

Model-based Security Engineering of Distributed Information Systems using UMLsec

Bastian Best, [Jan Jürjens](#), Bashar Nuseibeh

BMW Group

Munich, Germany

Department of Computing

The Open University, GB



J.Jurjens@open.ac.uk

<http://www.jurjens.de/jan>

Challenge: Security

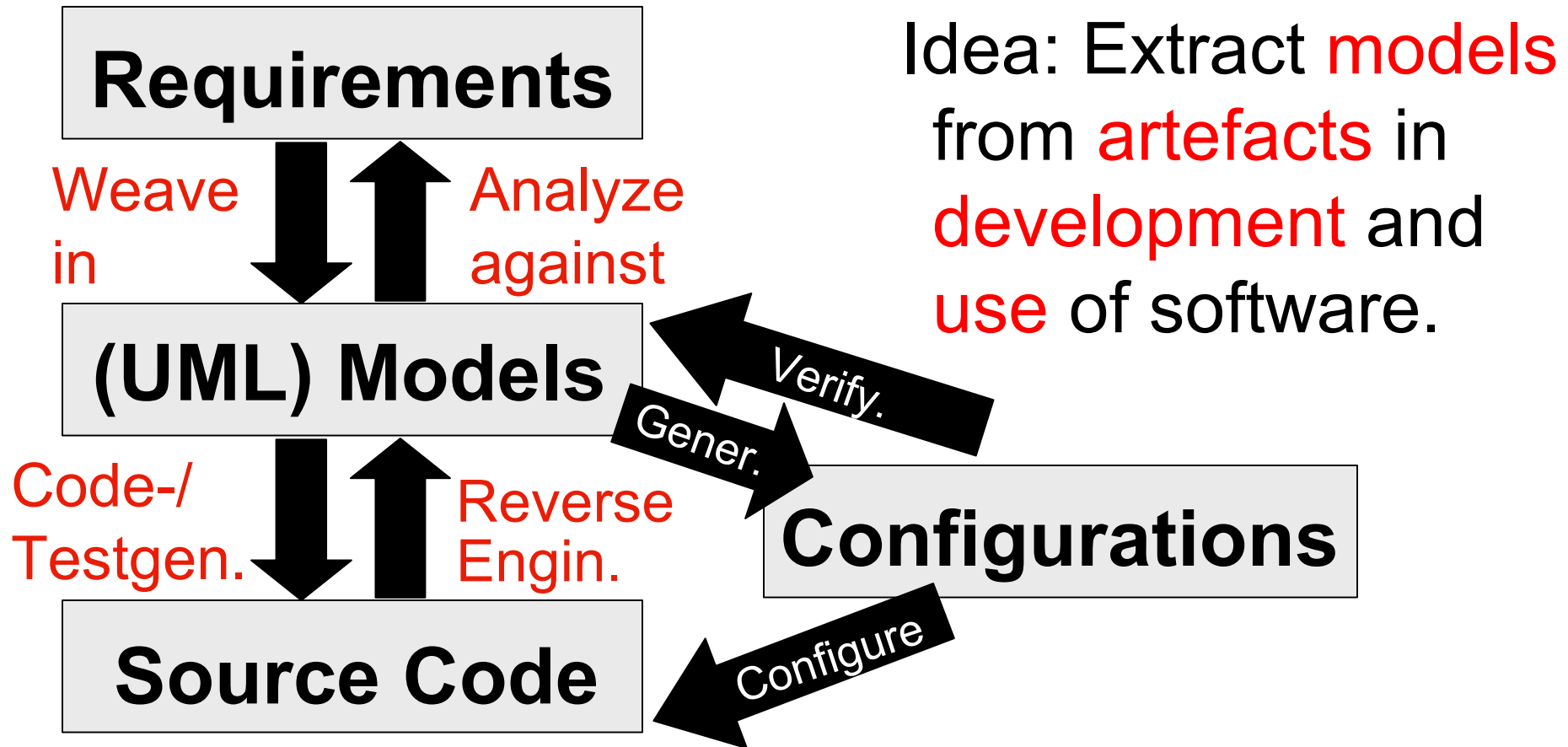
Security is **holistic** property:

- Attackers often **circumvent** (not: **break**) mechanisms.
- **Transform** (in)secure components to **secure systems** ?



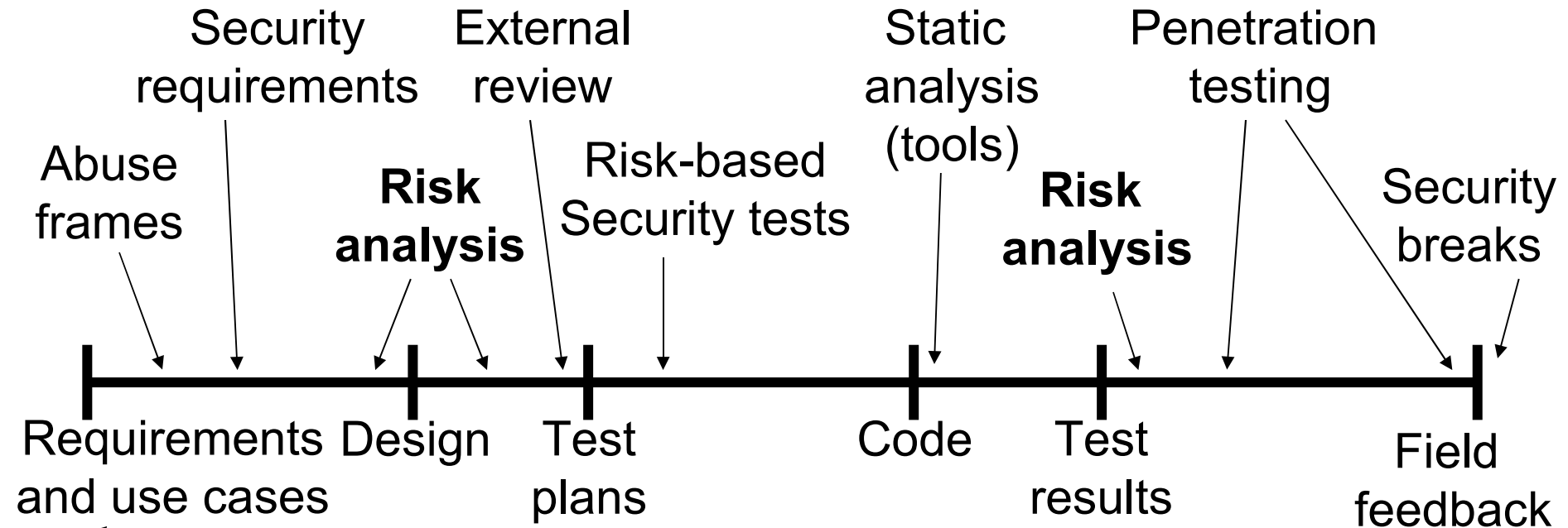
„Those who think that their problem can be solved by simply applying cryptography don't understand cryptography and don't understand their problem“
(B. Lampson / R. Needham).

Model-based Security Engineering



→ Tool-supported, theoretically sound, efficient automated security design & analysis.

Secure System Lifecycle



Model-based Security Engineering

[McGraw 2003]

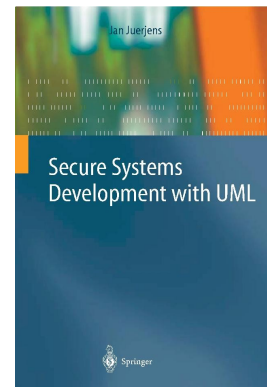
Design: Encapsulate prudent security engineering rules.

Analysis: Formally based, automated, efficient tools.

Note: emphasis on high-level requirements.

Model-based Security with UMLsec

- Extension of the Unified Modeling Language (UML) for **secure systems** development.
- evaluate UML models for security
 - encapsulate **established rules** of prudent secure engineering
 - make available to developers **not specialized** in secure systems
 - consider security requirements from **early** design phases, in system **context**
 - can use in certification

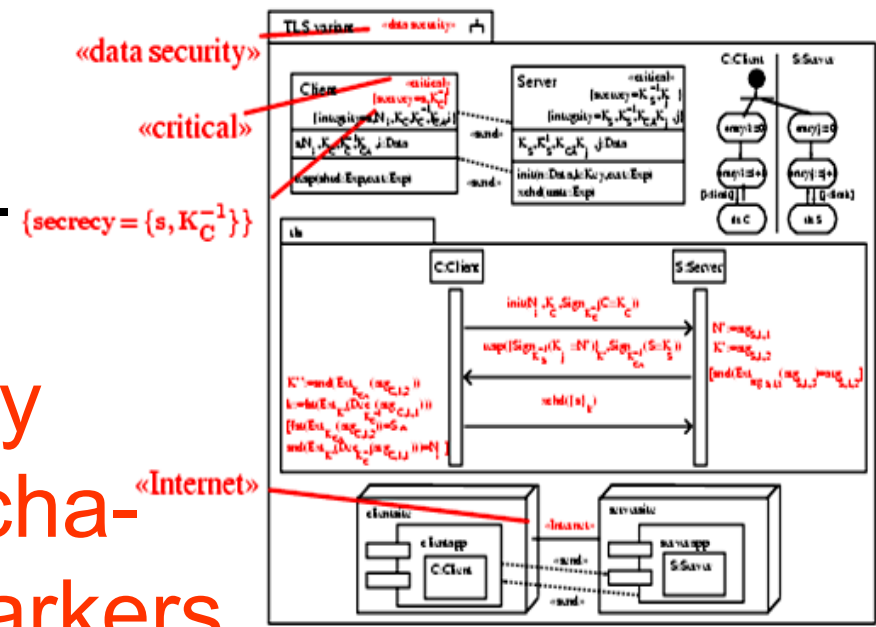


UMLsec

Insert recurring security requirements, adversary scenarios, security mechanisms as predefined markers.

Use associated logical constraints to verify specifications using model checkers and ATPs based on formal semantics.

Ensures that UML specification enforces the relevant security requirements wrt Dolev-Yao type adversaries. [FASE01,UML02,FOSAD05,ICSE05]

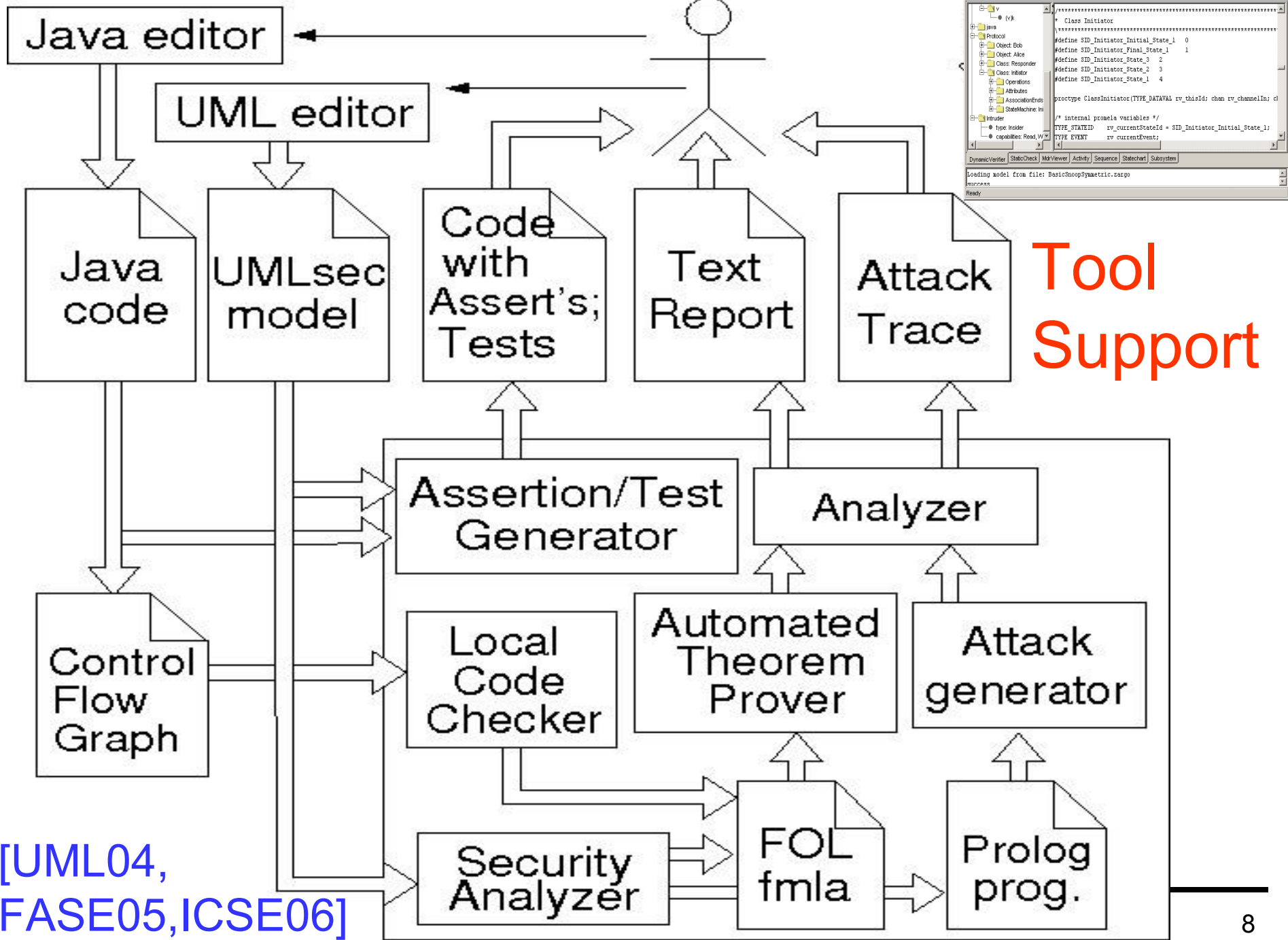


Tool Support

For example:

- consistency checks
- mechanical analysis of complicated requirements on model level (bindings to model-checkers, constraint solvers, automated theorem provers, ...)
- code generation
- test-sequence generation
- configuration data analysis against UML.





[UML04, FASE05, ICSE06]

Intranet Information System

MetaSearch Engine: **Personalized search** in company **intranet** (including **password protected**).

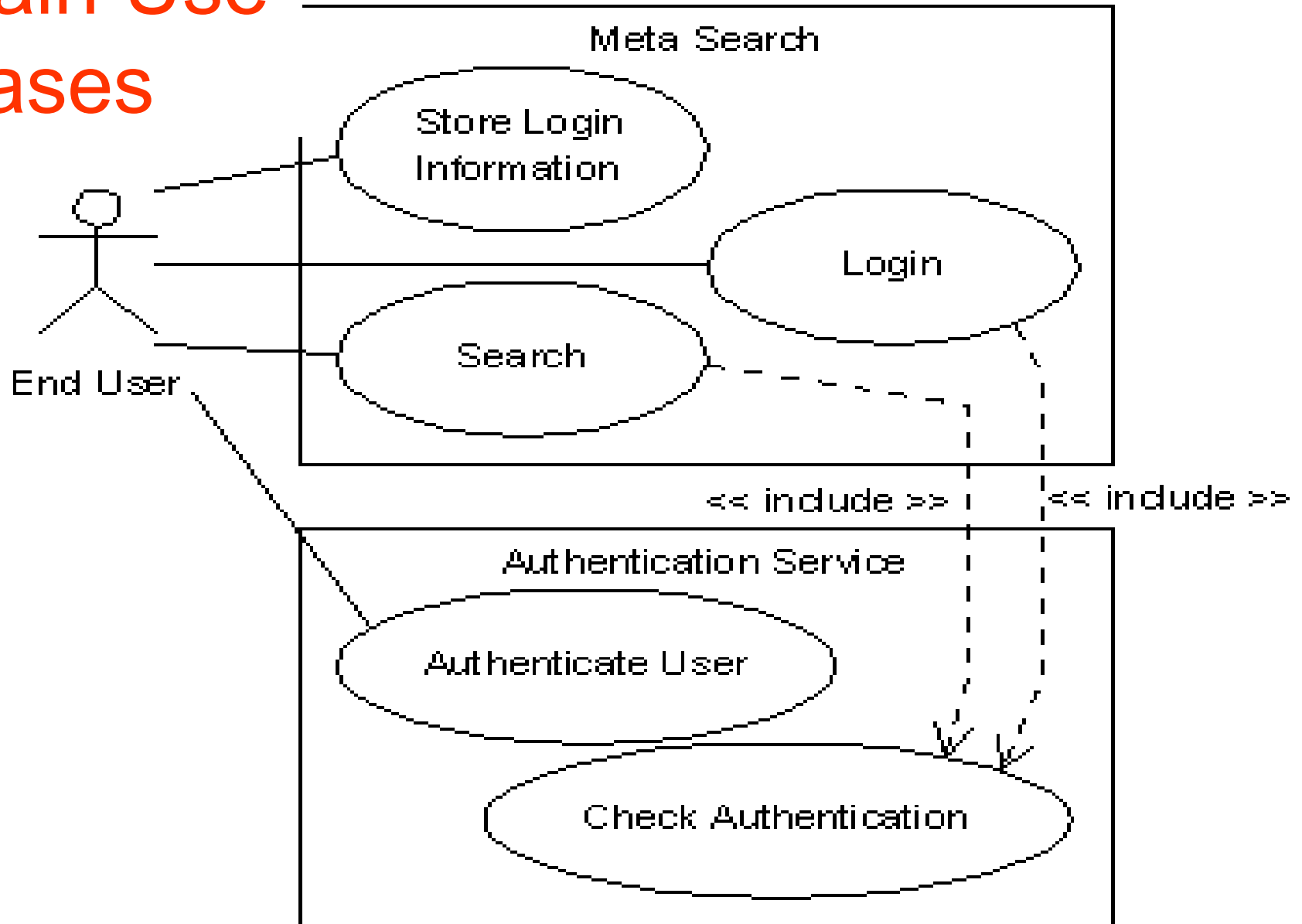
Some documents highly **security-critical**.

More than 1,000 potential users, index 280,000 documents, allow 20,000 queries per day.

Seamlessly integrated in **enterprise-wide security reference architecture**. Provides **security services to applications**, including **user authentication**, **role-based access control**, **global single-sign-on** and hook-up of **new security apps**.

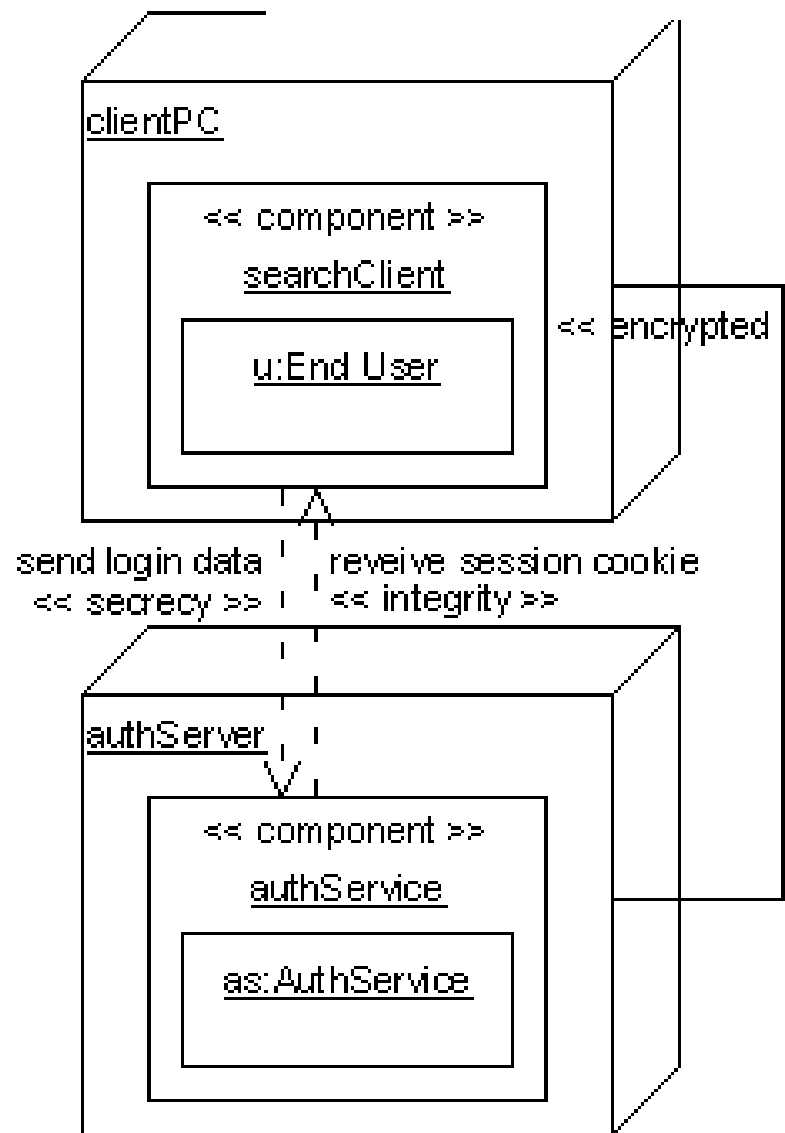


Main Use Cases

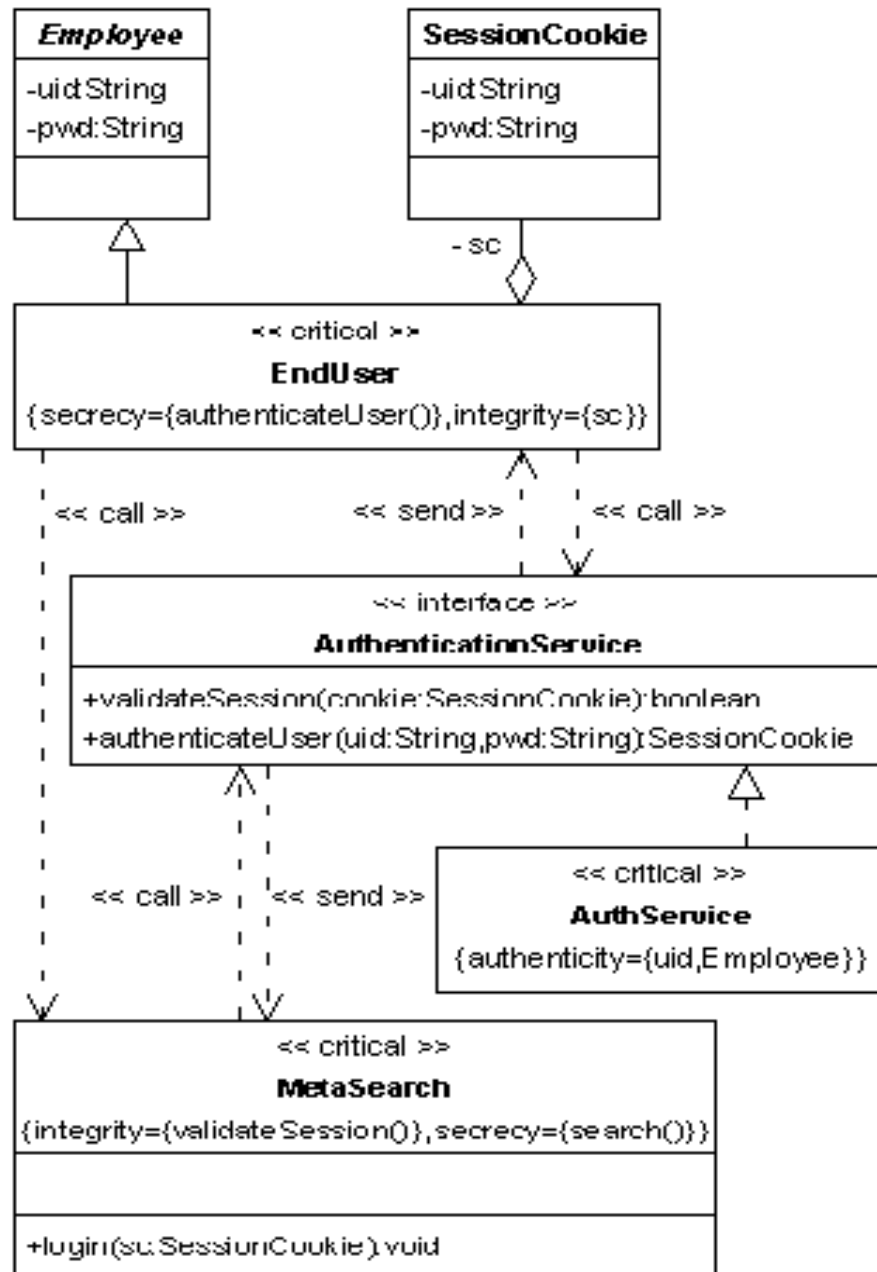


Some Static Models

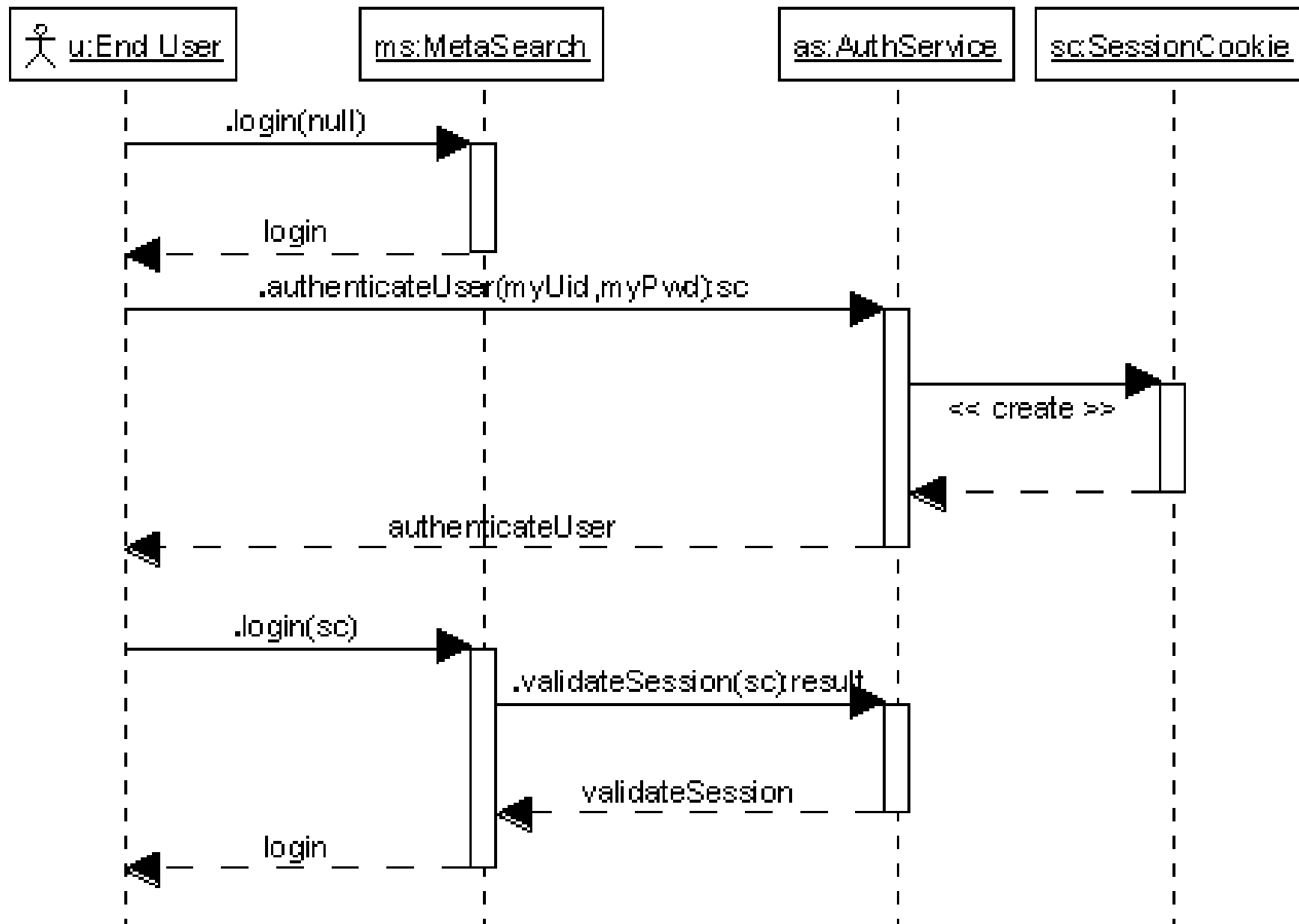
Physical Allocation



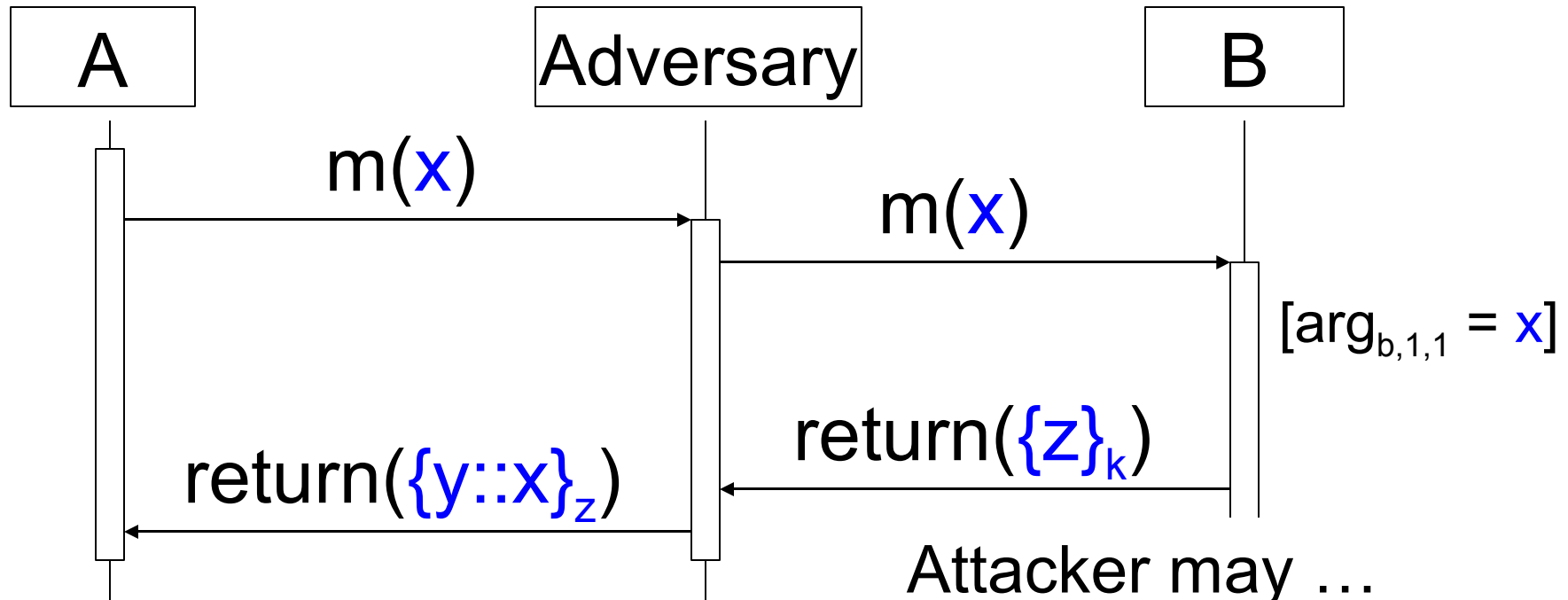
Structuring Entries



A Dynamic Model



Security Analysis of Crypto-based Systems



Adversary
knowledge:

k^{-1}, y, x
 $\{z\}_k, z$

(cf. [Dolev, Yao 1982])

Attacker may ...

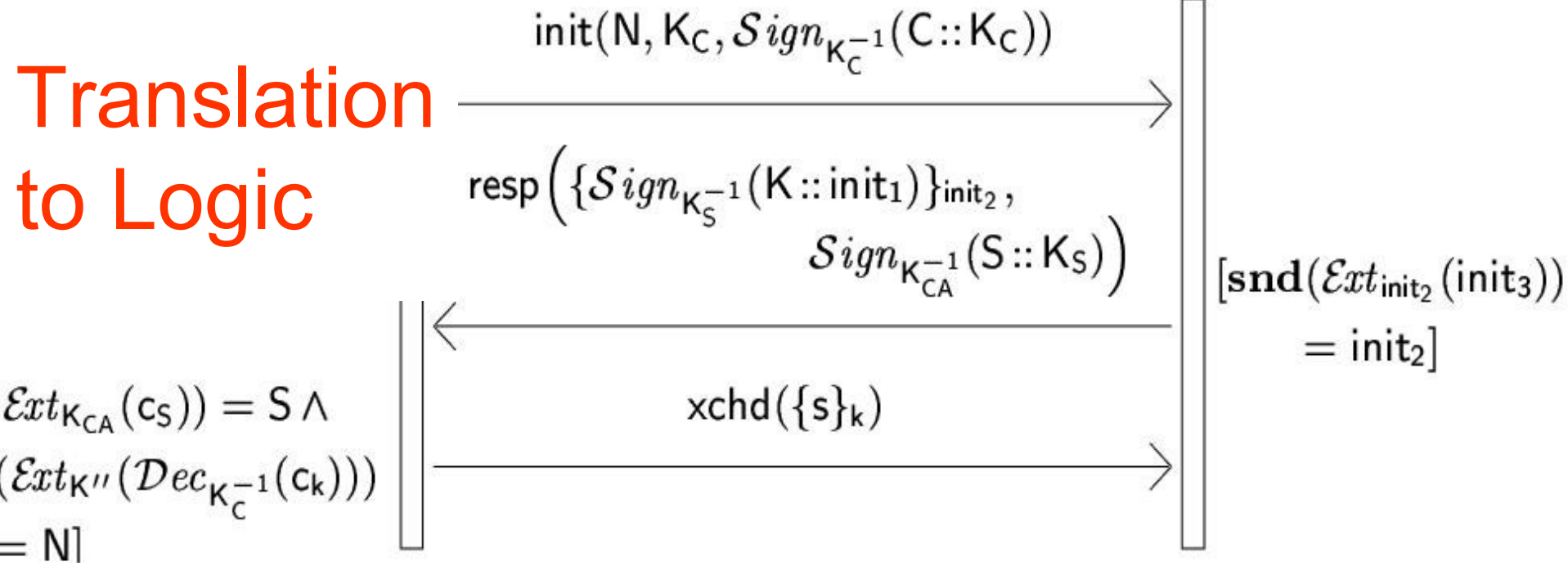
- **control** system parts,
- **know** data in advance,
- **intercept** messages,
- **delete** messages,
- **inject** messages.



C:Client

S:Server

Translation to Logic



$\text{knows}(N) \wedge \text{knows}(K_C) \wedge \text{knows}(\text{Sign}_{K_C^{-1}}(C::K_C))$
 $\wedge \forall \text{init}_1, \text{init}_2, \text{init}_3. [\text{knows}(\text{init}_1) \wedge \text{knows}(\text{init}_2) \wedge$
 $\text{knows}(\text{init}_3) \wedge \text{snd}(\text{Ext}_{\text{init}_2}(\text{init}_3)) = \text{init}_2$
 $\Rightarrow \text{knows}(\{\text{Sign}_{K_S^{-1}}(\dots)\}_{\dots}) \wedge [\text{knows}(\text{Sign} \dots)]$
 $\wedge \forall \text{resp}_1, \text{resp}_2. [\dots \Rightarrow \dots]]$

Analysis

Check e.g. whether
can derive *knows(s)*
e.g. using e-Setheo.

If yes:

→ Protocol does **not**
preserve secrecy of *s*.

Why ? Use Prolog-
based **attack**
generator.

```
input_formula(tls_abstract_protocol, axiom, (  
  ![ArgS_11, ArgS_12, ArgS_13, ArgC_11, ArgC_12] : (  
    ![DataC_KK, DataC_k, DataC_n] : (  
      % Client -> Attacker (1. message)  
      (  
        knows(n)  
        & knows(k_c)  
        & knows(sign(conc(c, k_c),  
& % Server -> Attacker (2. message)  
      (  
        knows(ArgS_11)  
        & knows(ArgS_12)  
        & knows(ArgS_13)  
        & ( ? [X] :  
=> ( knows
```

E-SETHEO csp03 single processor running on host ...
(c) 2003 Max-Planck-Institut fuer Informatik and
Technische Universitaet Muenchen
+levariant-freshbench-fresh
analyzing results ...
proof found
time limit information: 298 total / 297 strategy
... , inv(k_ca)), ArgC_1
... e_k, DataC_n), inv(DataC
& (... 11)
& (... equal(sign(conc(s, DataC_ks), i
ArgC_12))
& equal(... gn(conc(DataC_k, n), inv(DataC_KK))
ArgC_11)
& equal(enc(sign(conc(DataC_k, DataC_n), inv(DataC
ArgC_11)
)
=> (knows(symenc(secret, DataC_k)))))



Security Analysis of Metasearch

Applied above approach to **security-critical core** of Metasearch (**single sign-on** mechanism).

To **assess effectiveness**, performed iterative approach. Start with model with several **flaws**. Subsequently **found by the tools**.

Note **100% security proof** is **impossible** for principled reasons. **Goal is optimal cost effectiveness** for finding weaknesses in highly **security-critical system parts**.

The **security properties** that were considered were found to be **enforced** (details in paper).



Some Insights

- **Model-based development** with notations such as UML does incur **effort**.
- That effort seems **manageable** when applied to **core critical parts** of the system.
- It seems to be **justifiable** in case of **high assurance** needs (e.g. in **security**).
- We believe it to **compare favorably** with **traditional assurance methods** offering a similar degree of **trustworthiness**.
- **UMLsec** seems to be **well-suited** for the domain of **distributed information systems**.



Possible Improvements

Experiences indicated potential improvements:

- **extend notation** further (e.g. to cover specialized domains such as mobility)
- provide **tool support** for these extensions
- Improve **intuitiveness** of parts of the existing tool support

Need further case-studies to investigate recent work on relating **models to code** [ASE05,06].



Some Other Applications

Analyzed designs / implementations / configurations for

- biometry, smart-card or RFID based identification
- authentication (crypto protocols)
- authorization (user permissions, e.g. SAP systems)

Analyzed security policies, e.g. for privacy regulations.

T-Systems

Allianz

Deutsche Bank

HypoVereinsbank

CEPS™

BMW Group

msg systems

Münchener Rück
Munich Re Group

Bundesministerium
für Bildung
und Forschung

Bundesministerium
der Verteidigung

O₂

infineon

Bundesministerium
für Wirtschaft
und Technologie

Related Approaches

UML + security:

- RBAC: Fernandez et al., Basin et al., Breu et al., Koch/Parisi-Presicce, ...
- Aspect-Oriented Modeling (France et al.)
- Model-based Risk Assessment (CORAS project, Stoelen, Houmb et al.)
- Agents: Yoshioka, Honiden, Finkelstein
- ...

Conclusions

Application of the **UMLsec** approach in **industrial setting**.

Model-based **security analysis** of **intranet search engine**. Benefits were:

- consideration of security goals within a **standard industrial design technique**
- **automated security analysis**

The approach was found to be applicable with **justifiable training and time effort**.





What Does UMLsec Cover ?

Security requirements: `<<secrecy>>`,...

Threat scenarios: Use `Threatsadv(ster)`.

Security concepts: For example `<<smart card>>`.

Security mechanisms: E.g. `<<guarded access>>`.

Security primitives: Encryption built in.

Physical security: Given in deployment diagrams.

Security management: Use activity diagrams.

Technology specific: Java, CORBA security.



UMLsec as Integrating Formal Framework

Have formalizations of major security requirements in one integrated notation.

Want to **relate** / **combine** requirements; get **modularity** / **composability**, hierarchical **decomposition**, **refinement**, ... :

For example:

- If system satisfies **«secure links»** and subsystems satisfy **«data security»** then system satisfies **«data security»**.



Refinement & Composability

Need to **refine models** down to code.

Common formalizations of security properties **not preserved** by refinement.

Bad: **re-verify** after each step (incl **code**).

Theorem: Our notion of model **refinement** **preserves security** requirements. [FME01]

Similar: Established **composability** for certain security requirements under suitable assumptions. [Concur01]

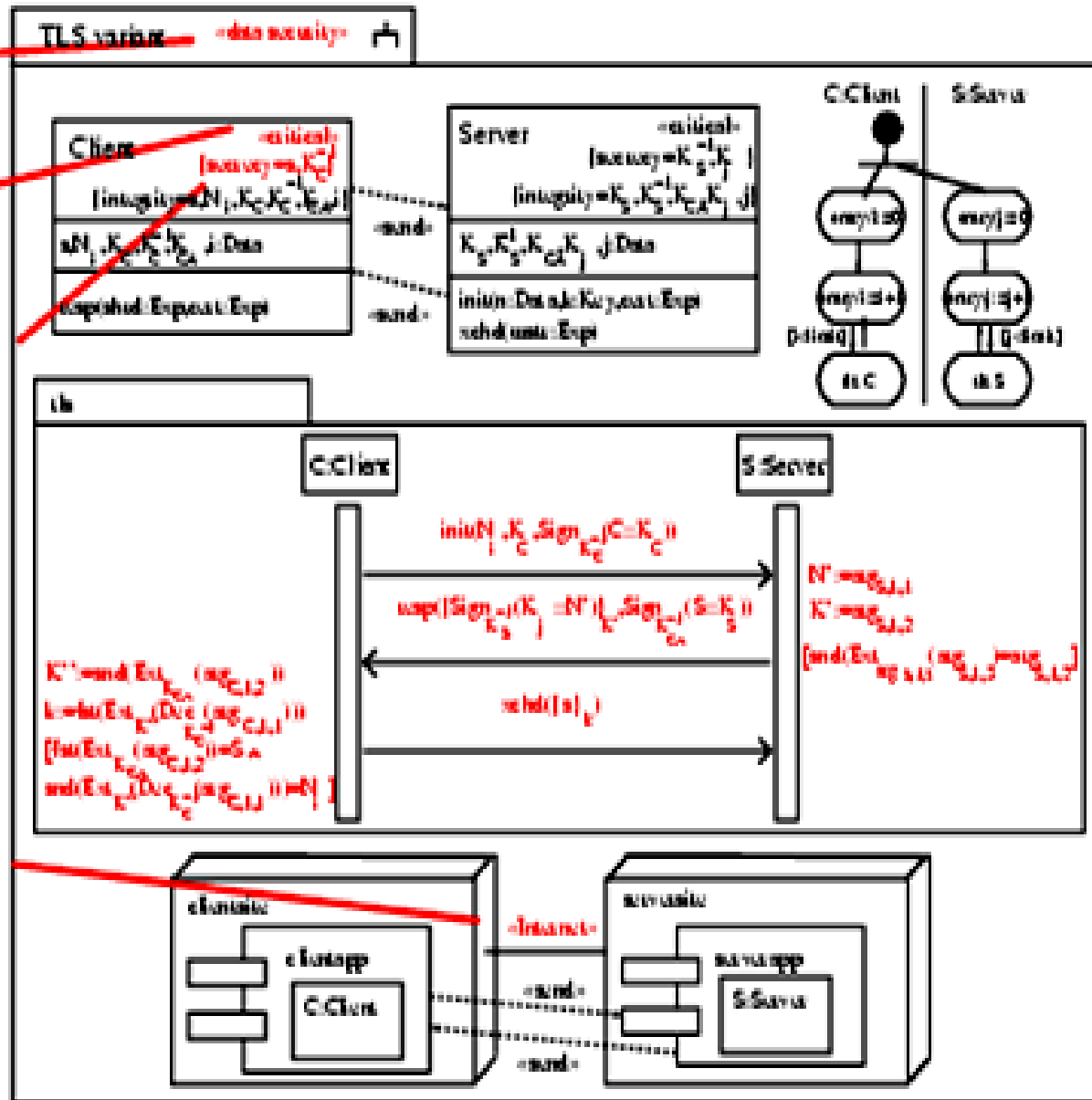


Example: TLS Variant

«data security»

«critical»

{secrecy = {s, K_C^{-1} }}

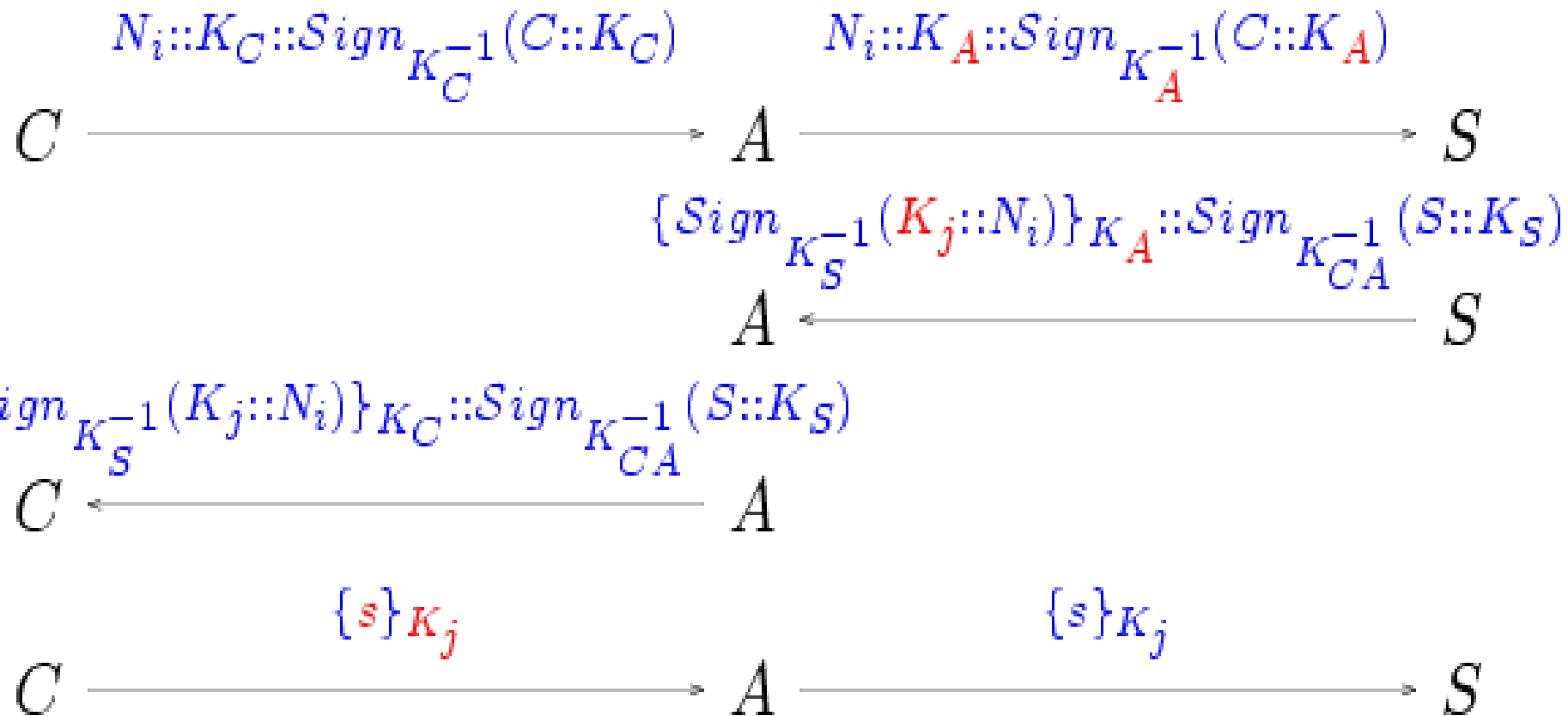


«Internet»

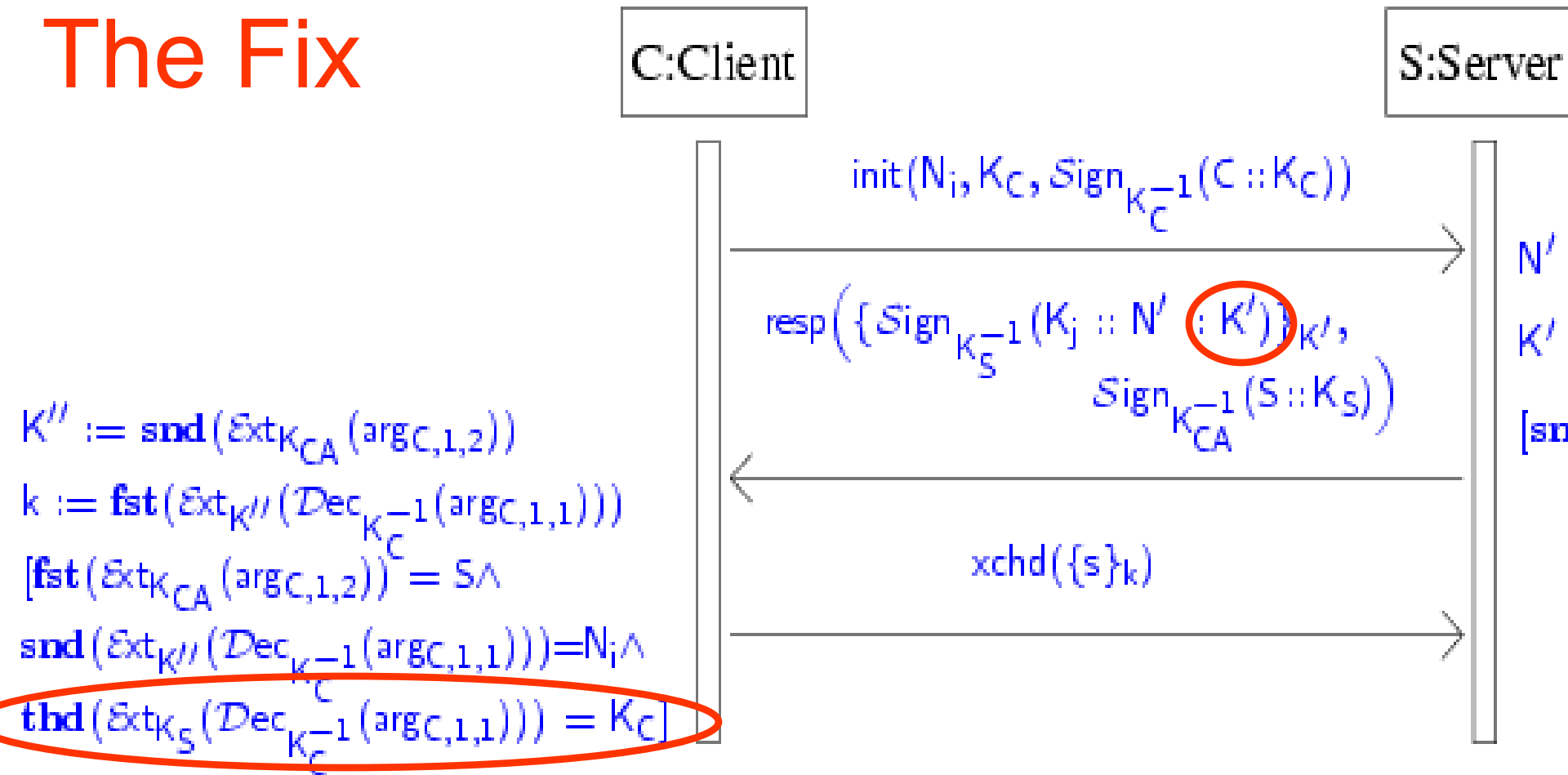
Presented at
 IEEE Infocom
 1999
 Goal: send
 secret protected
 by session key
 using fewer
 server
 resources.



Man-in-the-Middle Attack



The Fix



e-Setheo: **Proof** that $knows(s)$ not derivable.

Note **completeness** of FOL (but also undecidability).

Layered Security Protocols

System layer on **top uses security** services **below.**

client authenticity



confidentiality, integrity, server authenticity



=

confidentiality, ... + client authenticity



Security properties **additive** ? [Safecom03]

Theorem: **Yes**, under suitable conditions.



Model-based Security Aspects

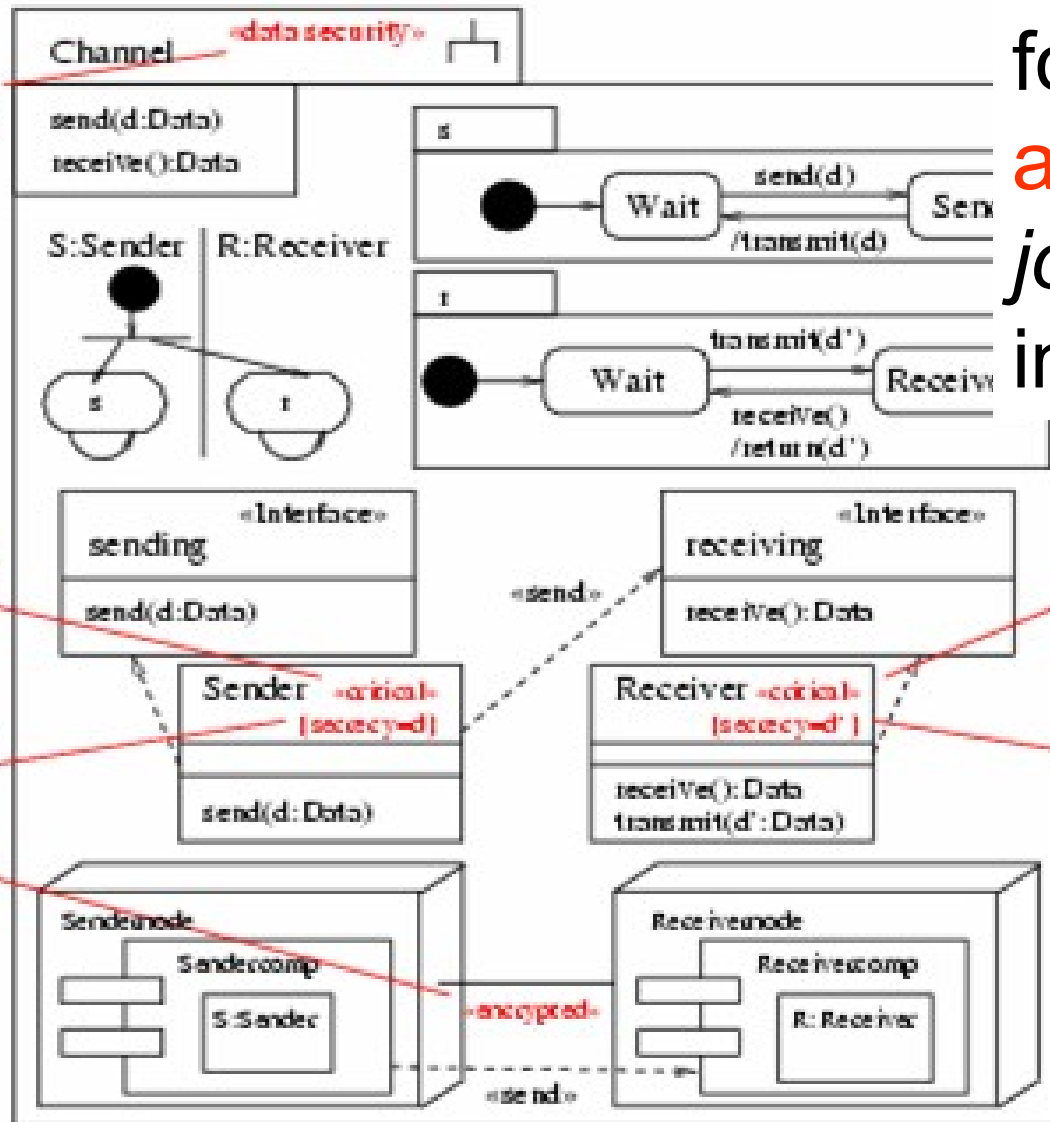
- Define **abstract security aspect**.
- Define **concretization** (e.g. protocol).
- Applying aspects results in transformation on models.
- If possible, give conditions under which it is **secure** to **weave in** aspect using concretization, e.g. by simulation argument.



Secure Channel Aspect

Primary model with directives for security aspects (cf. join points in AspectJ).

«data security»



«critical»

{ secrecy=d }

«encrypted»

«critical»

{ secrecy=d' }

Aspect Validation

Need to prove concretization securely refines abstract aspect. Challenging problem in security.

For secure channel, have generic result.

Not always possible. Then use **translation validation** on the weaving transformation, before or after code generation.



Bank Application

[SAFECOMP03]

Security analysis of web-based banking application, to be put to commercial use (clients **fill out** and **sign** digital order forms).

Layered security protocol (first layer: SSL protocol, second layer: client authentication protocol)

Security requirements:

- **confidentiality**
- **authenticity**

The screenshot shows the HypoVereinsbank website interface. At the top, there is a navigation bar with links: Über uns | Presse | Investor Relations | Research | Jobs und Karriere | Überblick | Hilfe | Datenschutz | Kontakt | HVB Group. Below the navigation bar is a banner with the text "Leben Sie. Wir kümmern uns um die Details." and the HypoVereinsbank logo. The main content area is divided into several sections: a "Hier empfehlen wir Ihnen mal einen Fonds der Konkurrenz!" section with a photo of a man; a "TOOLBOX" section with links to Lexikon, Filialfinder, Formularfinder, Newsletter, Geschäftsbedingungen & Konditionen, and Kurssuche; a news section with several articles; a "Privatkunden in Sachen Privatleben" section; a "Businesskunden In Businessangelegenheiten" section; a "Log In Direct B@nking" section with a login form; and a "e@sy credit. Einfach Wünsche erfüllen." section with several promotional items. The bottom of the page features a search bar and a "Gastzugang" link.



Common Electronic Purse Specifications

Global elec. purse standard (Visa, 90% market).
Smart card contains account **balance**, performs **crypto** operations securing each transaction.
Formal analysis of load and purchase protocols:
three significant weaknesses: purchase redirection, fraud bank vs. load device owner.



Biometric Authentication System

In development by company in joint project.

Uses bio-reference template on smart-card.

Analyze given UML spec.

Discovered **three major weaknesses** in subsequently improved versions (**misuse counter circumvented** by dropping / replaying messages, **smart-card insufficiently authenticated** by mixing sessions). [\[ACSAC05\]](#)



Ongoing Research

Ongoing work on most of the above issues:

- **Secure systems** out of (in)secure mechanisms.
- Security as **pervasive property** - problem: no **integration / coherence**. **Necessary** for security (attacks on **boundaries** between views / aspects / levels ...).
- Security properties: E.g. **composability**

Applications: **Security reference architecture**
e.g. for **mobile** or **ubiquitous** systems.