Valikrypt:

Validation von kryptographischen Sicherheitsprotokollen unter Verwendung formaler Analysemethoden

Werner Stephan, Georg Rock

Frankfurt, 29.09.2003

Source: Stephan

Übersicht

1. Übersicht über das Projekt Valikrypt
2. Formale Spezifikation von Protokollen
3. Analysetechniken
4. Einbettung in den Entwicklungsprozess
5. Zusammenfassung / Ausblick

Source: Stephan

Projektbeteiligte

VALIkrypt

Auftraggeber

Bundesamt für Sicherheit in der Informationstechnik

Konsortium

SIT Institut Sichere Telekooperation

DFKI

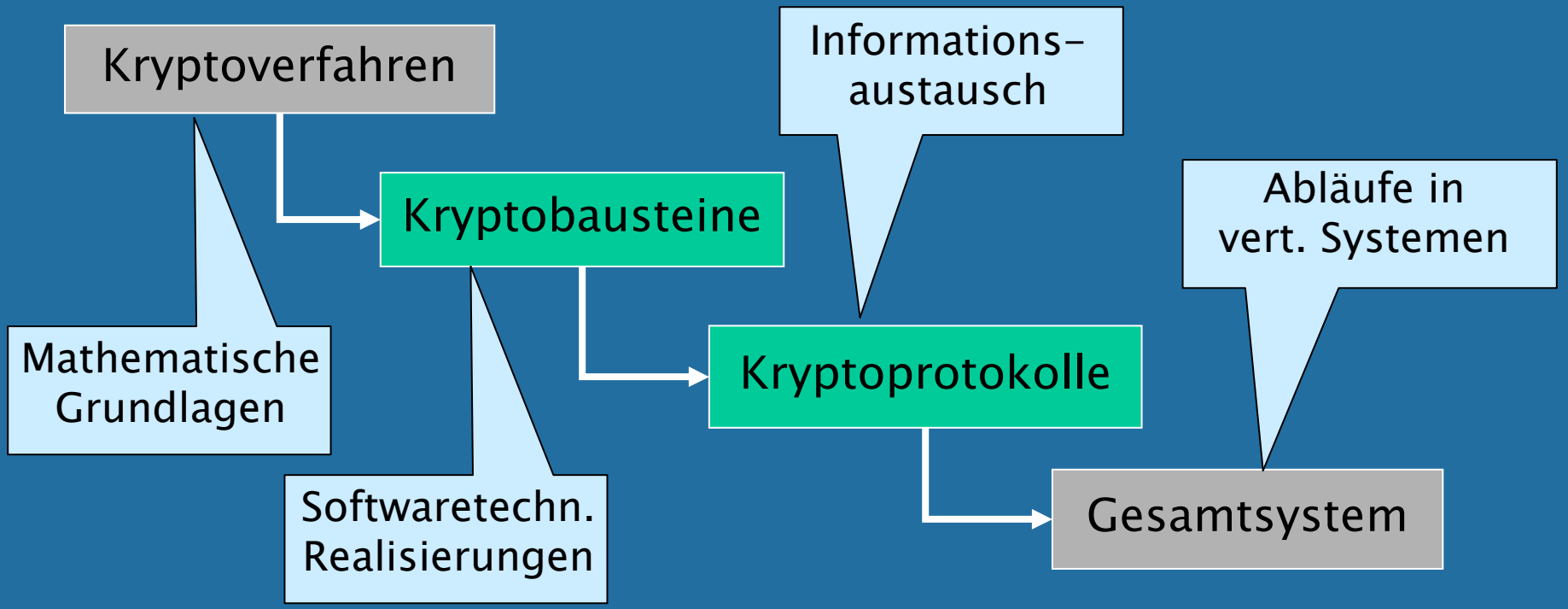
SIEMENS

Source: Stephan

DFKI

Kryptographische Techniken

Gegenstand: Protokolle mit kryptographischen Bausteinen und deren (Sicherheits-) Eigenschaften



Source: Stephan

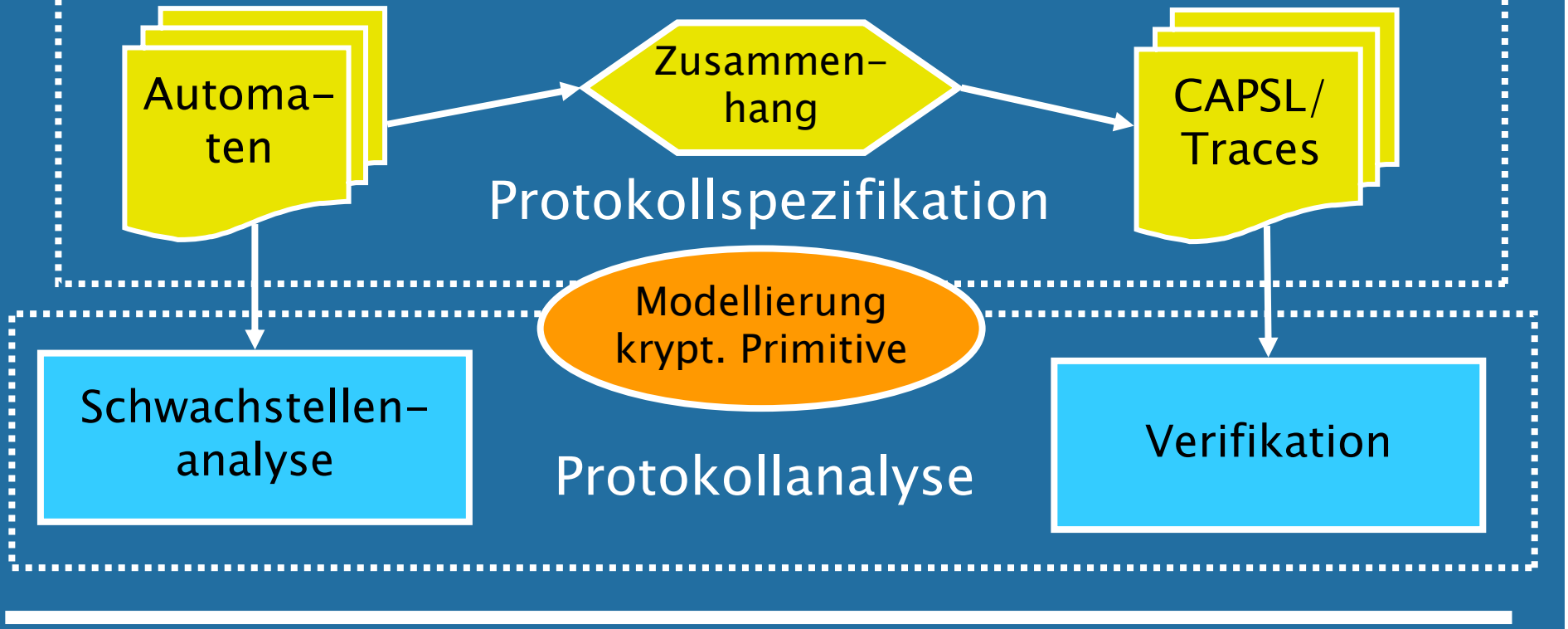
Aufgabenstellung

Entwicklungsumgebung für kryptographische Protokolle :

- Formalisierung von Protokollen
- Schwachstellenanlyse und (induktiver) Beweis als Analyseverfahren
- Unterstützung bei Auswahl und Parametrisierung von Kryptobausteinen

Source: Stephan

Gesamtansatz



Regelbasierte Auswahl u.
Parametrisierung von Bausteinen

Source: Stephan

Kryptographie: Bausteine

VALIkrypt

SIEMENS

- Umsetzung von mathematischen Verfahren in implementierte Routinen
  - Auswahl von Verfahren, Architektur (Bausteine)
  - Implementierung
  - Parametrisierung

Source: Stephan

DFKI

Kryptographie: Bausteine

- Regelbasierte Auswahl / Parametrisierung
  - Wissensbasis
  - Ergebnisorientiert: keine math. Beweise
  - Unterstützung von Nichtspezialisten (Krypto)
- Eigenschaften von Verfahren/Bausteinen
  - Einbringen in die Analyse

Source: Stephan

Informelle Protokollnotation

- Abfolge von Schritten bestimmter fester Bauart
- Standardform verbreitet in der Literatur
- Ausgangpunkt für CAPSL
- (zunächst) ohne Semantik
  - „Ausführungsmodell“ , Angreifer

1. $A \longrightarrow S: \quad A, B, R_A$
2. $S \longrightarrow A: \quad \{R_A, B, K_{AB}, \{K_{AB}, A\}_{K_{BS}}\}_{K_{AS}}$
3. $A \longrightarrow B: \quad \{K_{AB}, A\}_{K_{BS}}$
4. $B \longrightarrow A: \quad \{R_B\}_{K_{AB}}$
5. $A \longrightarrow B: \quad \{R_B - 1\}_{K_{AB}}$

Source: Stephan

CAPSL

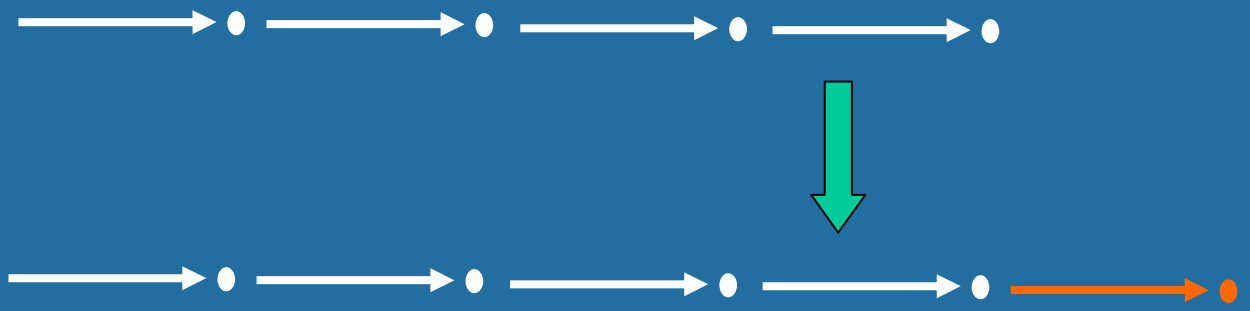
- Spezifikationssprache für Kryptoprotokolle
- zusätzliche Information
  - Anfangswissen
  - Datenstrukturen
  - Abarbeitung/Sichten
  - Eigenschaften
- formale Semantik
  - Übersetzung in tracebasierte Spezifikationen in VSE

```
PROTOCOL NSch_sym;
CONSTANTS
  S : Server;
VARIABLES
  A, B : Client;
  Na, Nb : Nonce, CRYPTO;
  Kab : Skey, CRYPTO, FRESH;
  F : Field;
ASSUMPTIONS
  HOLDS A: B;
  not(eqn(A,B));
MESSAGES
  1. A -> S: A, B, Na;
  2. S -> A: {Na, B, Kab,
                {Kab, A}sek(S,B,0)%F}
              (sek(S,A,0)%sek(A,0));
  3. A -> B: F%{Kab, A}sek(B,0);
  4. B -> A: {Nb}Kab;
  5. A -> B: {Nb, Nb}Kab;
GOALS
  SECRET Kab;
  PRECEDES B: A | Nb, Kab;
END;
```

Source: Stephan

Tracebasierte- Spezifikation

- Induktive Definition von zulässigen Traces (axiomatisch)
- zentraler (techn.) Punkt (in Valikrypt): Datenstrukturen
  - Angriffsmöglichkeiten
- globale Sicht auf Protokollschritte + Angreifer
- Tracebasierte Axiomatisierung in Verification Support Environment (VSE)



Source: Stephan

Tracebasierte- Spezifikation

- Induktive Definition von zulässigen Traces (axiomatisch)
- zentraler (techn.) Punkt (in Valikrypt): Datenstrukturen
  - Angriffsmöglichkeiten
- globale Sicht auf Protokollschritte + Angreifer

```
  /* Step3: */
  NSch_symStep3 : NSch_sym_Says3(ev,evs) <->
                  EX A, B, Na, Kab, F :
                   (eventIN(Says(A,S,pair(agent(A),pair(agent(B),
nonce(Na)))),evs) AND
                    eventIN(Gets(A,crypt(csk(A,0),pair(nonce(Na),
pair(agent(B),pair(key(Kab),F))))),evs) AND
                   ev = Says(A,B,F));
```

Source: Stephan

Automatenbasierte Spezifikation

- Protokollabwicklung (operational) durch Zustandsautomaten
- Abstraktion bei Zuständen und Netz
- Unterscheidung: Schemata – Läufe
- Protokollschritte als Regelschemata
- offen für spezielle Formen der Protokollabarbeitung
- Angreifer als zusätzliche Komponente
- Beschreibungssprache für Protokollschemata in Simple Homomorphism Verification Tool (SHVT)

Source: Stephan

Automatenbasierte Spezifikation

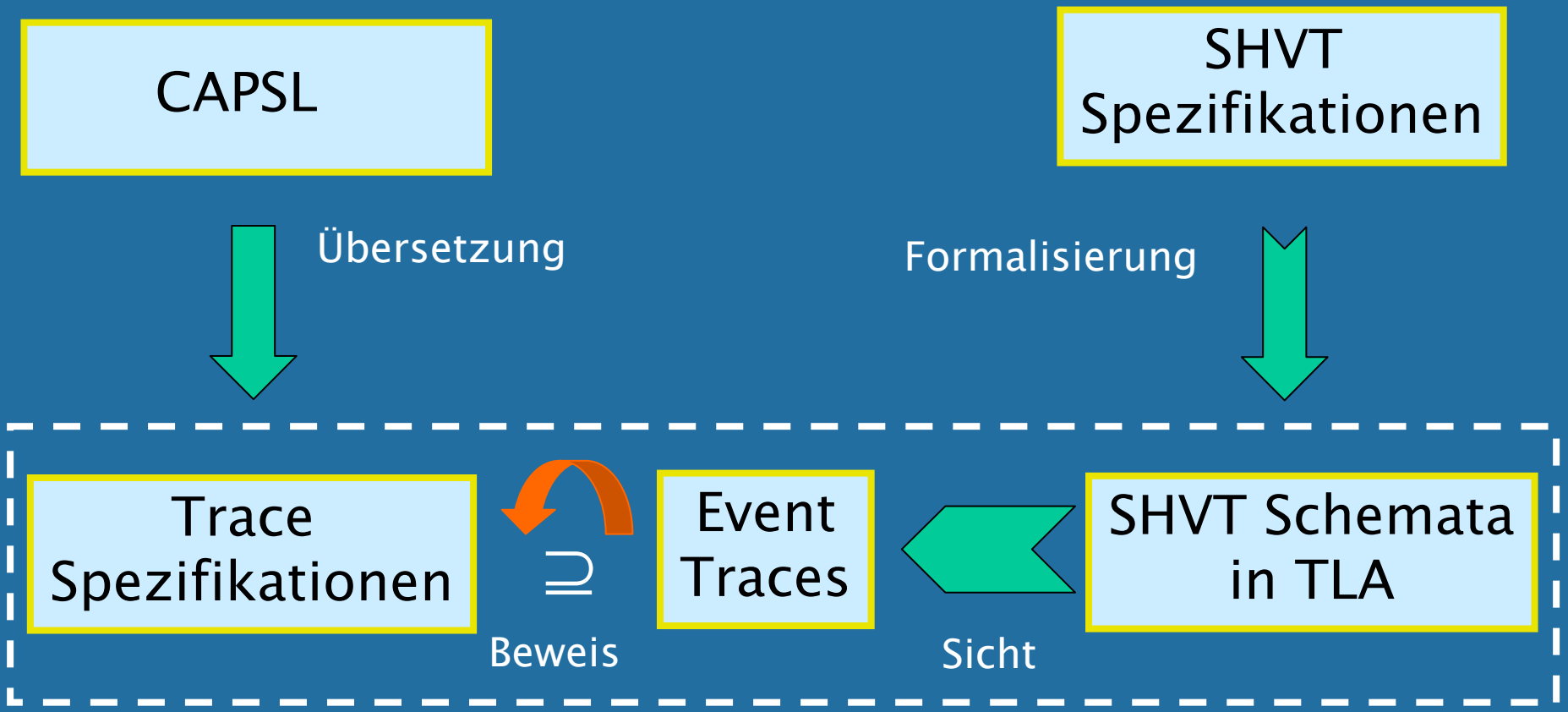
3. Variables: $X, B, S, M, Chiffretext, K_{AS}, K_{AB}, R_A$

$(X, A, M) \in \text{Network}$
$(B, R_A, S) \in \text{State}_\text{A}$
$(S, sym, K_{AS}) \in \text{Symkeys}_\text{A}$
$elem(1, decrypt(K_{AS}, M)) = R_A$
$elem(2, decrypt(K_{AS}, M)) = B$
$\xrightarrow{\text{A}}$
$K_{AB} := elem(3, decrypt(K_{AS}, M))$
$Chiffretext := elem(4, decrypt(K_{AS}, M))$

(X, A, M)	$\hookleftarrow$	Network
(B, R_A, S)	$\hookleftarrow$	State_A
$(new\ session\ key, B, K_{AB})$	$\hookrightarrow$	State_A
$(A, B, Chiffretext)$	$\hookrightarrow$	Network

Source: Stephan

Zusammenhang zwischen Spezifikationen



Source: Stephan

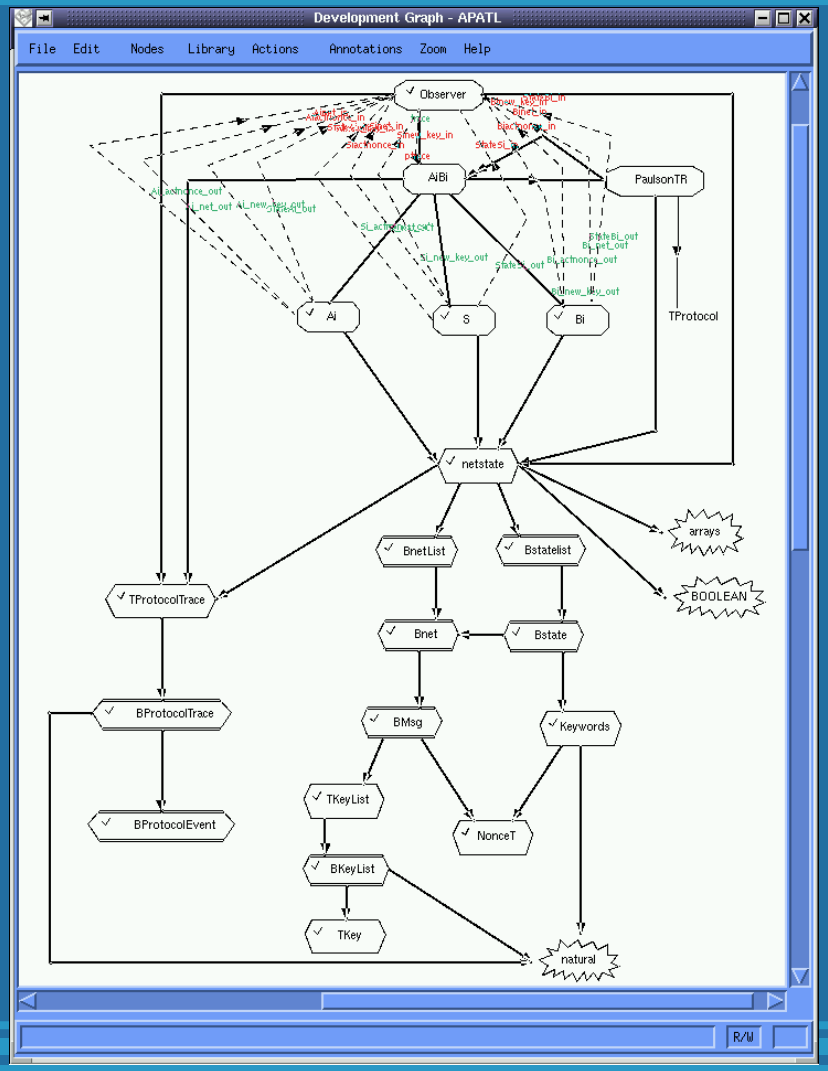
Zusammenhang zwischen Spezifikationen

- Modellierung von Automaten in VSE
  - Temporal Logic of Actions (TLA)
- alle möglichen Instanzen (der Schemata)
- „Observer“-Komponente extrahiert Traces
- Beweisverpflichtung: Die Menge der extrahierten Traces ist Teilmenge der induktiv definierten Traces
- Beweis: Invariante verwendet Vergangenheit
  - weitgehend uniform

Source: Stephan

Zusammenhang zwischen Spezifikationen

Entwicklungsgraph in VSE



Source: Stephan

Zusammenhang zwischen Spezifikationen

Zustandsübergang (Aktion) in VSE

```
A3i ::= Obs_enab = T AND
        EX S, X, A, B, M, Kas, Kab :
        (A <= max AND
         B <= max AND
         S <= max AND
         X <= max AND
         netmember(mknetentry(X, A, M), net) AND
         statemember(pairstateentry(mkagentnameentry(B),
                                    pairstateentry(mknonceentry(actnonce),
                                                   mkagentnameentry(S))),
                     select(StateAi, A)) AND
         statemember(pairstateentry(mkagentnameentry(S),
                                    pairstateentry(mkkeywordentry(Keywords.sym),
                                                   mkkeyentry(Kas))),
                     select(StateAi, A)) AND
         nonceMsg(actnonce) = get_elem(1, decrypt(Kas, M)) AND
         agentMsg(B) = get_elem(2, decrypt(Kas, M)) AND
         StateAi' = enterstateentry(A,
                                    pairstateentry(mkkeywordentry(newsessionkey),
                                     pairstateentry(mkagentnameentry(B),
                                                    mkkeyentry(getKey(get_elem(3, decrypt(Kas, M)))))),
                                    delstateentry(A, pairstateentry(mkagentnameentry(B),
                                                      pairstateentry(mknonceentry(actnonce'),
                                                                     mkagentnameentry(S))),
                                                  StateAi)) AND
         StateAi_out' = StateAi' AND
         net' = addnetentry(mknetentry(A, B, get_elem(4, decrypt(Kas, M))),
                 deletenetentry(mknetentry(X, A, M), net))) AND
```

Source: Stephan

Schwachstellenanalyse

- Finden von unerwünschten Konstellationen in konkreten Protokollläufen
  - effizient (vollautomatisch)
- Realisierung von Abläufen in SHVT
  - Instanzen von Schemata
  - Steuerung
  - Auswertung
- SHVT Analyseverfahren

Source: Stephan

Verifikation von Protokollen

- Induktiver Beweis von Eigenschaften von Traces
  - Formulierung von Eigenschaften
    - Vertraulichkeit, Authentizität, ...
- Interaktive Beweiserzeugung
  - Zerlegung in Lemmata
  - Anwendungsspezifische Heuristiken
  - Simplifikation
- Methodisch nach Abschluss des Debugging
- Beweisstrategie für Protokolle

Source: Stephan

Verifikation von Protokollen

```
SecretKab :
  (NSch_sym(evs) AND
   eventIN(Says(S,A,crypt(csk(A,0),pair(nonce(Na),
           pair(agent(B),pair(key(Kab),crypt(csk(B,0),
           pair(key(Kab),agent(A))))))),evs) AND
   NOT isBad(A)  AND  NOT isBad(B)  AND
   NOT EX Nb : eventIN(Notes(spy,pair(nonce(Na),
                       pair(nonce(Nb),key(Kab)))),evs)
  ) -> NOT msgIN(key(Kab), analz(gotInfo(spy, evs)))
```

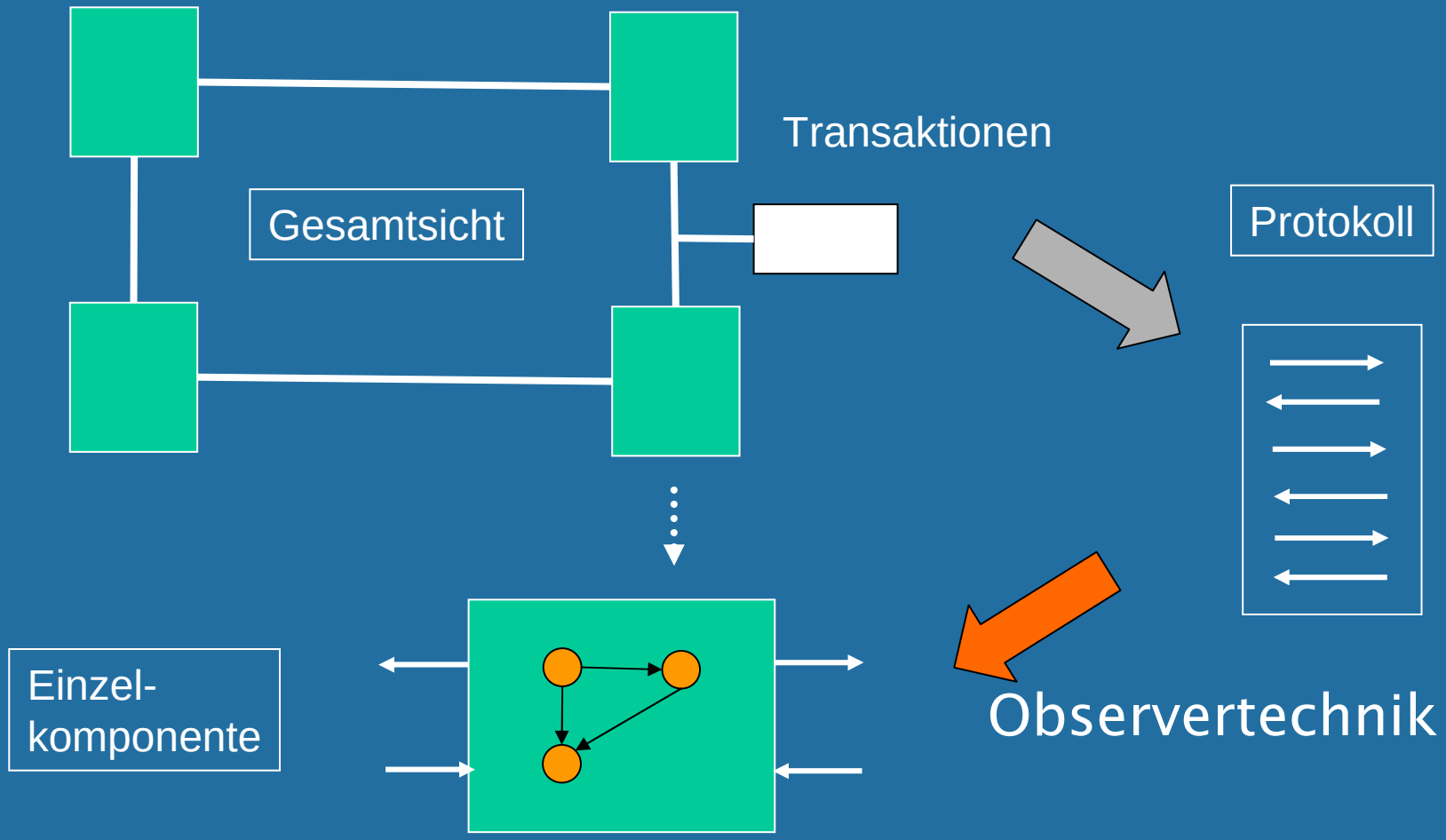
Source: Stephan

Einbettung in den Entwicklungsprozess

- Probleme:
  - Kryptoprotokolle bilden nur einen Aspekt.
  - Protokollanalyse zwischen Modellierung der Gesamtarchitektur und Komponentenspezifikation
- Valikrypt
  - Konzept für „Sichten“
  - Formalisierungstechniken und Beweistechniken für den Zusammenhang zwischen Komponenten und Trace-Spezifikationen

Source: Stephan

Einbettung in den Entwicklungsprozess



Source: Stephan

Zusammenfassung

Engineering (Ingenieurmäßige Bearbeitung) von Kryptoprotokollen:

- Integration von Verifikation und Schwachstellenanlyse
- Unterstützung der Analyseverfahren
  - Handhabung
  - Automatisierung
- Eigenschaften kryptographischer Bausteine
- Einbettung in den Entwicklungsprozess

Source: Stephan