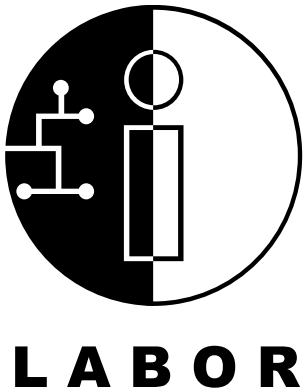
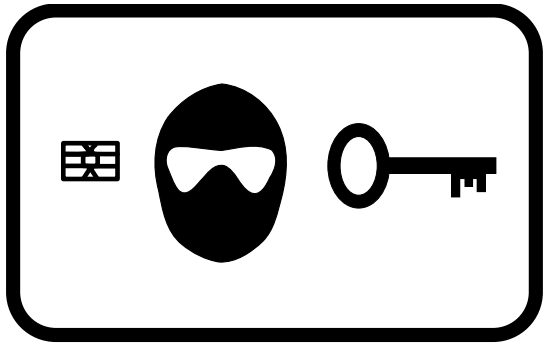
AnonAccess

Sören Heisrath Daniel Otte





27. Dezember 2007

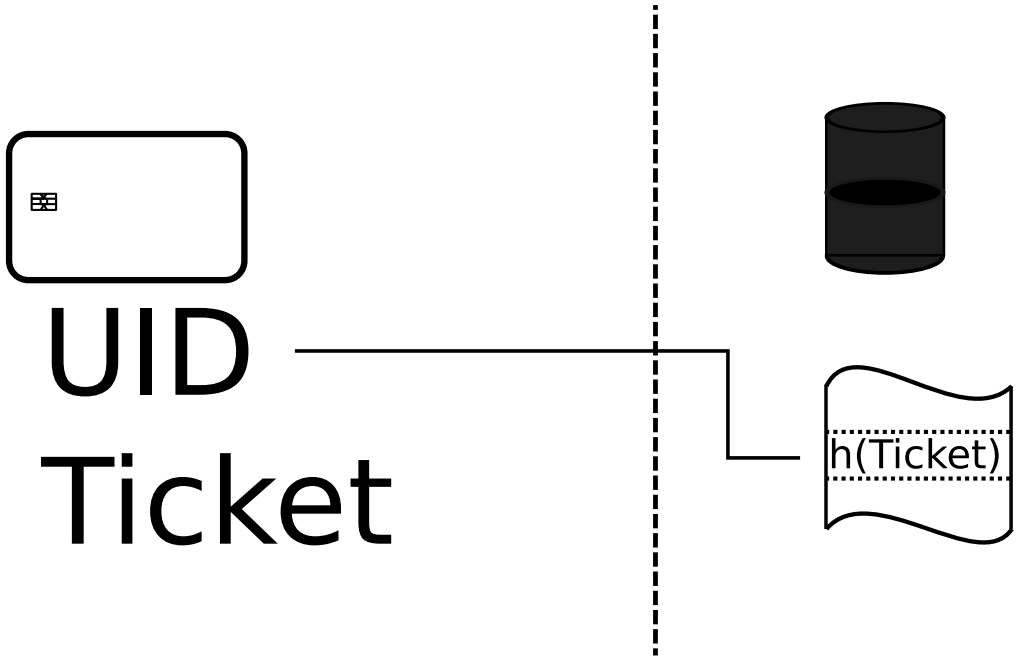
Unser Schlüsselproblem

- viele Leute
- wenig Schlüssel
- wenig Geld

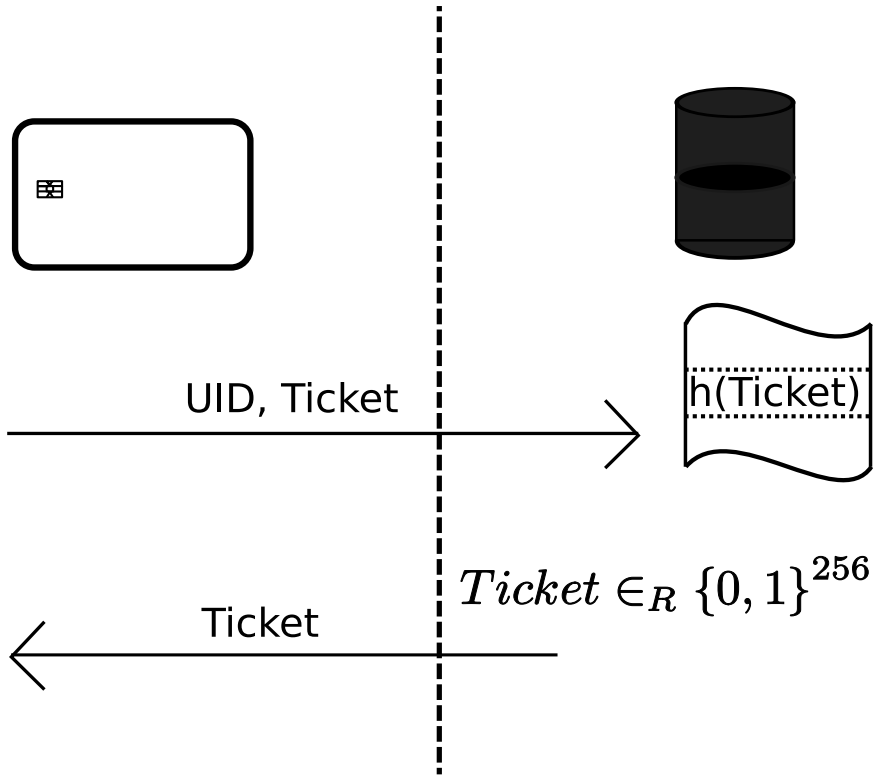
Unsere Anforderungen

- einfach zu realisieren
- günstig
- sicher

Ansatz 1



Ansatz 1 – Ablauf



Ticket-DB Struktur

Struktur eines Eintrags in der Ticket-DB:

flags	Flags (siehe unten)
nickname	Nickname (wenn Speicherung gewünscht)
ticketmac	MAC vom Ticket

Struktur der Flags:

exist	Benutzer existiert
admin	Benutzer hat Admin Status
locked	Benutzer ist gesperrt
notify_lostadmin	Benutzer hat Admin Status verloren
anonymous	Benutzer hat keinen Namen hinterlegt

Ansatz 1 – Probleme

Ansatz 1 – Probleme

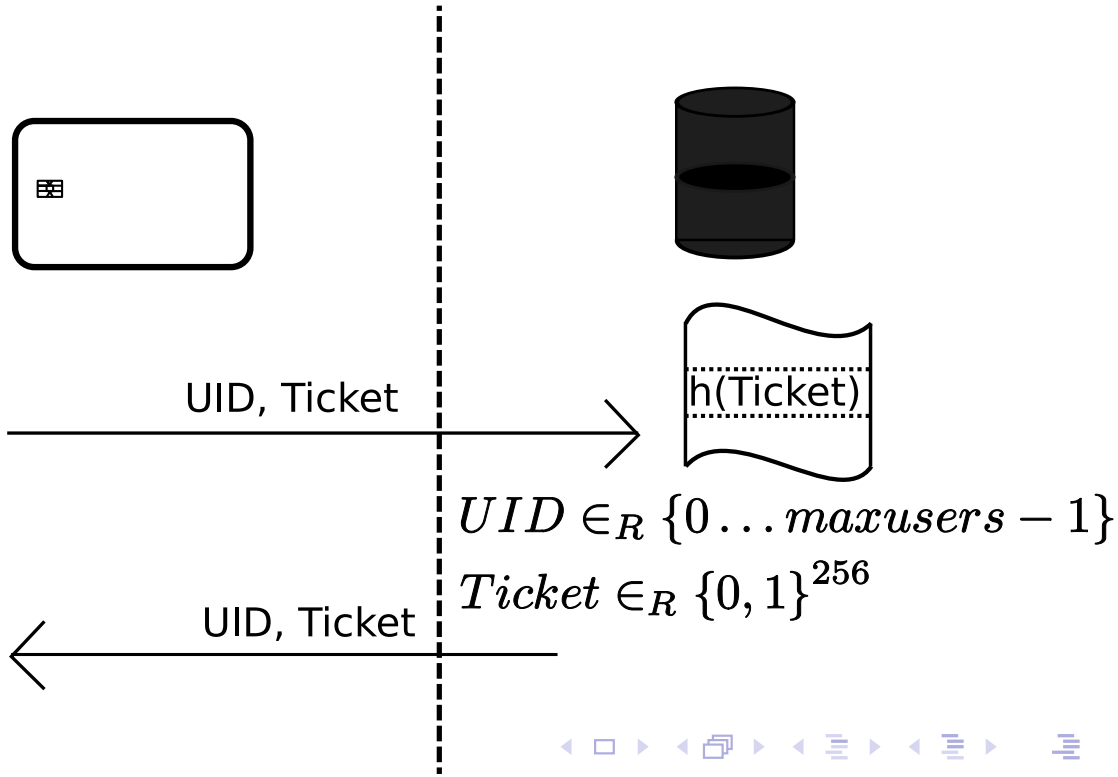
- nur pseudonym

Ansatz 1 – Probleme

- nur pseudonym
- pseudonym schwer merkbar

Ein zweiter Ansatz

dynamische UIDs



Ansatz 2 – Probleme

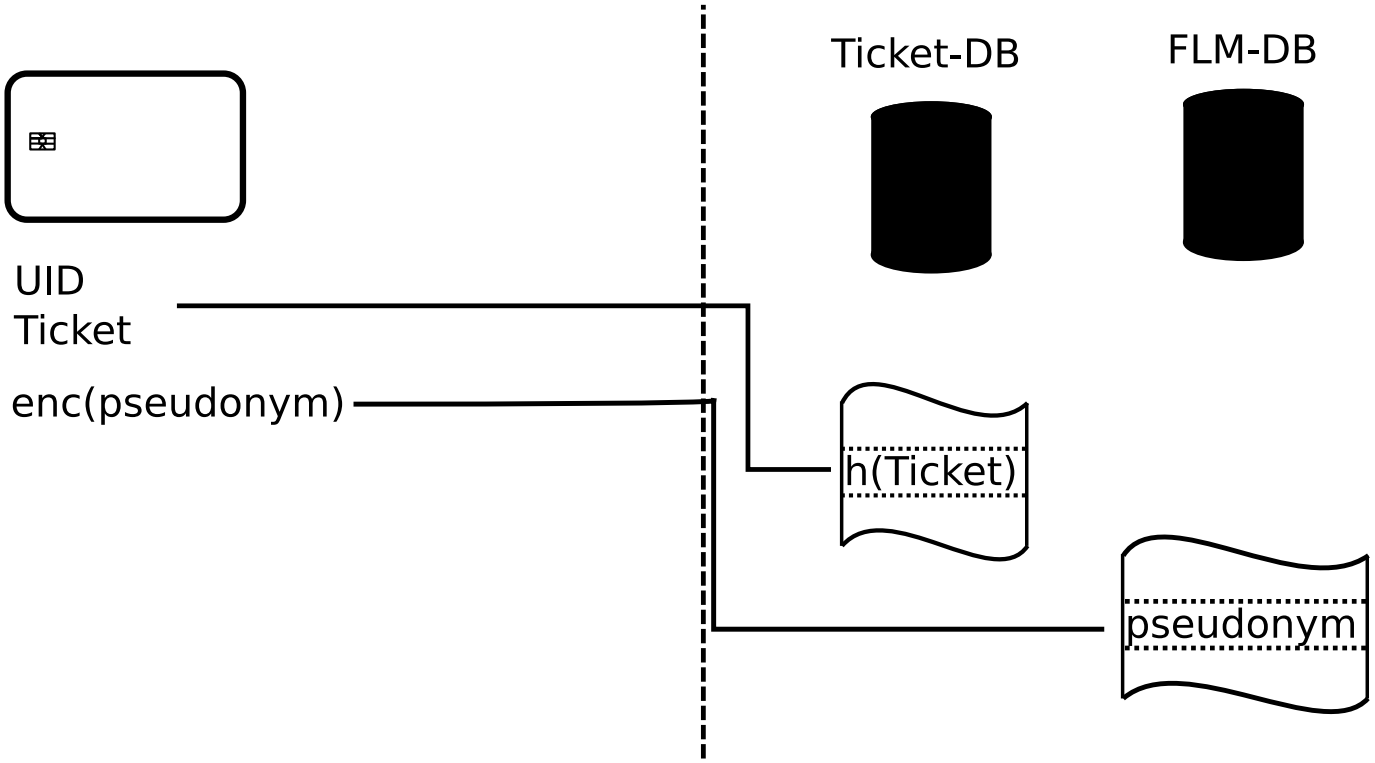
Ansatz 2 – Probleme

- keine feste Referenz auf Nutzer

Ansatz 2 – Probleme

- keine feste Referenz auf Nutzer
- nicht mehr ”wartbar”

Zusätzliche Datenbank für Änderungen



AuthBlock Struktur

UID	User ID; Index in die Ticket-DB
Ticket	$enc_{TicketKey}(24ByteZufall \parallel 8ByteZeitstempel)$
rkey	32 Byte zufälliger Schlüssel
rid	$enc_{ridKey}(enc_{rkey}(Pseudonym))$
HMAC	HMAC über die vorrangegangenen Daten

AuthBlock Struktur

UID	User ID; Index in die Ticket-DB
Ticket	$enc_{TicketKey}(24ByteZufall \parallel 8ByteZeitstempel)$
rkey	32 Byte zufälliger Schlüssel
rid	$enc_{ridKey}\left(enc_{rkey}\left(h\left(Pseudonym\right)\right)\right)$
HMAC	HMAC über die vorrangegangenen Daten

FLM-DB Struktur

active	ist dieser Eintrag aktiviert
permanent	soll der Eintrag nach anwendung glöscht werden
last	letzter Eintrag in der Datenbank
setflags	Flags die gesetzt werden sollen
clearflags	Flags die zu löschen sind
timestamp	Zeitstempel zur Erstellung des Eintrags
hnick	Hash des Pseudonyms

End Of Presentation