

Information Security

Holger Schlingloff

Nov 21st, 2001



Where are we?

Chapter 0: Introductory example (nimda)

Chapter 1: Fundamental terms and definitions

Chapter 2: Viruses, Worms, Trojan Horses (threats)

- Threats by code execution
 - ▷ Viruses, worms, trojans, mobile code
- Network threats
 - ▷ application-, transport-, network-, link-layer

Chapter 3: Construction of secure systems

- General security principles
- Security engineering process
- Threat and risk analysis
- Security strategy and modelling
- Security classification; security infrastructure

Chapter 4: Cryptographical algorithms

Further Chapters: Identification, Authorization

Mensa cash card example

System components: (Smart) cards, cash registers, charge terminals, display stations, server, counter, database

Design decisions: Dumb or smart card? Banking/Cash card? w/o PIN? w/o user name?

- Cash amount on card \Rightarrow subject to tampering
- Banking/Cash cards \Rightarrow Additional threats by interconnection with banks
- (No) user name on card \Rightarrow (no) immediate assignment possible

- No PIN for card \Rightarrow low protection of users against unauthorized use
(lost/stolen cards, but also: copying of card information)
- Similar considerations for other system components

Security strategy and modelling

First step: formulation of the **security goals** (properties the system is supposed to guarantee)

Derived from negation of threats from the fault tree analysis

Example: “it is always the case that the user id on a card is unchanged”

Ideally: Formulation in some logical language

Classification of properties according to their importance (is directly derived from the risk analysis)

Predicate logic

Syntax:

Definition 1 (Signature)

A *signature* $\Sigma = (\mathbf{D}, \mathbf{F}, \mathbf{R}, \tau)$ consists of a finite set \mathbf{D} of *domain names*, a finite set \mathbf{F} of *function symbols*, and a finite set \mathbf{R} of *relation symbols*. Associated with each function and relation symbol is its type τ , which is a sequence of domain names (nonempty in the case of functions).

Usually assumed: reals $\mathfrak{R} \in \mathbf{D}$, $\{0, 1, -, +, *, \dots\} \subseteq \mathbf{F}$, $\{<, \leq, \dots\} \subseteq \mathbf{R}$.

Given a signature Σ , let \mathcal{V} be a set of **individual variables**, where each variable has an appropriate type.

Definition 2 (Term)

object term t of type D is

- x , where x is an individual variable of type D , or
- $ft_0\dots t_{n-1}$, where f is a function symbol of type (D_0, \dots, D_{n-1}, D) and each t_i is an object term of type D_i for $i < n$.

As a special case, each constant symbol of type (D) is an object term of type D . Sometimes, parenthesis $f(t_0\dots t_{n-1})$ or infix notation t_0ft_1 is used for better readability.

Examples: 17 , $input(i)$, $\sin x$, $x + 3$, $17 + 4$, ...

Definition 3 (Formula)

well formed formula φ :

- $\rho t_1 \dots t_n$, where ρ is a relation symbol of type (D_1, \dots, D_n) , and t_i is an object term of type D_i for all $i \leq n$,
- $t_1 = t_2$, where t_1 and t_2 are object terms of the same type
- \perp , $(\varphi \rightarrow \psi)$, where φ and ψ are well formed formulas, and
- $\exists x \varphi$, where φ is a well formed formula, and x is an

individual variable

Abbreviations:

$$\top \triangleq \neg \perp,$$

$$\neg \varphi \triangleq (\varphi \rightarrow \perp),$$

$$(\varphi_1 \vee \varphi_2) \triangleq (\neg \varphi_1 \rightarrow \varphi_2),$$

$$(\varphi_1 \wedge \varphi_2) \triangleq \neg(\neg \varphi_1 \vee \neg \varphi_2),$$

$$\forall x \varphi \triangleq \neg \exists x \neg \varphi$$

Examples:

- $17 + 4 = 21,$
- $\exists x \sin x = 1,$
- $input(t) \rightarrow output(t + 1),$

- $\forall \epsilon \exists \delta (x + \epsilon < c_1 \rightarrow f(x) + \delta < c_2)$

Semantics:

Definition 4 (Structure)

A *structure* S (or *algebra*) S consists of a collection of disjoint sets called *domains*, and a collection of functions and relations over these domains. Elements of the domains are called *objects*.

Definition 5 (Interpretation)

An *interpretation* \mathcal{I} for a signature Σ on a structure S is a mapping $\mathcal{I} : \Sigma \rightarrow S$ assigning a nonempty domain $\mathcal{I}(D)$ for each domain name D , and a function $\mathcal{I}(f)$ and relation $\mathcal{I}(R)$ of appropriate type for each function and relation symbol, respectively.

Definition 6 (Valuation)

A **variable valuation** \mathbf{v} is a mapping assigning an object $\mathbf{v}(x) \in D$ to every individual variable x of type D . A **model** $\mathcal{M} \triangleq (\mathcal{S}, \mathcal{I}, \mathbf{v})$ for the signature Σ consists of a structure \mathcal{S} , an interpretation \mathcal{I} , and a variable valuation \mathbf{v} .

Definition 7 (Model)

A model $\mathcal{M} \triangleq (\mathcal{S}, \mathcal{I}, \mathbf{v})$ consists of a structure, interpretation and valuation. It determines a unique object $t^{\mathcal{M}}$ for every term t , and a unique truth value $\varphi^{\mathcal{M}} \in \{\mathbf{tt}, \mathbf{ff}\}$ for any formula φ . This **denotation** of terms and formulas is defined as follows:

- $x^{\mathcal{M}} \triangleq \mathbf{v}(x)$, if $x \in \mathcal{V}$ is an individual variable,

- $(ft_1\dots t_n)^{\mathcal{M}} \triangleq \mathcal{I}(f)(t_1^{\mathcal{M}} \dots t_n^{\mathcal{M}}),$
- $(\rho t_1\dots t_n)^{\mathcal{M}} = \mathbf{tt}$ *iff* $(t_1^{\mathcal{M}}, \dots, t_n^{\mathcal{M}}) \in \mathcal{I}(\rho),$
- $(t_1 = t_2)^{\mathcal{M}} = \mathbf{tt}$ *iff* $t_1^{\mathcal{M}} = t_2^{\mathcal{M}}$
- $\perp^{\mathcal{M}} \triangleq \mathbf{ff},$
- $(\varphi \rightarrow \psi)^{\mathcal{M}} = \mathbf{tt}$ *iff* $\varphi^{\mathcal{M}} = \mathbf{tt}$ *implies* $\psi^{\mathcal{M}} = \mathbf{tt},$ *and*
- $(\exists x \varphi)^{\mathcal{M}} = \mathbf{tt}$ *iff* $\varphi^{(S, \mathcal{I}, \mathbf{v}')} = \mathbf{tt}$ *for some valuation* \mathbf{v}' *which differs from* \mathbf{v} *at most in* x

$\mathcal{M} \models \varphi$ if φ denotes the value \mathbf{tt} in $\mathcal{M}.$

Models are executions of programs; Formulas are used to specify which executions are allowed

Example: specifications for charge terminal

Signature:

update(c, x, y, t): An update request is being sent from the terminal to the server for user x with amount y at time t

query(c, x, t): A query is being sent from the terminal to the server for user x at time t

response(c, x, y, t): A response is being received by the terminal from the server for user x with amount y at time t

insert(c, x, t): Card for user x is inserted in the terminal at time t

eject(c, t): Card is ejected from the terminal at time t

payment(c, y, t): Cash amount y is paid at the terminal at time t

More detailed specification level:

describe payment mechanism (input slot open/closed, motor on/off/reverse, validity check for money passed, ...)

describe card identification mechanism (reading of information, decryption, ...)

Properties:

If the cash amount y is paid at terminal c for user x , an appropriate *update* will be sent to the server.

$$has_card(c, x, t_1, t_2) \stackrel{\Delta}{=} insert(c, x, t_1) \wedge t_1 < t_2 \wedge eject(c, t_2) \wedge \neg \exists t (t_1 < t < t_2 \wedge eject(c, t))$$

$$(has_card(c, x, t_1, t_3) \wedge t_1 < t_2 < t_3 \wedge payment(c, y, t_2) \rightarrow update(c, x, y, t_3))$$

Other properties

for each insert exactly one eject after at most t time units,
between insert and eject at most one input,...

Security property:

Update-Befehle will be sent only after the appropriate amount has been paid (not for other reasons such as tampering):

$$\text{update}(c, x, y, t_3) \rightarrow \exists t_1, t_2 (\text{has_card}(c, x, t_1, t_3) \wedge t_1 < t_2 < t_3 \wedge \text{payment}(c, y, t_2))$$

Network correctness property:

Each *update* which is sent by any terminal is received by the server without modifications.

Each *update* received by the server was really generated by a cash terminal (and not inserted into the net).

$$receive_u(x, y, t) \rightarrow \exists c, t_1 (t_1 < t \wedge update(c, x, y, t_1))$$