph

- $G = (N, E)$
- $N = (C \times \mathbb{N})$
- $E = (N \times \mathbb{N} \times N)$



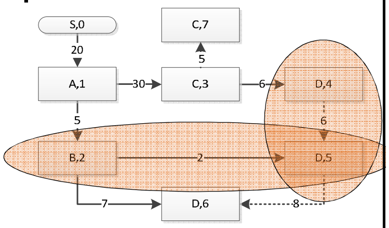
- $Init\ (c, m)$
- $Transfer\ (c_1, c_2, m)$
- $Truncate\ (c, m)$

$2{:}\ Transfer\ (A, B, 5)$

A. Pretschner: Usage Control. Bertinoro 2014 120

Provenance Graph

- $G = (N, E)$
- $N = (C \times \mathbb{N})$
- $E = (N \times \mathbb{N} \times N)$

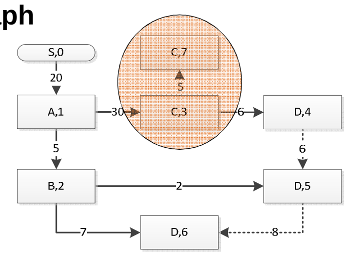


- *Init* (c, m)
- *Transfer* (c_1, c_2, m)
- *Truncate* (c, m)

5: *Transfer* $(B, D, 2)$

Provenance Graph

- $G = (N, E)$
- $N = (C \times \mathbb{N})$
- $E = (N \times \mathbb{N} \times N)$

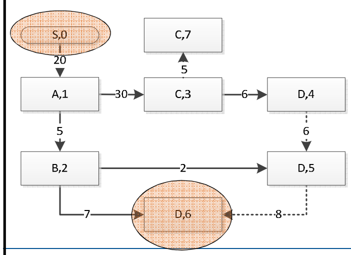


- *Init* (c, m)
- *Transfer* (c_1, c_2, m)
- *Truncate* (c, m)

7: *Truncate* $(C, 5)$

Provenance Graph

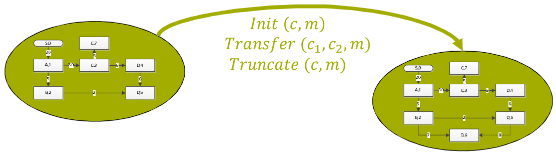
What is the maximum amount of different sensitive data possibly stored in D at time 6?



MAX-FLOW
((S,0),(D,6))

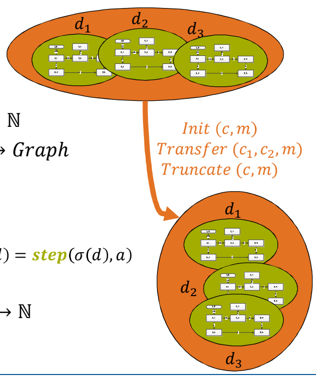
Formalization

- $Node = Container \times \mathbb{N}$
- $Graph = Node \times Node \to \mathbb{N}$
- ***Step***: $(Graph \times Event) \to Graph$



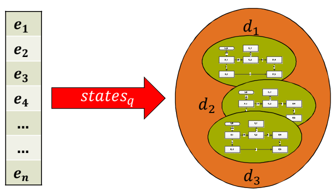
Formalization

- $Node = Container \times \mathbb{N}$
- $Graph = Node \times Node \to \mathbb{N}$
- ***Step***: $(Graph \times Event) \to Graph$
- $\Sigma = Data \to Graph$
- $\varrho: (\Sigma \times Event) \to \Sigma$
  - $\varrho(\sigma, a) = \sigma' \Rightarrow$
    $\forall d \in Data;\ \sigma'(d) = \mathbf{step}(\sigma(d), a)$
- $K: (Graph \times Container) \to \mathbb{N}$



Formalization (2)

- $states_q: (Trace \times \mathbb{N}) \to \Sigma$
  - $states_q(\mathrm{tr}, 0) = \sigma_i$
  - $states_q(\mathrm{tr}, t) = \varrho(\ states_q(\mathrm{tr}, t-1), tr(t-1))$



Technische Universität München

Formalization (2)

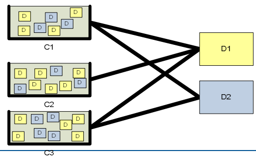
- $states_q : (Trace \times \mathbb{N}) \to \Sigma$
  - $states_q(\mathrm{tr}, 0) = \sigma_i$
  - $states_q(\mathrm{tr}, t) = \varrho(\, states_q(\mathrm{tr}, t-1), tr(t-1))$
- $\Phi_q = \boldsymbol{atMostInEach}\,(Data, \mathbb{N}, 2^{Container}) \mid \boldsymbol{atMostInSet}\,(Data, \mathbb{N}, 2^{Container})$ $\qquad \vDash_q$
- $\forall tr \in Trace, \forall t \in \mathbb{N}, \forall \phi \in \Phi_q, \forall \sigma \in \Sigma \cdot (tr, t) \vDash_q \phi \Leftrightarrow$ $\sigma = states_q(tr, t) \wedge \exists d \in Data, \exists C \in 2^{Container}, \exists Q \in \mathbb{N} \cdot$ $(\phi = atMostInEach(d, Q, C) \wedge \forall c \in C \cdot \boldsymbol{K}(\sigma(d), c) \leq Q) \vee$ $(\phi = atMostInSet(d, Q, C) \wedge \sum_{c \in C} \boldsymbol{K}(\sigma(d), c) \leq Q)$

A. Pretschner: Usage Control. Bertinoro 2014 127

Technische Universität München

State-based operators

$isNotIn\ (D_2, \{C_1, C_2\})$	true if data D_2 is stored in none of containers C_1 and C_2
$isNotIn(D_1, \{C_2, C_3\})$	true if no container other than C_2 or C_3 contains data D_1
$isCombinedWith\ (D_1, D_2)$	true if data D_1 and D_2 are stored in the same container
$atMostInEach\ (D_1, n, \{C_2, C_3\})$	true if both C_2 and C_3 contain less than n units of data D_1
$atMostInSet\ (D_2, n, \{C_1, C_3\})$	true if C_1 and C_3 contain in total less than n units of data D_1



A. Pretschner: Usage Control. Bertinoro 2014

Technische Universität München

State-based operators

$isNotIn\ (D_2, \{C_1, C_2\})$	true if data D_2 is stored in none of containers C_1 and C_2
$isNotIn(D_1, \{C_2, C_3\})$	true if no container other than C_2 or C_3 contains data D_1
$isCombinedWith\ (D_1, D_2)$	true if data D_1 and D_2 are stored in the same container
$atMostInEach\ (D_1, n, \{C_2, C_3\})$	true if both C_2 and C_3 contain less than n units of data D_1
$atMostInSet\ (D_2, n, \{C_1, C_3\})$	true if C_1 and C_3 contain in total less than n units of data D_1

"If a mail contains more than 10KB of sensitive data, then it cannot be sent"

$always(atMostInEach(d, 10KB, MAILS))$

$always(atMostInSet(d, 10KB, \mathrm{REMOVABLE}))$

"No more than 10KB of customer data can be saved on a removable device"

A. Pretschner: Usage Control. Bertinoro 2014

Technische Universität München

Theorem

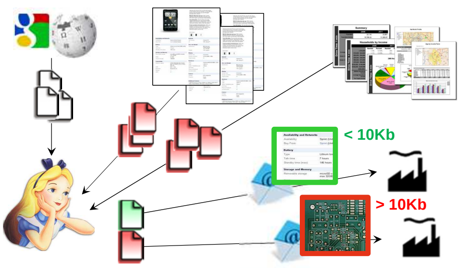
- Estimated number of tainted units in each node greater or equal than actual number of tainted units
- Induction over size of the provenance graph: different steps add nodes in a specific way

A. Pretschner: Usage Control. Bertinoro 2014 130

Technische Universität München

Experimental results

- Scenario: Phone specification

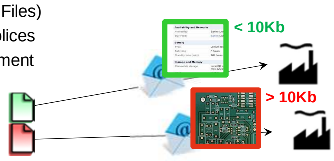


A. Pretschner: Usage Control. Bertinoro 2014 131

Technische Universität München

Experimental results

- Scenario: Phone specification
- Implementation: OpenBSD, Systrace
- Settings:
  - Repositories (→Files)
  - Monitor usage by Alice
  - Observe sent mails (→Files)
  - Enforce quantitative polices
    - Preventive enforcement



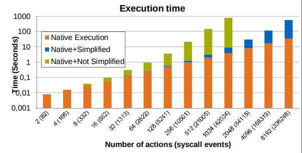
A. Pretschner: Usage Control. Bertinoro 2014 132

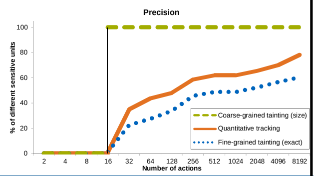
Results

- Precision = (Size – estimated # of tainted blocks) / (Size – exact # of tainted blocks)
  – If Size=Exact(=Estim), Precision=1
- Variables:
  – PS = Probability that source of a transfer is a specification (100% sensitive) vs existing report
  → No meaningful impact on precision
  – PN = Probabilty that destination of a transfer is a new report vs update existing report
  → Meaningful impact on precision
- 100 experiments for each (num_rep, PS, PN) triple

Experimental results

- Performance
  – Scalability issues
  – Graph simplification
- Precision
  – Reflects intuition
  – Trace dependent
  – Asymptotic behavior





Related work: quantitative information flow

- Information leakage defined by different measures: min-entropy, Shannon entropy, guessing entropy
- Require probability distribution of secrets (input to channel or system)
- In contrast, we want to protect one data item for which, in general, no probability distribution can be known

Concerns

- Precise meaning of numbers?
- Performance
- Coding and compression
  – May be assumed in controlled environments
- Technology possibly better suited for anomaly detection in IDS

Agenda

- Part I: Introduction
- Part II: Event-Based Usage Control
- Part III: Data-Centric Usage Control
- Part IV: Quantitative Usage Control
- **Part V: Local Single-Layer Enforcement**
- Part VI: Distributed Enforcement
- Part VII: Cross-Layer Enforcement
- Part VIII: Policy Derivation
- Part IX: Discussion

THURSDAY

Technische Universität München

Agenda

- Part I: Introduction
- Part II: Event-Based Usage Control
- Part III: Data-Centric Usage Control
- Part IV: Quantitative Usage Control
- **Part V: Local Single-Layer Enforcement**
- Part VI: Distributed Enforcement
- Part VII: Cross-Layer Enforcement
- Part VIII: Policy Derivation
- Part IX: Discussion

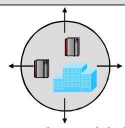
A. Pretschner: Usage Control. Bertinoro 2014 139

Technische Universität München

Prevention

- Provider can guarantee adherence to an obligation
- Control mechanisms in DRM
- Adobe LiveCycle, Windows RMS
- Amount of control platform-dependent

Detection



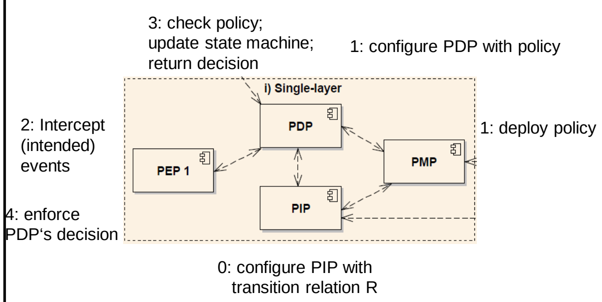
- Provider can detect violation of an obligation
- Take compensating action
- 100% security not desired
- Trustworthy signalers and monitors

Different requirements and trust models. Technically very similar.

A. Pretschner: Usage Control. Bertinoro 2014

Technische Universität München

Architecture



A. Pretschner: Usage Control. Bertinoro 2014 141

Technische Universität München

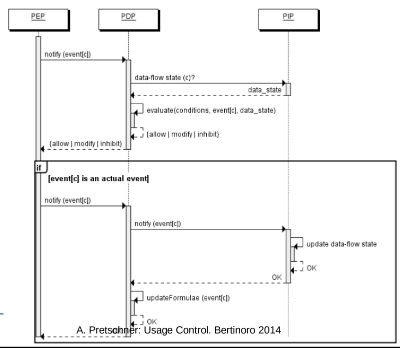
Runtime enforcement

- PMP manages policy livecycle
- Independent of deployed policies:
  - PEP intercepts intended and actual events
  - PIP implements transition relation and tracks data flows
- Depending on deployed policies:
  - PDP configured by ILPs
  - Technology: runtime verification, complex event processing
- Possible distinction in PEPs: signalers and monitors
  - Detective ~ preventive?

A. Pretschner: Usage Control. Bertinoro 2014 142

Technische Universität München

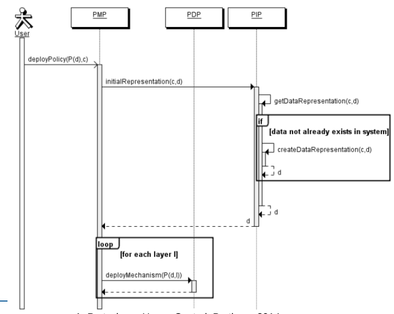
At Runtime



A. Pretschner: Usage Control. Bertinoro 2014 143

Technische Universität München

Deployment



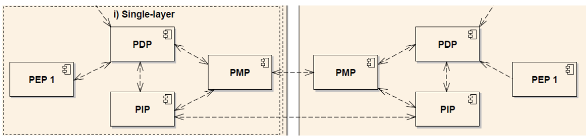
A. Pretschner: Usage Control. Bertinoro 2014 144

Instantiations

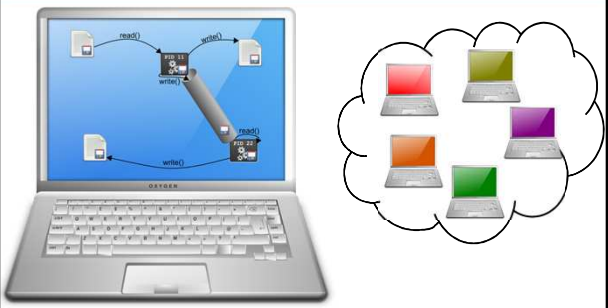
Agenda

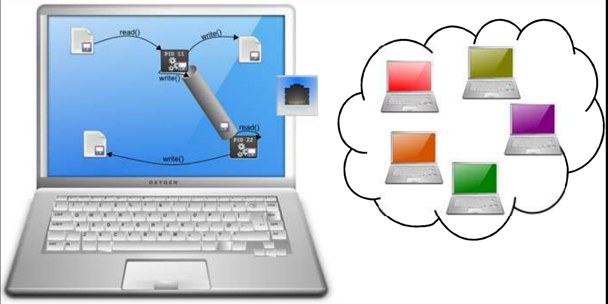
- Part I: Introduction
- Part II: Event-Based Usage Control
- Part III: Data-Centric Usage Control
- Part IV: Quantitative Usage Control
- Part V: Local Single-Layer Enforcement
- **Part VI: Distributed Enforcement slides by Florian Kelbert**
- Part VII: Cross-Layer Enforcement
- Part VIII: Policy Derivation
- Part IX: Discussion

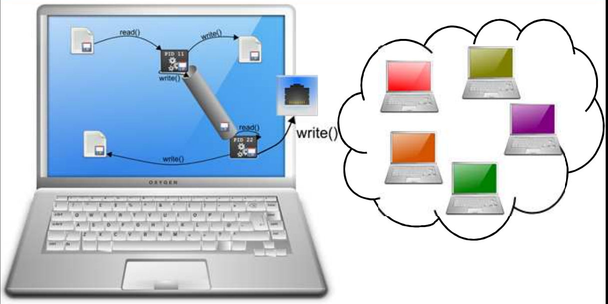
Architecture

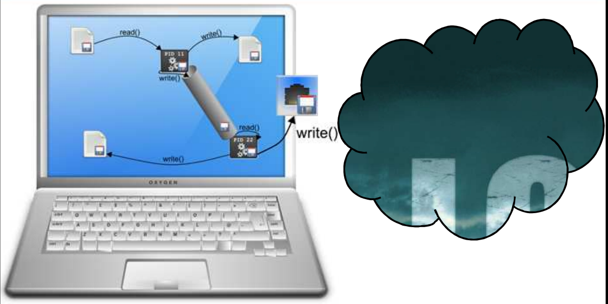


- So far: single-system data flow tracking
- E.g. at the level of the operating system
- Intercepting system calls using systrace
- Enforcement of state-based policies

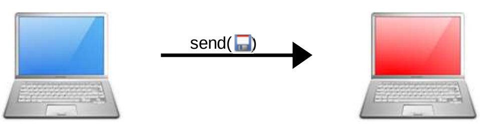








Goal



- Track the flow of data across systems
  - Connection Establishment
  - Data Transmission
- Enforce policy on receiving system
  - Sticking policies to data

Containers
Files
Processes
Pipes
Network sockets

Names
Filenames
File descriptors
Process IDs
Socket names

Principals
Processes

Actions
Systemcalls:
open(), read(), write(),
close(), pipe(), dup(),
unlink(), kill(), rename(),
execve(), fork(),
socket(), bind(), connect()
accept(), send(), recv(), ...

Setting

Technische Universität München

- Process with PID 11 running on host with IP Address A
- Process with PID 22 running on host with IP Address B

A. Pretschner: Usage Control. Bertinoro 2014 157

Setting

Technische Universität München

- Process with PID 11 running on host with IP Address A
- Process with PID 22 running on host with IP Address B

A. Pretschner: Usage Control. Bertinoro 2014 158

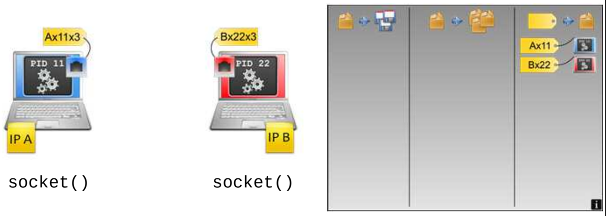
Setting

Technische Universität München

- Process with PID 11 running on host with IP Address A
- Process with PID 22 running on host with IP Address B

A. Pretschner: Usage Control. Bertinoro 2014 159

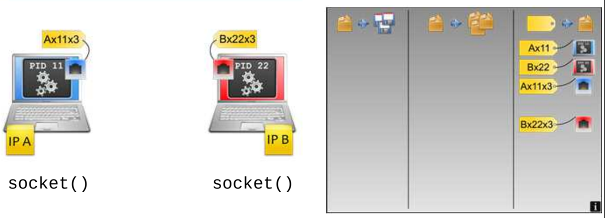
Technische Universität München



- **`socket()`**
  - Create an endpoint (Socket) for communication
  - Socket is identified by a file descriptor relative to the process

A. Pretschner: Usage Control. Bertinoro 2014 160

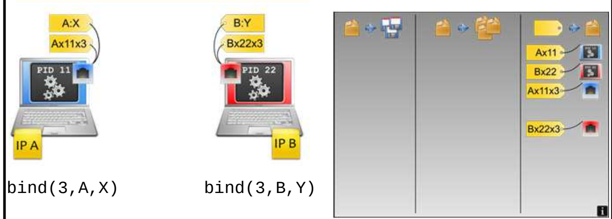
Technische Universität München



- **`socket()`**
  - Create an endpoint (Socket) for communication
  - Socket is identified by a file descriptor relative to the process

A. Pretschner: Usage Control. Bertinoro 2014 161

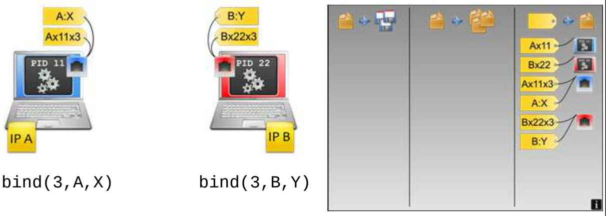
Technische Universität München



- **`bind()`**
  - Bind a name (IP Address + Port) to a socket
  - Either explicit or implicit on succeeding system call
  - PID 11 binds socket to IP Address A and Port X
  - PID 22 binds socket to IP Address B and Port Y

A. Pretschner: Usage Control. Bertinoro 2014 162

Technische Universität München



- **`bind()`**
  - Bind a name (IP Address + Port) to a socket
  - Either explicit or implicit on succeeding system call
  - PID 11 binds socket to IP Address A and Port X
  - PID 22 binds socket to IP Address B and Port Y

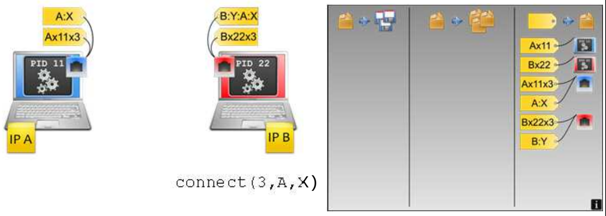
A. Pretschner: Usage Control. Bertinoro 2014 163

Technische Universität München

- **`listen()`**
  - Marks a socket as passive and listening for connections
  - Socket will accept incoming connection requests using `accept()`

A. Pretschner: Usage Control. Bertinoro 2014 164

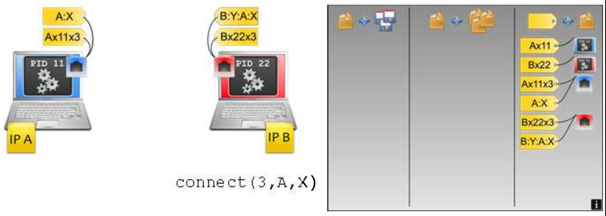
Technische Universität München



- **`connect()`**
  - Initiate a connection on a socket
  - Connects to the socket A:X that must be listening for connections

A. Pretschner: Usage Control. Bertinoro 2014 165

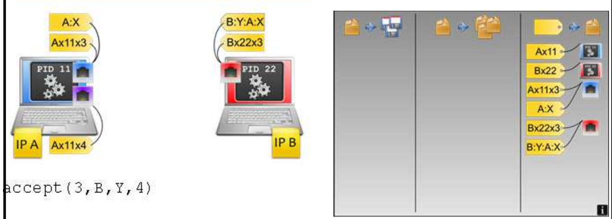
Technische Universität München



- **`connect()`**
  - Initiate a connection on a socket
  - Connects to the socket A:X that must be listening for connections

A. Pretschner: Usage Control. Bertinoro 2014 166

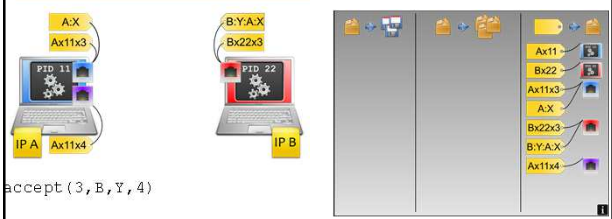
Technische Universität München



- **`accept()`**
  - Accept a connection on a socket
  - Creates a new connected socket
  - Original socket unaffected by this call
    - Will listen for further connection requests

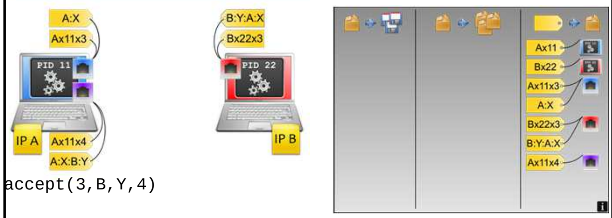
A. Pretschner: Usage Control. Bertinoro 2014 167

Technische Universität München

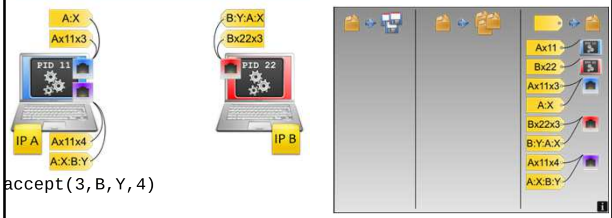


- **`accept()`**
  - Accept a connection on a socket
  - Creates a new connected socket
  - Original socket unaffected by this call
    - Will listen for further connection requests

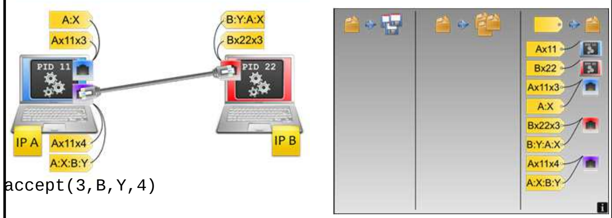
A. Pretschner: Usage Control. Bertinoro 2014 168



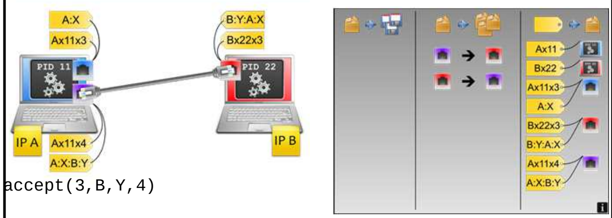
- `accept()`
  - Accept a connection on a socket
  - Creates a new connected socket
  - Original socket unaffected by this call
    - Will listen for further connection requests



- `accept()`
  - Accept a connection on a socket
  - Creates a new connected socket
  - Original socket unaffected by this call
    - Will listen for further connection requests



- `accept()`
  - Accept a connection on a socket
  - Creates a new connected socket
  - Original socket unaffected by this call
    - Will listen for further connection requests



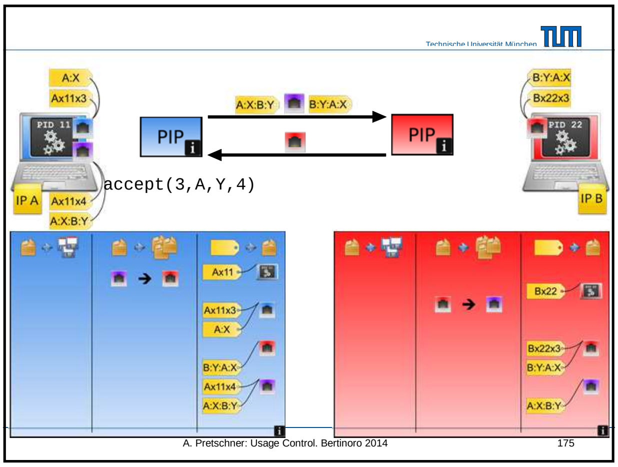
- `accept()`
  - Accept a connection on a socket
  - Creates a new connected socket
  - Original socket unaffected by this call
    - Will listen for further connection requests

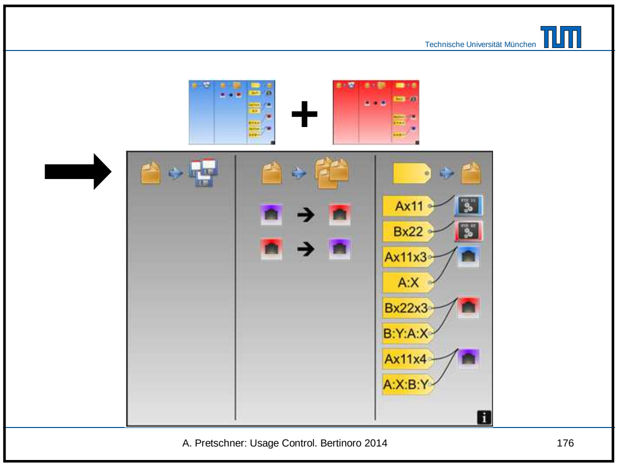
Distributed Data Flow State

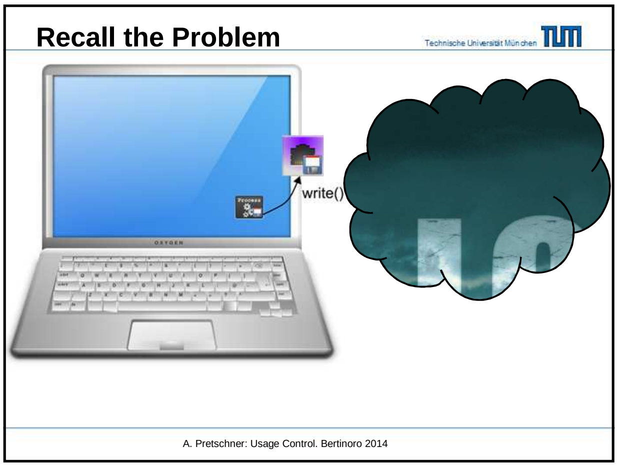
- Things are slightly more complicated: no global PIP
- Each system keeps track of its own Data Flow state
  - One PIP per system
- PIPs of independent systems must be synchronized
- Union of the local Data Flow states results in the global Data Flow state

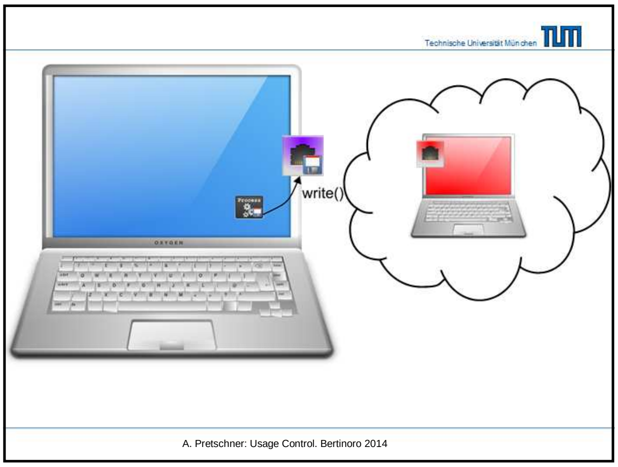
Distributed Data Flow State

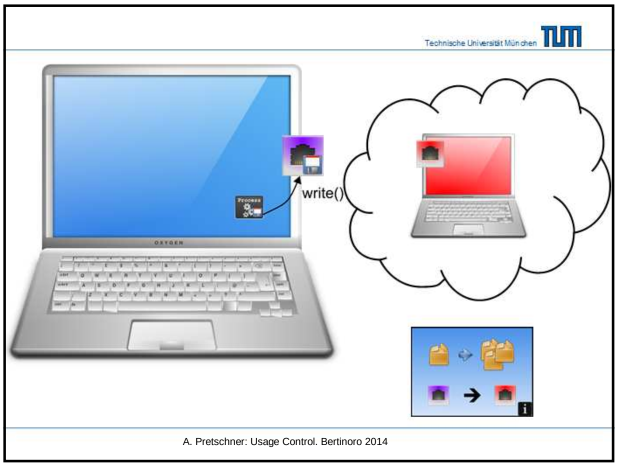
- Each system keeps track of its own Data Flow state
  - One PIP per system
- PIPs of independent systems must be synchronized
- Union of the local Data Flow states results in the global Data Flow state

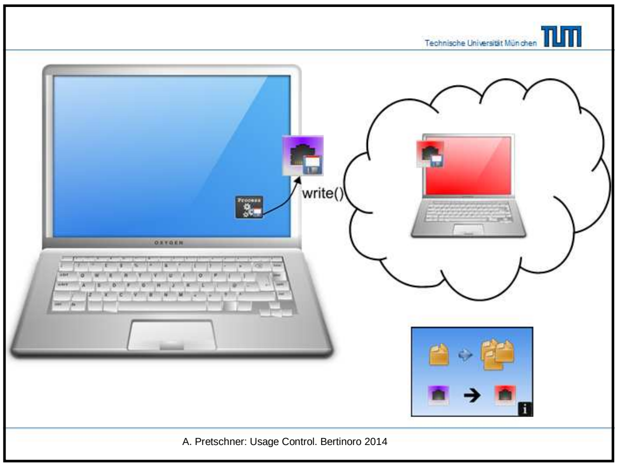

Technische Universität München
A:X
Ax11x3
A:X:B:Y
B:Y:A:X
PID 11
PIP
PIP
PID 22
B:Y:A:X
Bx22x3
accept(3,A,Y,4)
IP A
Ax11x4
A:X:B:Y
IP B
Ax11
Ax11x3
A:X
B:Y:A:X
Ax11x4
A:X:B:Y
Bx22
Bx22x3
B:Y:A:X
A:X:B:Y
A. Pretschner: Usage Control. Bertinoro 2014
175

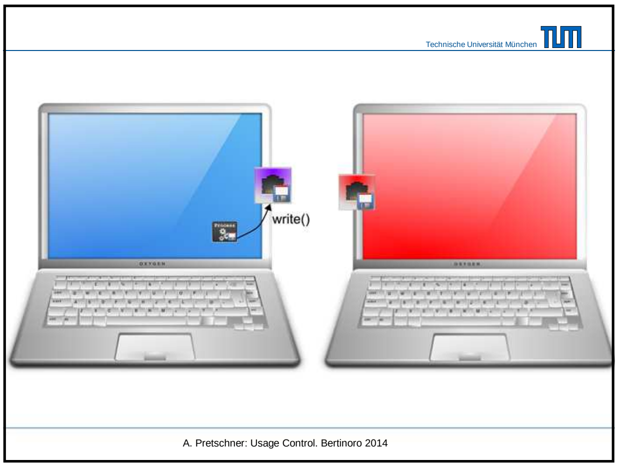

Technische Universität München
Ax11
Bx22
Ax11x3
A:X
Bx22x3
B:Y:A:X
Ax11x4
A:X:B:Y
A. Pretschner: Usage Control. Bertinoro 2014
176

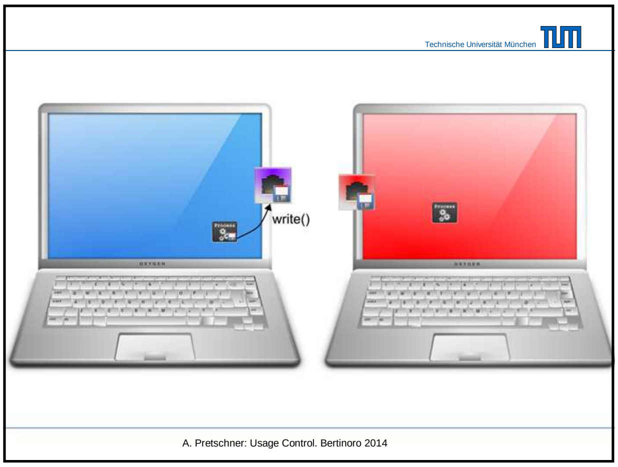

Recall the Problem
Technische Universität München
write()
A. Pretschner: Usage Control. Bertinoro 2014

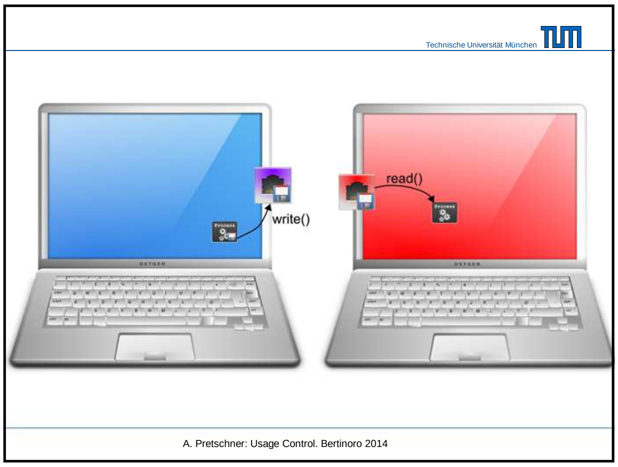

Technische Universität München
write()
A. Pretschner: Usage Control. Bertinoro 2014

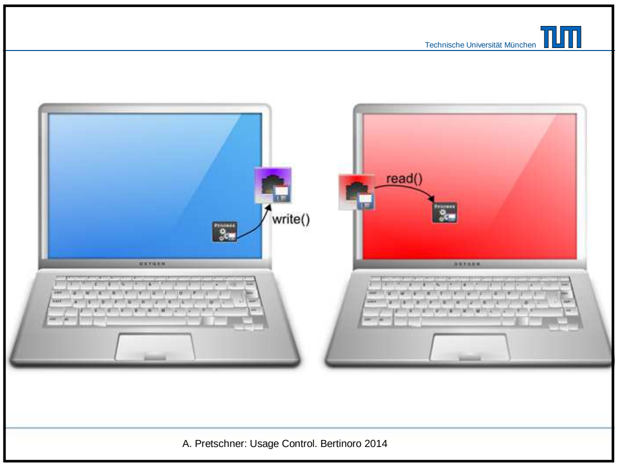

Technische Universität München
write()
A. Pretschner: Usage Control. Bertinoro 2014

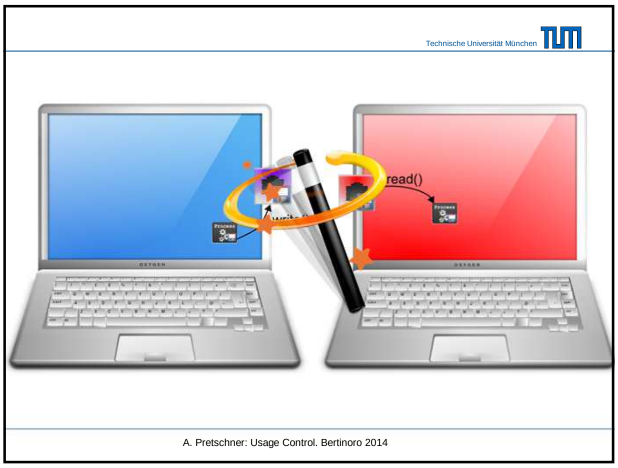

Technische Universität München
write()
A. Pretschner: Usage Control. Bertinoro 2014

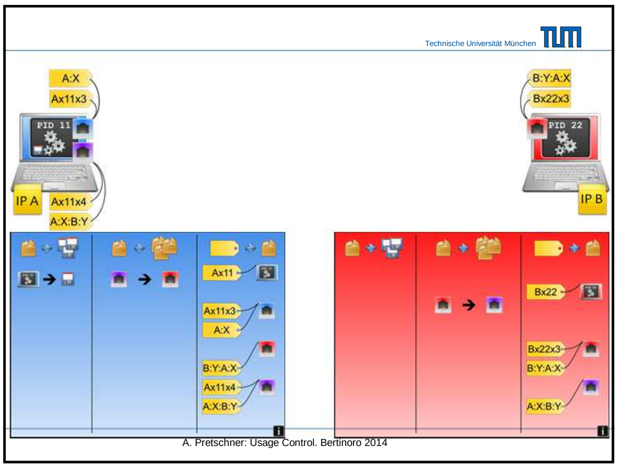

Technische Universität München
write()
A. Pretschner: Usage Control. Bertinoro 2014

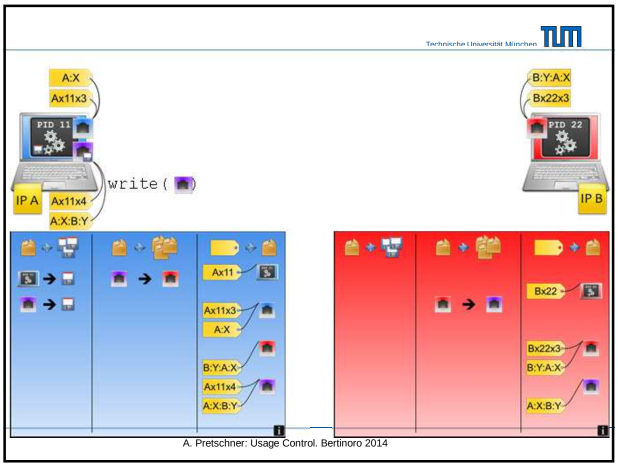

Technische Universität München
write()
A. Pretschner: Usage Control. Bertinoro 2014

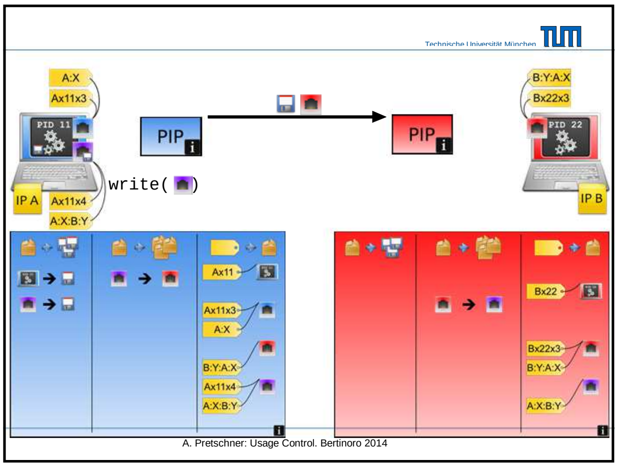

Technische Universität München
write()
read()
A. Pretschner: Usage Control. Bertinoro 2014

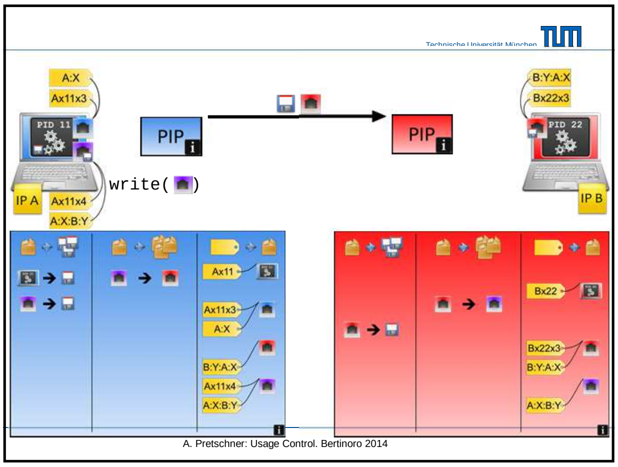

Technische Universität München
write()
read()
A. Pretschner: Usage Control. Bertinoro 2014

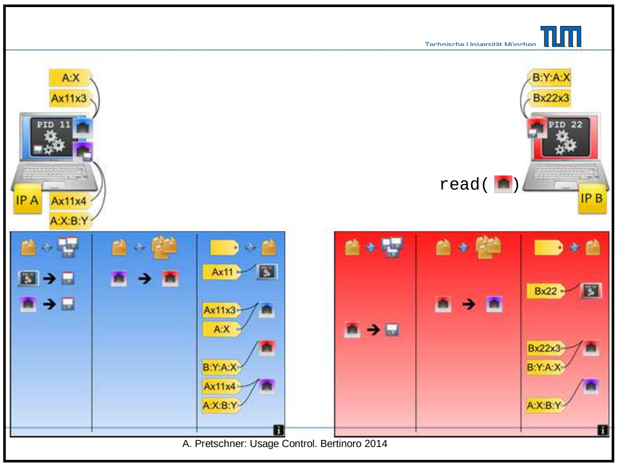

Technische Universität München
read()
A. Pretschner: Usage Control. Bertinoro 2014

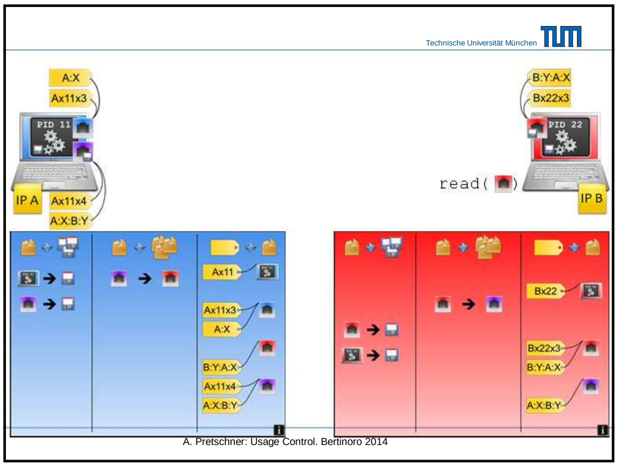

Technische Universität München
A:X
Ax11x3
PID 11
IP A
Ax11x4
A:X:B:Y
B:Y:A:X
Bx22x3
PID 22
IP B
Ax11
Ax11x3
A:X
B:Y:A:X
Ax11x4
A:X:B:Y
Bx22
Bx22x3
B:Y:A:X
A:X:B:Y
A. Pretschner: Usage Control. Bertinoro 2014

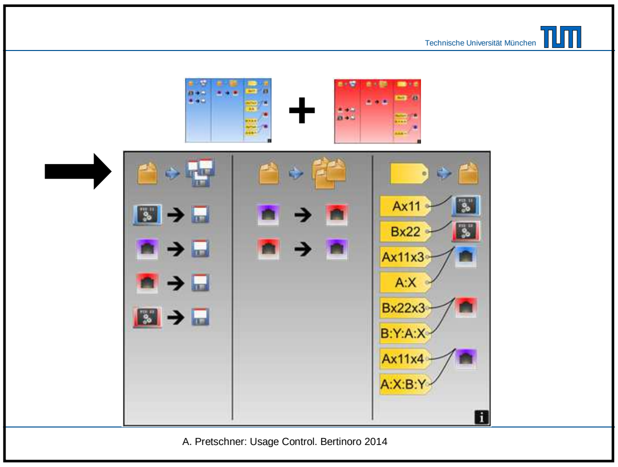

Technische Universität München
A:X
Ax11x3
PID 11
write( )
IP A
Ax11x4
A:X:B:Y
B:Y:A:X
Bx22x3
PID 22
IP B
Ax11
Ax11x3
A:X
B:Y:A:X
Ax11x4
A:X:B:Y
Bx22
Bx22x3
B:Y:A:X
A:X:B:Y
A. Pretschner: Usage Control. Bertinoro 2014


Technische Universität München
A:X
Ax11x3
PID 11
PIP
PIP
write( )
IP A
Ax11x4
A:X:B:Y
B:Y:A:X
Bx22x3
PID 22
IP B
Ax11
Ax11x3
A:X
B:Y:A:X
Ax11x4
A:X:B:Y
Bx22
Bx22x3
B:Y:A:X
A:X:B:Y
A. Pretschner: Usage Control. Bertinoro 2014

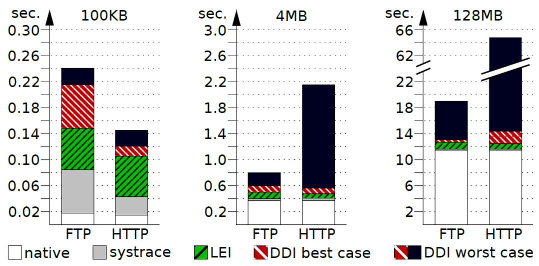

Technische Universität München
A:X
Ax11x3
PID 11
PIP
PIP
write( )
IP A
Ax11x4
A:X:B:Y
B:Y:A:X
Bx22x3
PID 22
IP B
Ax11
Ax11x3
A:X
B:Y:A:X
Ax11x4
A:X:B:Y
Bx22
Bx22x3
B:Y:A:X
A:X:B:Y
A. Pretschner: Usage Control. Bertinoro 2014


Technische Universität München
A:X
Ax11x3
PID 11
IP A
Ax11x4
A:X:B:Y
B:Y:A:X
Bx22x3
PID 22
read( )
IP B
Ax11
Ax11x3
A:X
B:Y:A:X
Ax11x4
A:X:B:Y
Bx22
Bx22x3
B:Y:A:X
A:X:B:Y
A. Pretschner: Usage Control. Bertinoro 2014


Technische Universität München
A:X
Ax11x3
PID 11
IP A
Ax11x4
A:X:B:Y
B:Y:A:X
Bx22x3
PID 22
read( )
IP B
Ax11
Ax11x3
A:X
B:Y:A:X
Ax11x4
A:X:B:Y
Bx22
Bx22x3
B:Y:A:X
A:X:B:Y
A. Pretschner: Usage Control. Bertinoro 2014


Technische Universität München
+
Ax11
Bx22
Ax11x3
A:X
Bx22x3
B:Y:A:X
Ax11x4
A:X:B:Y
A. Pretschner: Usage Control. Bertinoro 2014

Performance



(ftp: overhead by one control and one data channel)

Performance

number of syscalls	FTP Server: vsftpd 100KB	4MB	128MB	HTTP Server: Apache 100KB	4MB	128MB
write()	43	139	4107	22	996	32740
total	148	358	8294	52	1523	49139
performance overhead factor w.r.t. native execution						
best case	11.66	0.61	0.14	7.29	0.53	0.25
worst case	13.15	1.16	0.65	9.01	4.90	4.65

Video: Distributed Data-Centric UC

- Video 5: usage control via ftp
  http://www22.in.tum.de/fileadmin/demos/uc/FTPDemo.avi

Agenda

- Part I: Introduction
- Part II: Event-Based Usage Control
- Part III: Data-Centric Usage Control
- Part IV: Quantitative Usage Control
- Part V: Local Single-Layer Enforcement
- Part VI: Distributed Enforcement
- **Part VII: Cross-Layer Enforcement**
- Part VIII: Policy Derivation
- Part IX: Discussion

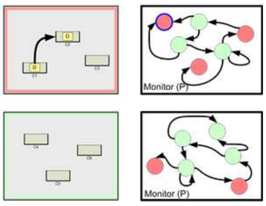
Different levels of abstraction (not stacked!)

Business Process
User application
Application wrapper
Service wrapper
Infrastructure Application
Runtime System
Operating System
CPU/VM (ring -1)

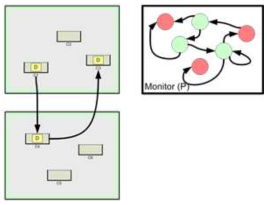
Why multiple layers?

- Alternative: tracking at level of machine code
- Loss of semantic information: how to determine „screenshot“ at level of machine code?
  - Precision
- Performance

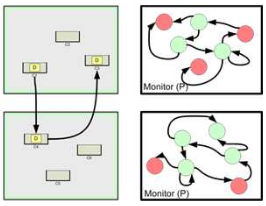
Problem description



Problem description

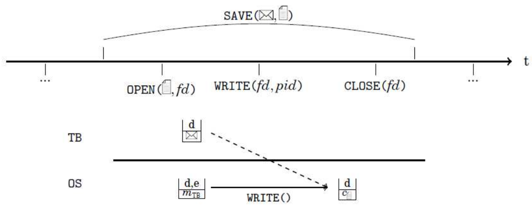


Problem description

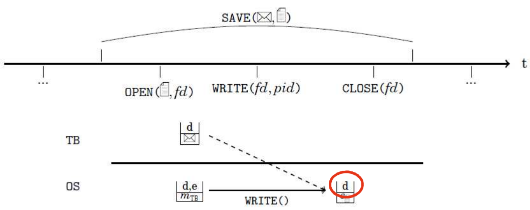


How can we track flows of data in-between representations across different l.o.a.?

Thunderbird example: save email



Thunderbird example: save email

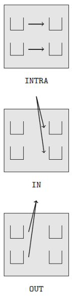


considering OS only would add e as well!

Five ingredients

- To define cross-layer flows: **Three kinds of behaviors**
- Scope
- Intermediate containers
- Cross-layer aliases
- Cross-layer transition relations

Behaviors $BEHAV = \{\mathtt{IN}, \mathtt{OUT}, \mathtt{INTRA}\}$



Container receives data from container at same level

Container receives data from container at different level

Container sends data to container at different level

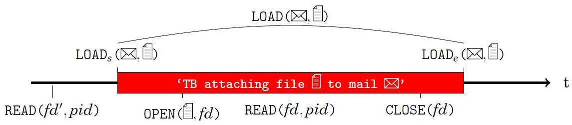
Cross-layer flow: A pair of causally related IN and OUT events

Five ingredients

- Three kinds of behaviors
- **Scope**
- Intermediate containers
- Cross-layer aliases
- Cross-layer transition relations

Scopes I

Events have duration. Scopes capture start, events at other layers, end.

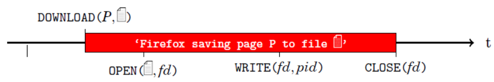


$SCOPE = \{sc \mid sc \text{ is a label for a cross-layer flow}\}$
$\Sigma = \Sigma_A \times \Sigma_B \times \mathbb{P}(SCOPE)$

Scopes II

Possibly multiple scopes at the same time.
Yet, one event belongs to exactly one scope:
$X_{BEHAV} : (\Sigma \times \mathcal{E}) \to (BEHAV \times SCOPE)$

Events may activate or deactivate scopes:
$X_{DELIM} : (\Sigma \times \mathcal{E}) \to \mathbb{P}(\{\mathtt{OPEN}, \mathtt{CLOSE}\} \times SCOPE)$

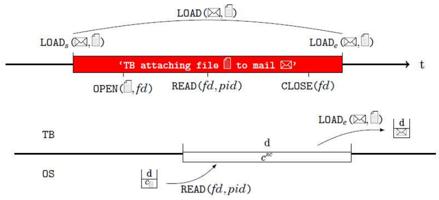


Example of scope activated by an event at one layer (DOWNLOAD(P,)) and deactivated by an event at the other layer (CLOSE(fd))

Five ingredients

- Three kinds of behaviors
- Scope
- **Intermediate containers:** the pipe between two layers
- Cross-layer aliases
- Cross-layer transition relations

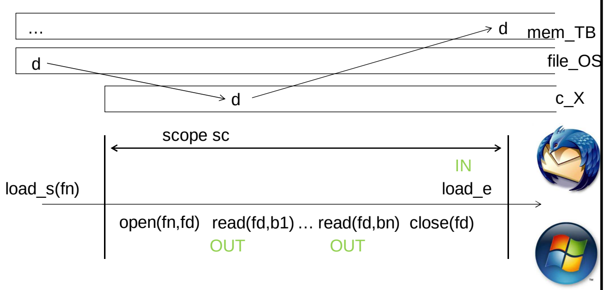
Intermediate Containers



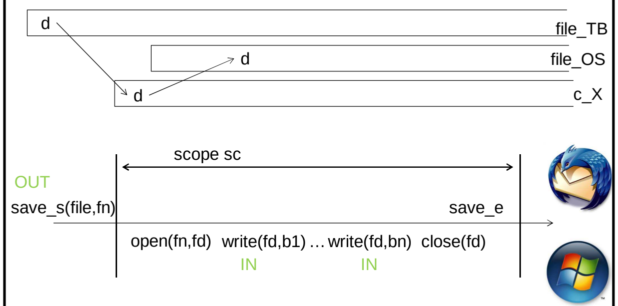
Graphic representation of an intermediate container for the scope sc = 'TB attaching file to mail'. In this example, c^{sc} stores data d read by the outgoing READ() system call at layer OS until it is read by the incoming LOAD$_e$() event in TB.

$\mathcal{C} = \mathcal{C}_A \cup \mathcal{C}_B \cup \mathcal{C}_X$
$s_x : \mathcal{C}_X \to \mathbb{P}(\mathcal{D})$
$\Sigma = \Sigma_A \times \Sigma_B \times SCOPE \times (\mathcal{C}_X \to \mathbb{P}(\mathcal{D}))$

Example: TB+OS – read



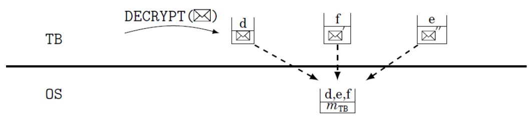
Example: TB+OS – write



Five ingredients

- Three kinds of behaviors
- Scope
- Intermediate containers
- **Cross-layer aliases**
- Cross-layer transition relations

Cross-Layer Aliases



Cross-layer alias example. In correspondence of event DECRYPT(✉) at TB layer a new data d appears in the container ✉, but no corresponding event takes place at the OS layer. Nevertheless, data d is propagated via cross-layer alias to the container that represents the memory of the Thunderbird process at the OS layer m_{TB}.

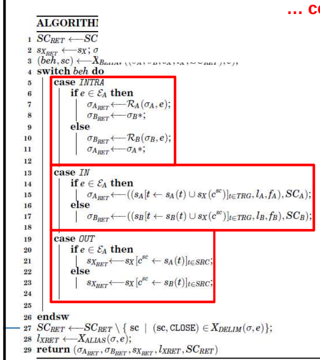
$$l_X : (\mathcal{C}_A \rightarrow \mathbb{P}(\mathcal{C}_B)) \cup (\mathcal{C}_B \rightarrow \mathbb{P}(\mathcal{C}_A))$$
$$\Sigma = \Sigma_A \times \Sigma_B \times SCOPE \times (\mathcal{C}_X \rightarrow \mathbb{P}(\mathcal{D})) \times \underline{((\mathcal{C}_A \rightarrow \mathbb{P}(\mathcal{C}_B)) \cup (\mathcal{C}_B \rightarrow \mathbb{P}(\mathcal{C}_A)))}$$
$$X_{ALIAS} : (\Sigma \times \mathcal{E}) \rightarrow ((\mathcal{C}_A \rightarrow \mathbb{P}(\mathcal{C}_B) \cup (\mathcal{C}_B \rightarrow \mathbb{P}(\mathcal{C}_A))$$

Five ingredients

- Three kinds of behaviors
- Scope
- Intermediate containers
- Cross-layer aliases
- **Cross-layer transition relations**

Cross-layer transition relation

… copy data to intermediate container



$$\sigma_A = ((s_A, l_A, f_A), SC_A)$$
$$\sigma_B = ((s_B, l_B, f_B), SC_B)$$
$$s_B* = s_B[c \leftarrow s_{A_{RET}}(t)]_{t \in TRG, c \in l_X(t)}$$
$$s_A* = s_A[c \leftarrow s_{B_{RET}}(t)]_{t \in TRG, c \in l_X(t)}$$

Example Thunderbird+OS

ALGORITHM 2: $X_{DELIM}(e, \sigma)$

```
for each (✉, 🗋) in Thunderbird do
    if e = SAVE_s (✉, 🗋) then
        return {(OPEN, 'TB saving mail ✉ to file 🗋')};
    else if e = SAVE_e (✉, 🗋) then
        return {(CLOSE, 'TB saving mail ✉ to file 🗋')};
    else if e = LOAD_s (✉, 🗋) then
        return {(OPEN, 'TB attaching file 🗋 to mail ✉')};
    else if e = LOAD_e (✉, 🗋) then
        return {(CLOSE, 'TB attaching file 🗋 to mail ✉')};

end
return ∅
```


Example TB+OS

ALGORITHM 3: $X_{BEHAV}(\sigma, e)$

```
if e = WRITE(fd, pid) then
    for each (✉, 🗋) in Thunderbird do
        Scope sc ⟵ 'TB saving mail ✉ to file 🗋';
        if sc ∈ SCOPE then
            if (pid = ⟨process id of Thunderbird⟩ ) ∧ (σ.f(🗋) = σ.f(fd, pid)) then
                return (IN, sc)
            end
        end
    end
    return (INTRA, ∅)
else if e = READ(fd, pid) then
    for each (✉, 🗋) in Thunderbird do
        Scope sc ⟵ 'TB attaching file 🗋 to mail ✉';
        if sc ∈ SCOPE then
            if (pid = ⟨process id of Thunderbird⟩ ) ∧ (σ.f(🗋) = σ.f(fd, pid)) then
                return (OUT, sc)
            end
        end
    end
    return (INTRA, ∅)
else if e = SAVE_s (✉, 🗋) then
    return (OUT, 'TB saving mail ✉ to file 🗋');
else if e = LOAD_e (✉, 🗋) then
    return (IN, 'TB attaching file 🗋 to mail ✉');
else
 return (INTRA, ∅)
```

cross-layer flows:
one container identified
by names at two layers

Not trivial:

- Identification of start and stop events

Video: Cross-Layer Data-Centric UC

- Video 6: distributed data-driven usage control for smart meter readings rendered in a social network
  http://www22.in.tum.de/fileadmin/demos/uc/pec_uc4win6_internet.mp4
- Video 7: distributed data-driven usage control in a social network application
  http://www22.in.tum.de/fileadmin/demos/uc/final-thund-cloud.htm

TIME …

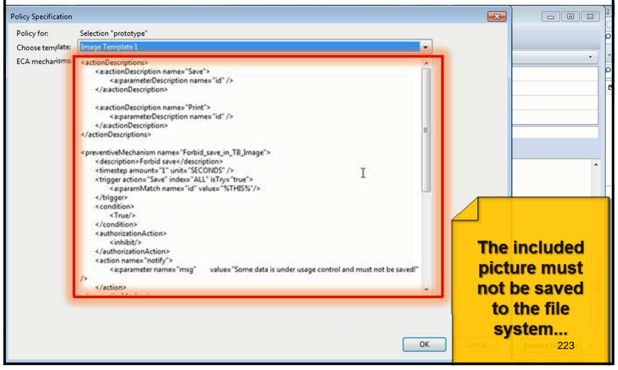
…for policy derivation?

Agenda

- Part I: Introduction
- Part II: Event-Based Usage Control
- Part III: Data-Centric Usage Control
- Part IV: Quantitative Usage Control
- Part V: Local Single-Layer Enforcement
- Part VI: Distributed Enforcement
- Part VII: Cross-Layer Enforcement
- **Part VIII: Policy Derivation (sketch)**
  slides by Prachi Kumari
- Part IX: Discussion

Technische Universität München

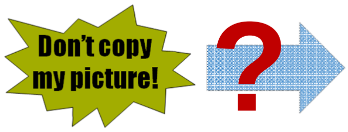
A Motivating Use Case



223

Technische Universität München

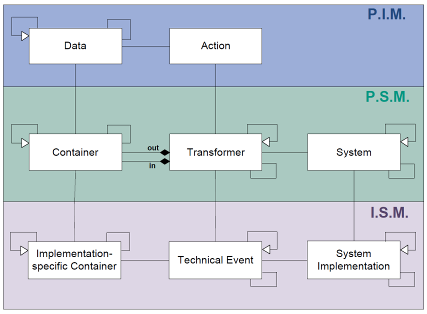
Problem




```
<! -- For Firefox Web Browser -->
<controlMechanism>
 <id>Browser_CopyPaste</id>
 <triggerEvent>
  <id>copy</id>
  <parameter name="obj" value="img_profile" type="dataUsage"/>
  <parameter name="isTry" value="true"/>
 </triggerEvent>
 <condition><true/></condition>
 <actions><inhibit/></actions>
</controlMechanism>
<! -- For X11 Windowing System -->
<controlMechanism>
 <id>X11_Screenshot</id>
 <triggerEvent>
  <id>GetImage</id >
  <parameter name="obj" value="0x1a00005" type ="dataUsage"/>
  <parameter name="isTry" value="true"/>
 </triggerEvent>
 <condition><true/></condition>
 <actions>
  <allow>
   <modify>
    <parameter name="planeMask" value="0x0"/>
   </modify>
  </allow>
 </actions>
</controlMechanism>
<! -- For Windows Operating System -->
<controlMechanism>
 <id>OS_Restrict_File_Usage</id>
 <triggerEvent>
  <id>open</id>
  <parameter name="obj" value="cacheFile" type ="dataUsage"/>
  <parameter name="isTry" value="true"/>
 </triggerEvent>
 <condition>
  <XPathEval>
   /triggerEvent/parameter[@name='PNAME']
   /@value!='c:\\Firefox\\firefox.exe'
  </XPathEval>
 </condition>
 <actions><inhibit/></actions>
</controlMechanism>
```

Technische Universität München

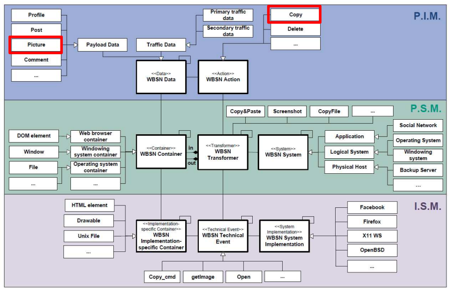
Domain Meta Model



A. Pretschner: Usage Control. Bertinoro 2014 225

Technische Universität München

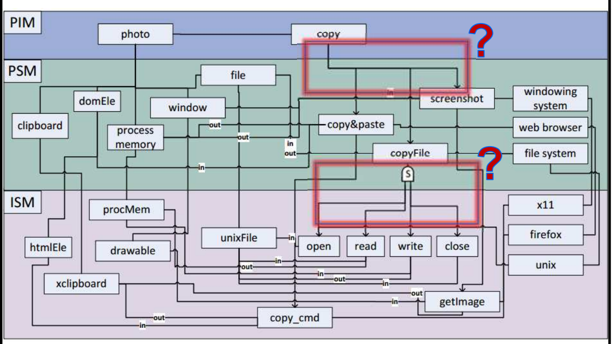
WBSN Domain Model



A. Pretschner: Usage Control. Bertinoro 2014 226

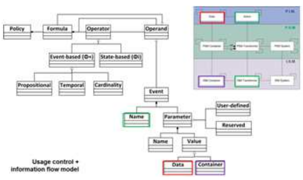
Technische Universität München

Excerpt



Technische Universität München

Usage Control + Domain meta -models



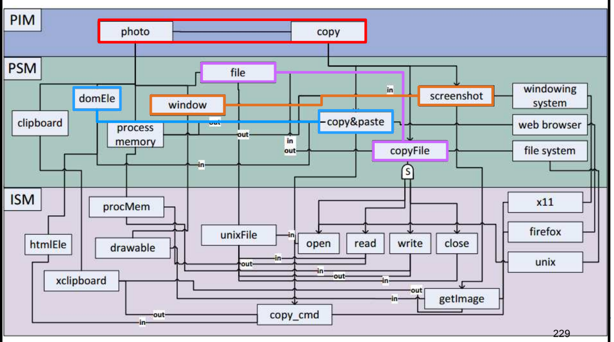
Combine the two models.

Define an action in terms of:

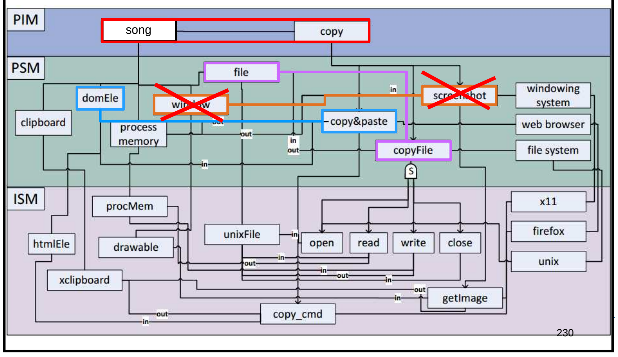
1. Sets/sequences of transformers
2. Resulting system state

228

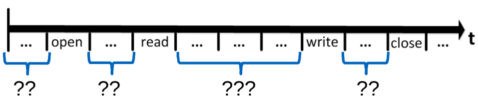
1- Refinement as disjunction



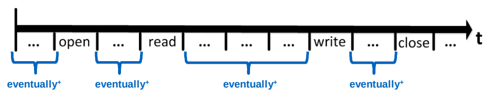
(Filter irrelevant transformers)



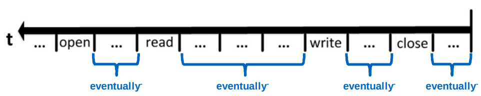
2- Refinement as sequence



2- Refinement as sequence

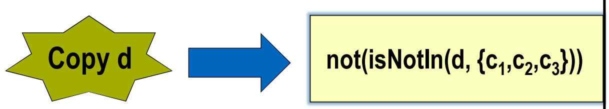


2- Refinement as sequence



3- Refinement using States

- Defining state formula encoded in definition of actions



Action Refinement using States

- Defining state formula encoded in definition of actions

Copy d → not(isNotIn(d, $\{c_1,c_2,c_3\}$))

??

Technische Universität München

Action Refinement using States

- Defining state formula encoded in definition of actions

Copy d → not(isNotIn(d, $\{c_1,c_2,c_3\}$))

??

Technische Universität München

Action Refinement using States

- Actual data and **containers** not known at declaration time
  - Use ISM/PSM containers (classes of containers) …

$$\Phi_i ::= \underline{isNotIn}(Data, \mathbb{P}\ Container) \mid \underline{isCombinedWith}(Data, Data) \mid \underline{isOnlyIn}(Data, \mathbb{P}\ Container)$$

$$\Downarrow$$

$$\Phi_i ::= isNotIn(Data, \mathbb{P}\ ISMContainer) \mid isNotIn(Data, \mathbb{P}\ PSMContainer) \mid isOnlyIn(Data, \mathbb{P}\ ISMContainer) \mid isOnlyIn(Data, \mathbb{P}\ PSMContainer) \mid isCombinedWith(Data, Data)$$

  - and decide at runtime which class an actual container belongs to

Technische Universität München

Action Refinement using States

- Actual **data** and containers not known at declaration time
  - Use ISM/PSM containers (classes of containers)
  - Use variables …

$$\Phi_i ::= isNotIn(Data, \mathbb{P}\ ISMContainer) \mid isNotIn(Data, \mathbb{P}\ PSMContainer) \mid isOnlyIn(Data, \mathbb{P}\ ISMContainer) \mid isOnlyIn(Data, \mathbb{P}\ PSMContainer) \mid isCombinedWith(Data, Data)$$

$$\Downarrow$$

$$Var ::= V(\mathbb{N}_1)$$
$$VarData == Var \cup Data$$

$$\Phi_{iv} ::= isNotIn(VarData, \mathbb{P}\ ISMContainer) \mid isNotIn(VarData, \mathbb{P}\ PSMContainer) \mid isOnlyIn(VarData, \mathbb{P}\ ISMContainer) \mid isOnlyIn(VarData, \mathbb{P}\ PSMContainer) \mid isCombinedWith(VarData, Data)$$

  - and bind these variables to actual data at deployment time

Technische Universität München

Action Refinement using States

Copy d → not(isNotIn(d, $\{c_1,c_2,c_3\}$))

Technische Universität München

Action Refinement using States

So, we'll check *at runtime* if a container is of class Clipboard

Copy d → not(isNotIn(d, Clipboard))

Action Refinement using States

Copy photo

not(isNotIn(d, Clipboard))

Action Refinement using States

… and we'll bind a concrete data item to the variable d *at deployment time*

Copy photo

not(isNotIn(photo, Clipboard))

Combining event and state –based refinements

- Complete action refinement is
  - A disjunction over the event- and state- based refinements

$$
\begin{aligned}
&\forall t \in Trace;\ n \in \mathbb{N};\ \sigma \in \Sigma \bullet \sigma = states(t,n) \wedge \\
&\forall d \in Data;\ a \in Action;\ \psi \in \Phi^-;\ \varphi \in \Phi_i^- \bullet \tau_{action}(\psi,\sigma) = \varphi \Leftrightarrow \\
&\psi \in \{true, false\} \wedge (\varphi = \psi) \\
&\boxed{\begin{aligned}&\vee\ \psi = E(a) \wedge (\varphi = \underline{or}^-(\tau_{state}(E(a)), \tau_{ev}(E(a),\sigma))) \\ &\vee\ \psi = T(a) \wedge (\varphi = \underline{or}^-(\tau_{state}(T(a)), \tau_{ev}(T(a),\sigma)))\end{aligned}} \\
&\vee\ \exists \chi \in \Phi^- \bullet \psi \in \{\underline{not}(\chi), \underline{not}^-(\chi)\} \wedge (\varphi = \underline{not}^-(\tau_{action}(\chi,\sigma))) \\
&\vee\ \exists \chi, \xi \in \Phi^- \bullet \psi \in \{\underline{or}(\chi,\xi), \underline{or}^-(\chi,\xi)\} \wedge (\varphi = \underline{or}^-(\tau_{action}(\chi,\sigma), \tau_{action}(\xi,\sigma))) \\
&\vee\ \exists \chi, \xi \in \Phi^- \bullet \psi \in \{\underline{and}(\chi,\xi), \underline{and}^-(\chi,\xi)\} \wedge (\varphi = \underline{and}^-(\tau_{action}(\chi,\sigma), \tau_{action}(\xi,\sigma))) \\
&\vee\ \exists \chi, \xi \in \Phi^- \bullet \psi \in \{\underline{implies}(\chi,\xi), \underline{implies}^-(\chi,\xi)\} \wedge (\varphi = \underline{implies}^-(\tau_{action}(\chi,\sigma), \tau_{action}(\xi,\sigma))) \\
&\vee\ \exists \chi, \xi \in \Phi^- \bullet \psi = \underline{since}^-(\chi,\xi) \wedge (\varphi = \underline{since}^-(\tau_{action}(\chi,\sigma), \tau_{action}(\xi,\sigma))) \\
&\vee\ \exists i \in \mathbb{N};\ \chi \in \Phi^- \bullet \psi = \underline{before}^-(i,\chi) \wedge (\varphi = \underline{before}^-(i, \tau_{action}(\chi,\sigma))) \\
&\vee\ \exists \chi \in \Phi^- \bullet \psi = \underline{always}^-(\chi) \wedge (\varphi = \underline{always}^-(\tau_{action}(\chi,\sigma))) \\
&\vee\ \exists i \in \mathbb{N};\ \chi \in \Phi^- \bullet \psi = \underline{within}^-(i,\chi) \wedge (\varphi = \underline{within}^-(i, \tau_{action}(\chi,\sigma))) \\
&\vee\ \exists i \in \mathbb{N};\ \chi \in \Phi^- \bullet \psi = \underline{during}^-(i,\chi) \wedge (\varphi = \underline{during}^-(i, \tau_{action}(\chi,\sigma))) \\
&\vee\ \exists i \in \mathbb{N};\ \chi \in \Phi^- \bullet \psi = \underline{repmax}^-(i,\chi) \wedge (\varphi = \underline{repmax}^-(i, \tau_{action}(\chi,\sigma))) \\
&\vee\ \exists l, x, y \in \mathbb{N};\ \chi \in \Phi^- \bullet \psi = \underline{replim}^-(l,x,y,\chi) \wedge (\varphi = \underline{replim}^-(l,x,y, \tau_{action}(\chi,\sigma))) \\
&\vee\ \exists i \in \mathbb{N};\ \chi, \xi \in \Phi^- \bullet \psi = \underline{repsince}^-(i,\chi,\xi) \wedge (\varphi = \underline{repsince}^-(i, \tau_{action}(\chi,\sigma), \tau_{action}(\xi,\sigma)))
\end{aligned}
$$

Policy derivation: from SLPs to ILPs

- Once the domain models are defined, we need to transform future-time SLPs to past-time ILPs
- No constructive methods known
- Use templates in graphical editor

Disjunctions and sequences of events

- Events may be refined into disjunction of events at more than one layer of abstraction
  - „Copy" becomes „save email" at Thunderbird level <u>and</u> „open() write()* close" at OS level
- Two (plus a third, later!) possible strategies
  - Deploy only one policy that captures all layers simultaneously
  - Project policy to layers of abstraction and deploy each projection

Disjunctions and sequences of events II

- Either way, a policy may be evaluated and enforced multiple times at different layers!
  - „Don't copy more than twice": one physical „copy" action captured by
    - one „save" event at TB layer
    - one „write" event at OS level
- Sometimes no problem: <u>always(not</u> copy) can be simultaneously enforced at several layers

Disjunctions and sequences of events III

- Sometimes problematic:
  - Executors at multiple layers lead to duplicate effects, e.g., always(copy implies notify)
  - Critical: Counting and „stateful“ temporal formulas (repmax, until; minimized monitor has more than one state)
  - In these cases, pick one layer and ignore the other one
  - Possible unless a policy correlates events at different layers, e.g., „no screenshot until mail client disabled“

Related: Distributed Systems

- „Distribute at most n times“
  - Local enforcement: each node can do n distributions, overall number not restricted
  - Global enforcement: overall number restricted to n
- Here, in contrast, we want to make sure that events in multiple systems (~ at multiple layers) are counted each
- Requires shared data state

Agenda

- Part I: Introduction
- Part II: Event-Based Usage Control
- Part III: Data-Centric Usage Control
- Part IV: Quantitative Usage Control
- Part V: Local Single-Layer Enforcement
- Part VI: Distributed Enforcement
- Part VII: Cross-Layer Enforcement
- Part VIII: Policy Derivation
- **Part IX: Discussion**

Assumptions

- Layer-specific events interceptable
- Omnipresence of UC infrastructures
- Integrity and authenticity of policies ensured
- End-to-end confidentiality of data ensured
- UC infrastructure and underlying technology secure

Attacker Models and Guarantees

- Assets, goals, detective vs. preventive enforcement
- Benevolent and malevolent regular users and administrators; external intruders
- Malware

- Guarantees layer-specific and depending on attacker

Protecting the infrastructure

- BonaFides system: record logs of file changes (iNotify) and protect them using TPM
  - Detective not preventive

Technische Universität München

Insecure Security

- Yes solution can be circumvented
- Like any other security solution: a matter of commitment and resources
- Security as a sub-discipline of risk management!

- How much technology will be in a „security solution“?

A. Pretschner: Usage Control. Bertinoro 2014 253

Technische Universität München

Concerns

- Overapproximations
  - Quantitative measurements (but what does this mean?)
  - Declassification
- Performance
- (Formal) guarantees
- Collection of usage information in itself likely to impact privacy
- Business model - omnipresence of UC infrastructures?

- DEPENDS ON SPECIFIC APPLICATION SCENARIO!

A. Pretschner: Usage Control. Bertinoro 2014 254

Technische Universität München

Related Work

- Access control
- Usage control
- Possibilistic information flow control
- Quantitative information flow measurements
- Complex event processing
- DRM
- Data loss prevention
- Android security
- Runtime verification
- Enforcement automata
- Model-based development
- Semantics of sequence charts

A. Pretschner: Usage Control. Bertinoro 2014 255

Technische Universität München

Wrap-Up

- Part I: Introduction
- Part II: Event-Based Usage Control
- Part III: Data-Centric Usage Control
- Part IV: Quantitative Usage Control
- Part V: Local Single-Layer Enforcement
- Part VI: Distributed Enforcement
- Part VII: Cross-Layer Enforcement
- Part VIII: Policy Derivation
- Part IX: Discussion

Technische Universität München

Is this a good idea?

A. Pretschner: Usage Control. Bertinoro 2014 257