AUSNAHMEN BEI ZUGRIFFSKONTROLLE

JA, NEIN, VIELLEICHT?

Helmut Petritsch

MOTIVATION

AGILE SOFTWAREENTWICKLUNG

Anforderungen im Zuge der technischen Umsetzung verfeinern

Iterativ

ZUGRIFFSKONTROLLE

Nicht-funktionale Anforderung "stört" beim Entwickeln

Abstimmung mit Experten, Dokumentation und Nachvollziehbarkeit tendieren zum Wasserfallmodell

Inhärente Annahme:

Man kann immer PERMIT/DENY-Entscheidung treffen.

⇒ Agilität? Iterativ?

HERAUSFORDERUNGEN

Komplexe Berechtigungen erfassen, verstehen, anpassen, monitoren

Berechtigungen als maschinenlesbares Regelwerk

⇒ Auslegung von Gesetztestexten?

⇒ Teil der Software? Konfiguration?

SCHWIERIGKEITEN IN DER IT

System kann seine Umwelt nicht vollständig verstehen

Regeln und deren Ausnahmen sind situationsabhängig,
Vorhersage aller Situationen aufwändig / nicht möglich

⇒ Perfekte Zugriffskontrolle kaum möglich

BEISPIEL

KRANKENHAUS-INFORMATIONSYSTEM

Pflegepersonal hat Zugriff auf Verwaltungsdaten

Ärtze haben Zugriff auf medizinische Daten

Zusätzlicher Schutz:

Zugriff nur bei Krankenhaus-Aufenthalt des Patienten

DENKBARE AUSNAHMEN

- Pfleger handelt auf Anweisung eines Arztes
- Notfallpatient
- Benutzer vergisst Passwort
- berechtigter Mitarbeiter ist abwesend
- ...

KONSEQUENZEN

ZU STRENG

Mitarbeiter arbeiten...

... um Regeln herum (geteilte Passwörter)

... um IT(-Systeme) herum
(Daten werden kopiert, gedruckt, ...)

⇒ Sicherheit und Nachvollziehbarkeit leiden

ZU SCHWACH

Sicherheitsziele werden nicht erreicht

BREAK-GLASS

BREAK-GLASS

In unklaren Situationen: Benutzer entscheidet

Mögliches Überschreiten der Kompetenz muss verantwortet werden

Idee: Einschlagen einer Scheibe gibt Zugang zu Ressourcen, aber Übergriff ist nachvollziehbar

JA – Zugriff ist erlaubt

NEIN – Zugriff ist verboten

VIELLEICHT – Benutzer kann entscheiden

EINSCHRÄNKUNGEN

- Nicht jede Aktion kann und soll durch Ausnahme TODO gedeckt sein.
- Es muss klar nachvollziehbar sein, wer wann was gemacht hat.
- Missbrauch muss bestrafbar sein.

BESTANDTEILE

Modell: Definition von Regeln – was ist mit Ausnahme-Rechten erlaubt

Versionierung: System-Kontext muss für Nachbearbeitung festgehalten werden

Post-Access: Analyse von Übergriffen

MODELL

OBLIGATIONS

Vielleicht: zusätzliche Bedingungen, z.B.

- Benutzer muss Notfall bestätigen
- erweitertes Logging anstoßen

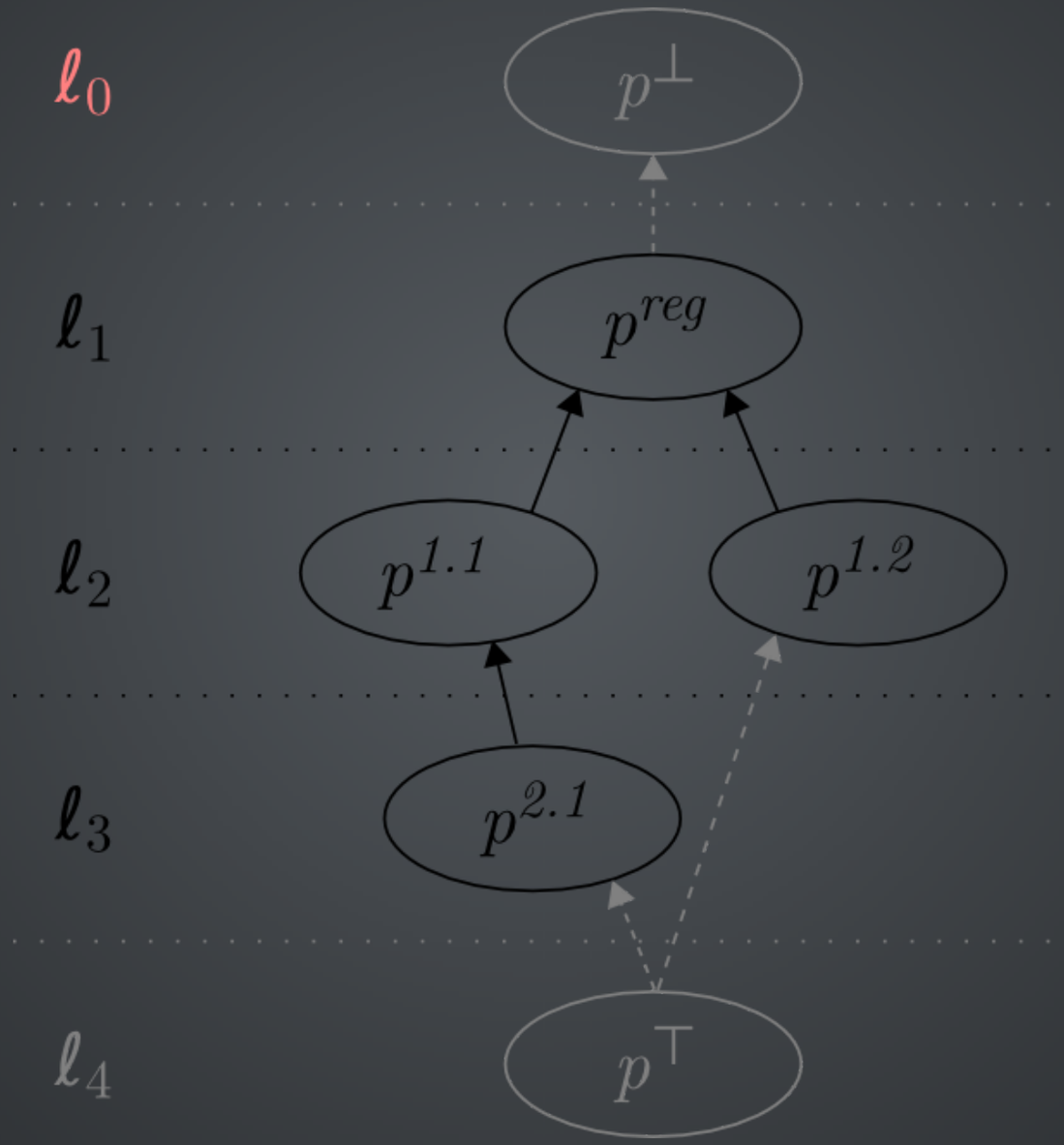
INHERITANCE (POSITIV)

PERMIT-Regeln

Notfall-Berchtigungen **erweitern** reguläre Berechtigungen

Präferiert reguläre über Notfall-Berechtigungen

LATTICE (VERBAND)

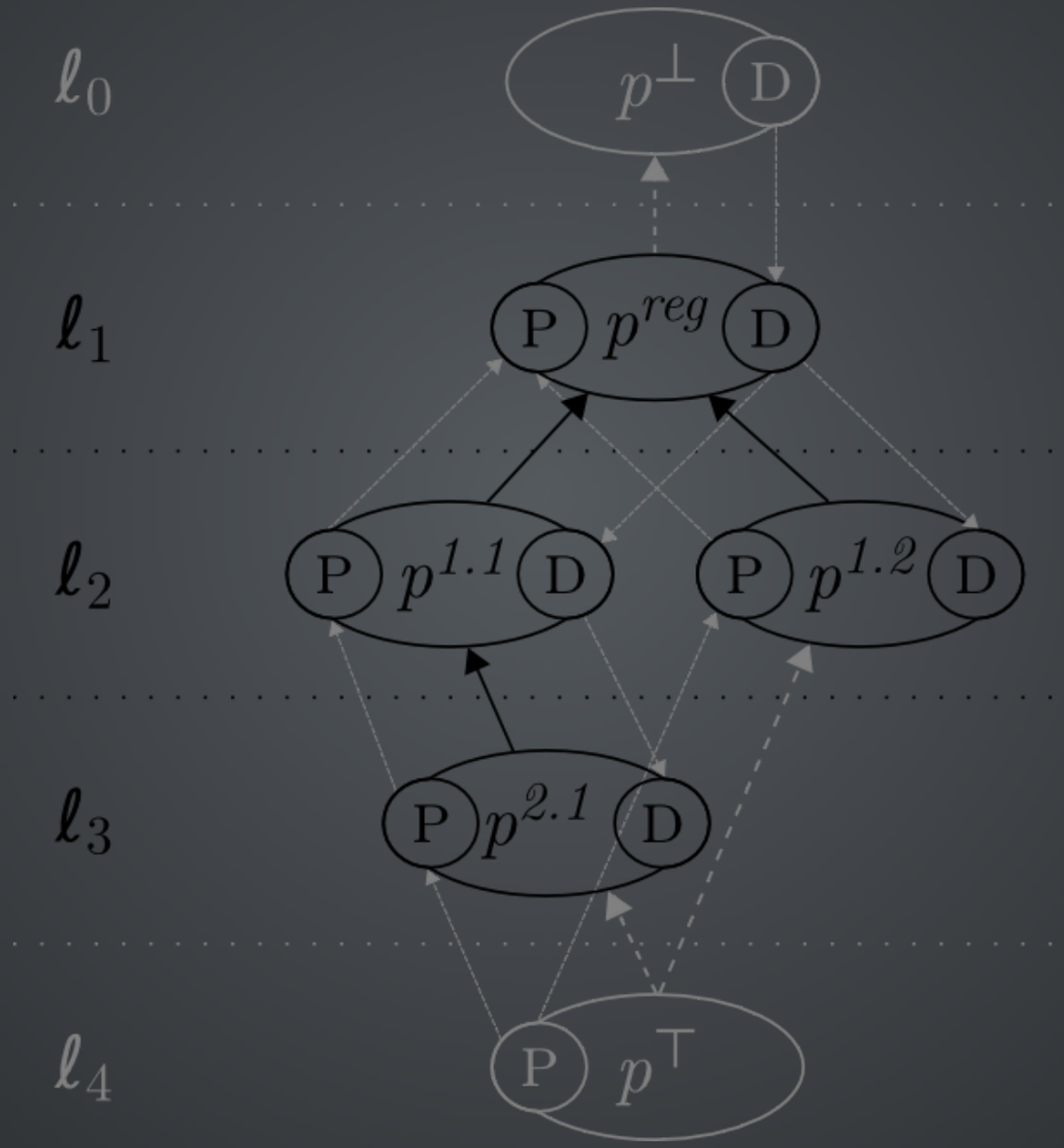


INHERITANCE (NEGATIV)

DENY-Regeln

Reguläre Verbote **erweitern** Notfall-Verbote

DENY-LATTICE



VERSIONIERUNG

VERSIONIERUNG

Vorhalten der für Zugriffskontrollentscheidung relevanten Parameter

Spätere Nachvollziehbarkeit der Entscheidung

IDEE

Alles, was eine Zugriffskontroll-Entscheidung beeinflussen kann, sollte selbst durch Zugriffskontrolle geschützt sein.

Versionierung von

- Regeln (policy permissions)
  z.B. in DSL/Text
- Security-Konfiguration (policy state)
  z.B. Key-Value Assignments

POST-ACCESS

ANALYSE

Nachvollziehbarkeit des Zugriffs

Debugging von Zugriffskontrolle

SEMI-AUTOMATISCHE ANALYSE

Analyse mit Regeln,
die Systemzustand nach Zugriff kennen

"Post-Delegation"

CONCLUSIO

WAS IST ANDERS?

Kontrolliertes "Aufweichen" für Notfälle erlaubt strengere Regeln für Normalfall

Iteratives Verstehen der gelebten Prozesse

Flexibilität bei neuen bzw.
sich neu etablierenden Prozessen

VORAUSSETZUNGEN

Klare API zum Zugriffskontrollsystem

Ausmodelliertes Zugriffskontrollmodell

Integration ins GUI

FRAGEN / DISKUSSION