GEFÖRDERT VOM

Bundesministerium
für Bildung
und Forschung

Verisoft

Verification as Engineering

Gerd Beuster

gb@uni-koblenz.de

Universität Koblenz-Landau

Verisoft

Overview

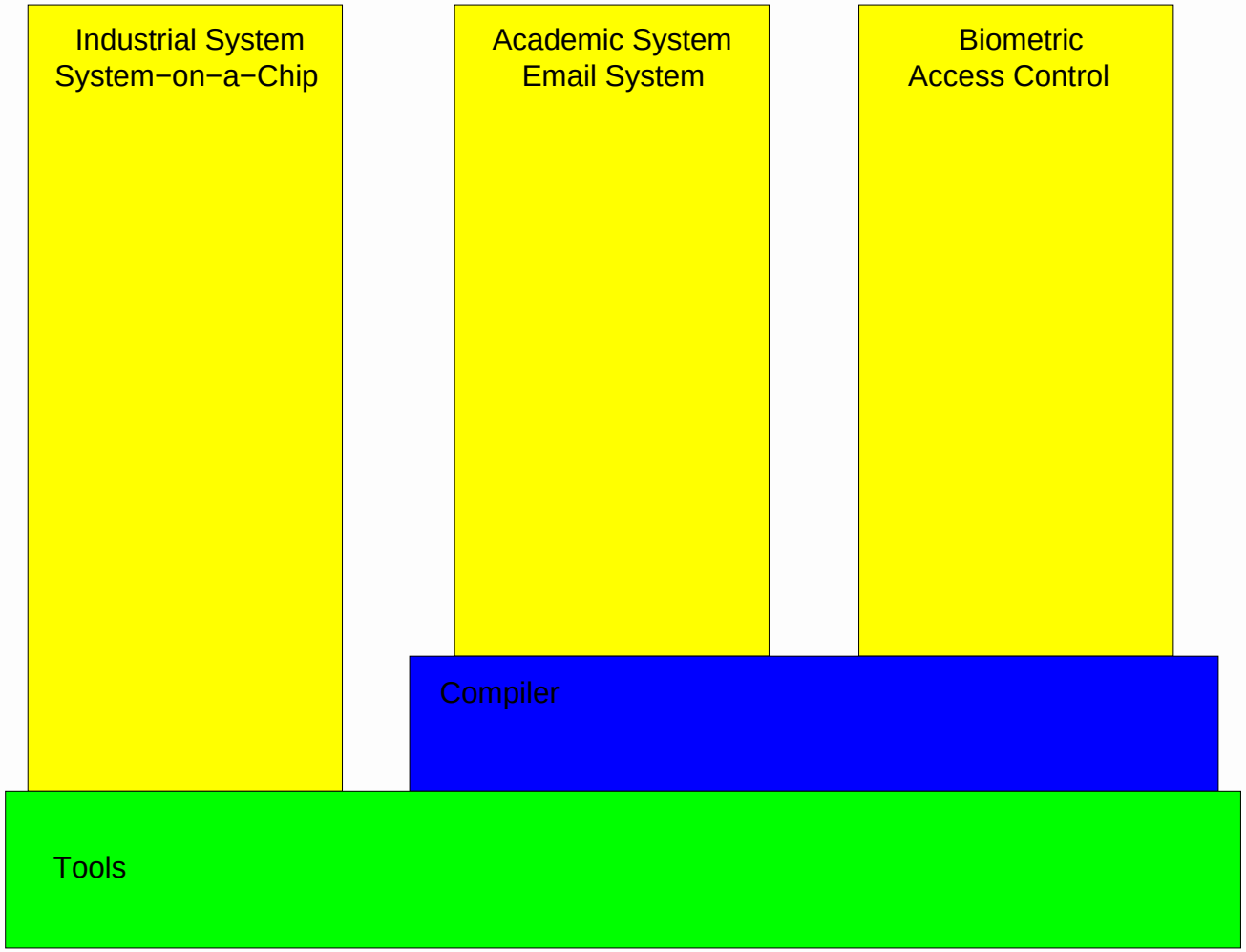
Part 1: Verisoft

Part 2: Formalizing Input and Output

Part 1: Verisoft

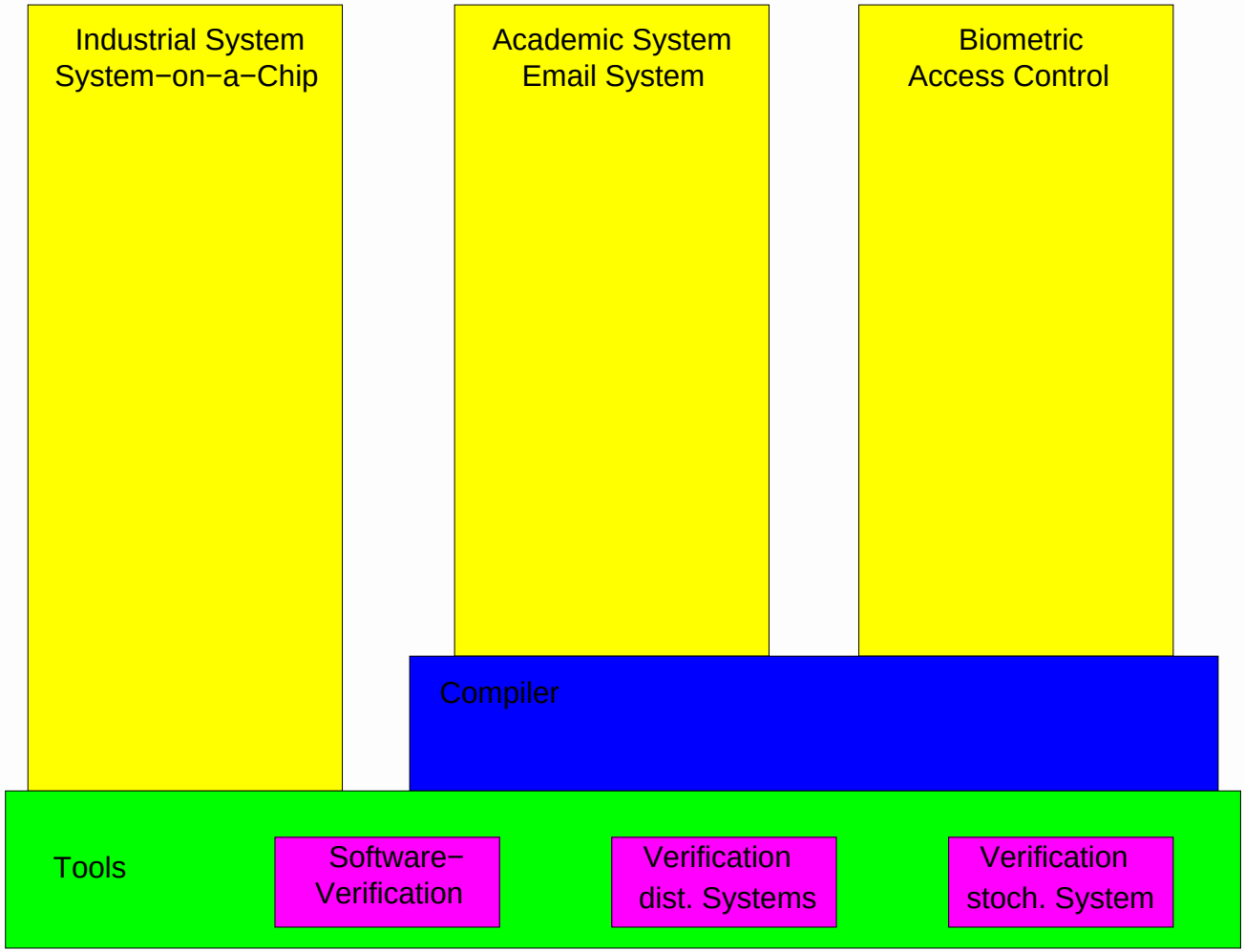
Verisoft — Project Goals

Completely verified systems:



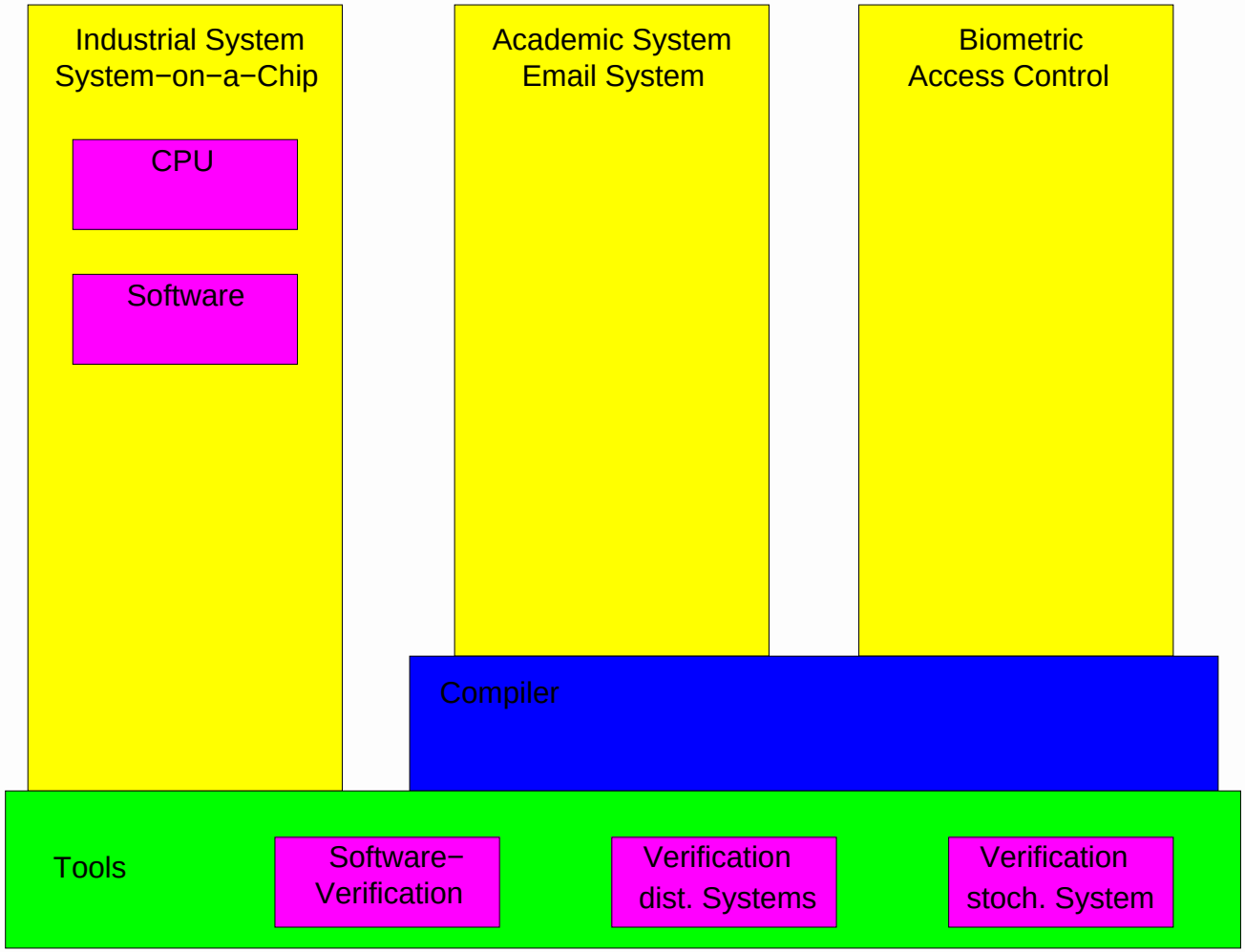
Verisoft — Project Goals

Completely verified systems:



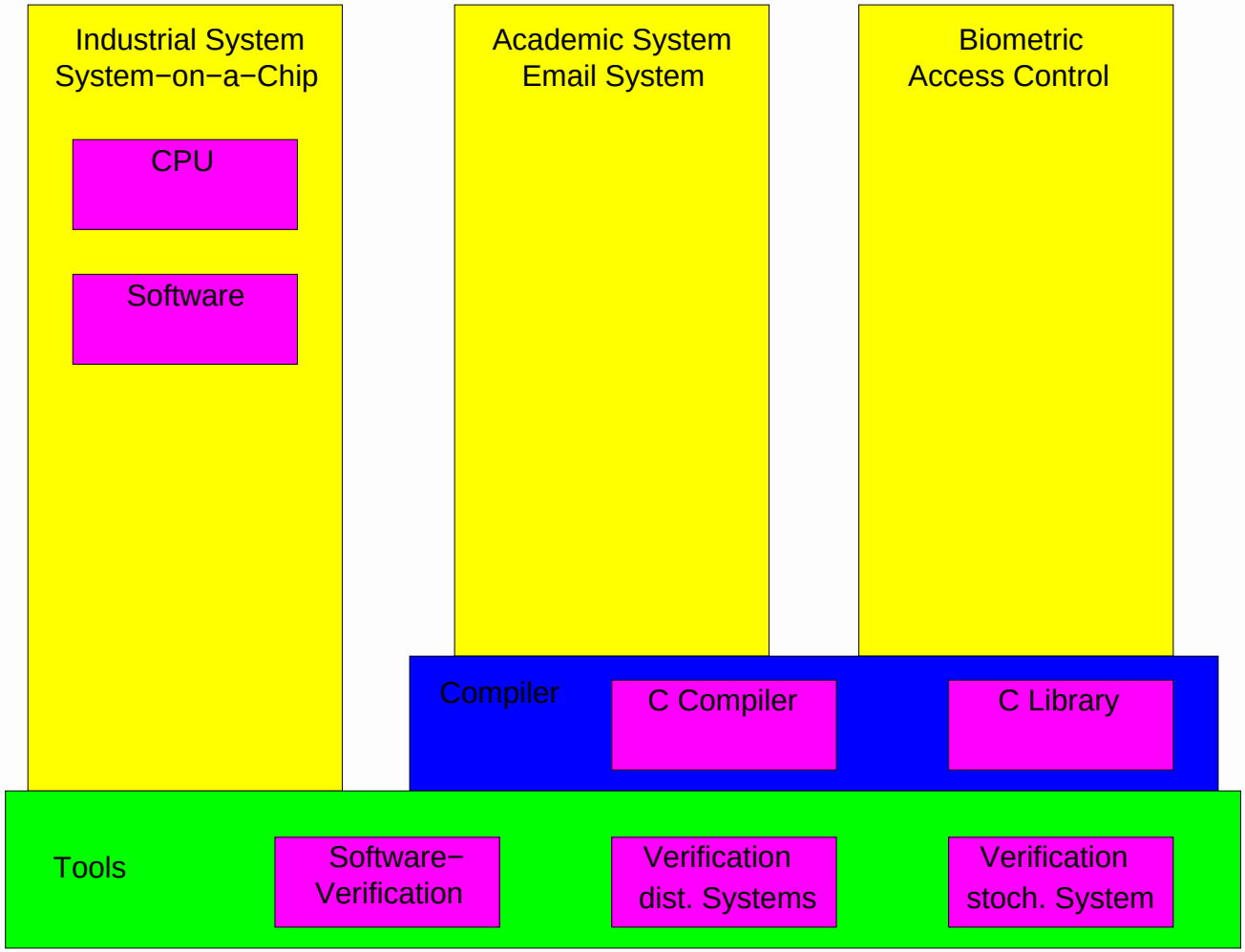
Verisoft — Project Goals

Completely verified systems:



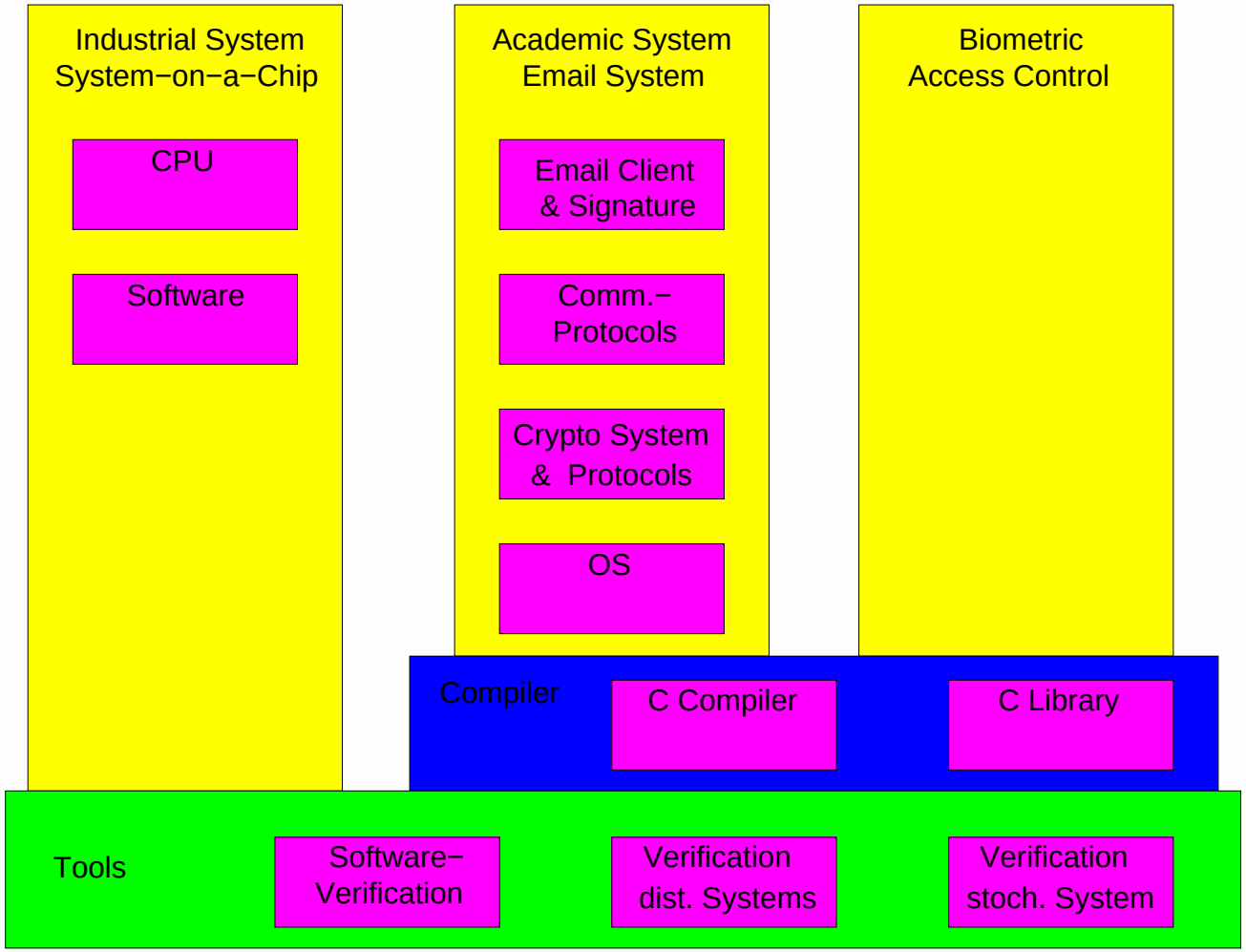
Verisoft — Project Goals

Completely verified systems:



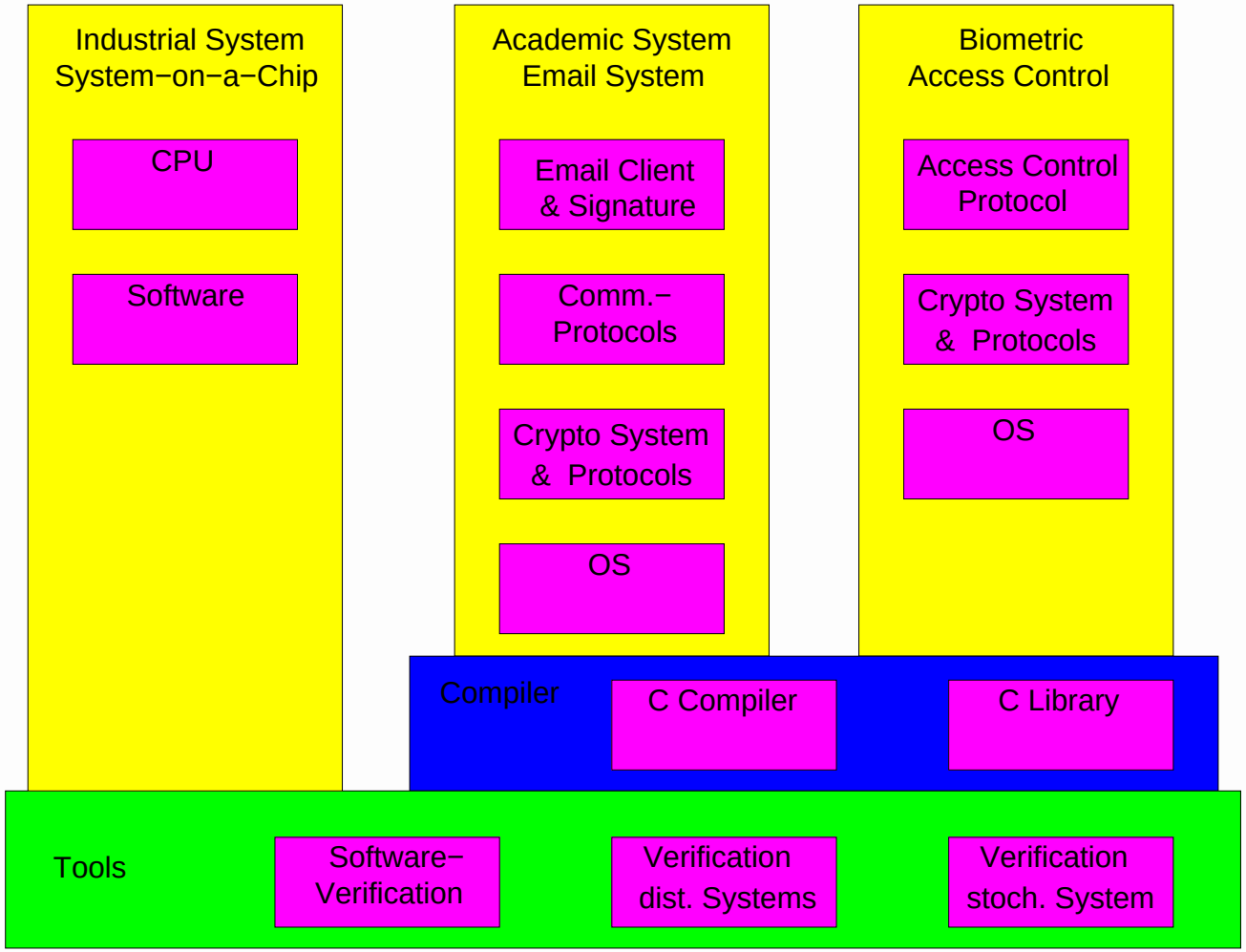
Verisoft — Project Goals

Completely verified systems:



Verisoft — Project Goals

Completely verified systems:



Email Client

Our part: A Simple Email Client

- Send & receive email via SMTP
- Sign email & check signatures
- Text based (ASCII)
- No folders (not even an Inbox)

Email Client—Screenshot

```
Keyboard locked by PID 57256 (bb)   |  Screen locked by PID 57256 (bb)
--------------------------------------------------------------------
From: "Gerd Beuster" <gb@uni-koblenz.de>
To: "Bernhard Beckert" <beckert@uni-koblenz.de>
Message-ID: <8d6701c3db02$76191724$cb29c023@uni-koblenz.de>
Subject: Verisoft-Spezifikation
X-Signature: SDLJK489342HJFVSFKJWQUI89237CFSDKJOIQ398LKSDFJSKLDJ
Date: Tue, 27 Jan 2004 16:53:48 +0100

Hallo Bernhard,

wir sollten uns mal ueber die Spezifikation des Email-Clients
unterhalten. Wann hast Du Zeit?

Gruesse,
Gerd

--------------------------------------------------------------------
Public Key:  DLFJDLSDCMVCDZ53DFDFJL9087/LDIEHJSDLFDJIOEJKLDST/GHSB2SLJ
Private Key: FDSLJF4O3489VNV XCKLJN3457896T87HSFDJVNS943ZFHFDIUSFHLA8V
--------------------------------------------------------------------
(s)end (p)oll | edit (m)ail p(u)b p(r)iv key | (a)dd (c)heck signature
--------------------------------------------------------------------
Last Cmd: Check Signature | Result: Signature valid  | Processing...
```

Email Client—Methods & Tools

Specification:

- Semi-formal specification in UML
- Formal specification in HOL-OCL
- Proofs in Isabelle

Part 2: Formalizing Input and Output

Security Aspects

Securing I/O against man-in-the-middle attacks

Security Aspects

Securing I/O against man-in-the-middle attacks

Security Aspects

Securing I/O against man-in-the-middle attacks

Software Attacks

- Locking screen & keyboard
- Providing information who locks the resource

```
Keyboard locked by PID 57256 (bb)   |  Screen locked by PID 57256 (bb)
------------------------------------------------------------------------
From: "Gerd Beuster" <gb@uni-koblenz.de>
To: "Bernhard Beckert" <beckert@uni-koblenz.de>
Message-ID: <8d6701c3db02$76191724$cb29c023@uni-koblenz.de>
Subject: Verisoft-Spezifikation
X-Signature: SDLJK489342HJFVSFKJWQUI89237CFSDKJOIQ398LKSDFJSKLDJ
Date: Tue, 27 Jan 2004 16:53:48 +0100

Hallo Bernhard,
```

The Boundary between Hard- and Software

In a text based application, input is a list of keystrokes,and output is a (multi-dimensional) list of characters.

The Boundary between Hard- and Software

In a text based application, input is a list of keystrokes,and output is a (multi-dimensional) list of characters.

- $keyboard$ = List of all keystrokes
- $keyboard(t)$ = List of all keystrokes received up to time t.
- $screenAt(t)[x, y]$ = The character shown at time t at screen position (x, y).

The Boundary between Hard- and Software

In a text based application, input is a list of keystrokes,and output is a (multi-dimensional) list of characters.

- $keyboard$ = List of all keystrokes
- $keyboard(t)$ = List of all keystrokes received up to time t.
- $screenAt(t)[x, y]$ = The character shown at time t at screen position (x, y).

$$screenAt(t) = f(keyboard(t))$$

Screen Up-To-Date

$screenAt(t)$ describes what's actually shown on the screen.

Screen Up-To-Date

$screenAt(t)$ describes what's actually shown on the screen.

$screenOutput(conf)$ describes what should be shown in a given system configuration. ("*observer*")

Screen Up-To-Date

$screenAt(t)$ describes what's actually shown on the screen.

$screenOutput(conf)$ describes what should be shown in a given system configuration. ("*observer*")

$\Rightarrow$ The screen is up-to-date if what we want to show ($screenOutput$) is identical to what is actually shown ($screenAt$).

Screen Up-To-Date

$screenAt(t)$ describes what's actually shown on the screen.

$screenOutput(conf)$ describes what should be shown in a given system configuration. ("*observer*")

$\Rightarrow$ The screen is up-to-date if what we want to show ($screenOutput$) is identical to what is actually shown ($screenAt$).

For security reasons, we also want to show who locks i/o ressources.

Constraints for Secure Systems

The display is correct (or up-to-date) at time t, if

$$\forall x, y : screenAt(t)[x, y] = screenOutput(conf(t))[x, y]$$

Constraints for Secure Systems

The display is correct (or up-to-date) at time t, if

$$\forall x, y : screenAt(t)[x, y] = screenOutput(conf(t))[x, y]$$

If resources are locked, this should be shown on the screen.

Constraints for Secure Systems

The display is correct (or up-to-date) at time t, if

$$\forall x, y : screenAt(t)[x, y] = screenOutput(conf(t))[x, y]$$

If resources are locked, this should be shown on the screen. $displayLocked(conf)$ provides information who locks the resources.

Constraints for Secure Systems

The display is correct (or up-to-date) at time t, if

$$\forall x, y : screenAt(t)[x, y] = screenOutput(conf(t))[x, y]$$

If resources are locked, this should be shown on the screen. $displayLocked(conf)$ provides information who locks the resources.

$$displayLocked(conf)[x] = screenOutput(conf)[x, 0]$$

Constraints for Secure Systems

The display is correct (or up-to-date) at time t, if

$$\forall x, y : screenAt(t)[x, y] = screenOutput(conf(t))[x, y]$$

If resources are locked, this should be shown on the screen. $displayLocked(conf)$ provides information who locks the resources.

$$displayLocked(conf)[x] = screenOutput(conf)[x, 0]$$

It is essential that only the operating system may change the area where this information is shown!

Conclusions

The method we introduced. . .

- . . . does not help against hardware based attacks.

Conclusions

The method we introduced. . .

- . . . does not help against hardware based attacks.
- . . . does not help against content based attacks.

Conclusions

The method we introduced. . .

- . . . does not help against hardware based attacks.
- . . . does not help against content based attacks.
- . . . does not guarantee that the output is perceived as intended.

Conclusions

The method we introduced. . .

- . . . does not help against hardware based attacks.
- . . . does not help against content based attacks.
- . . . does not guarantee that the output is perceived as intended.
- . . . *does* prevent software based attacks on i/o ressources.

Conclusions

The method we introduced. . .

- . . . does not help against hardware based attacks.
- . . . does not help against content based attacks.
- . . . does not guarantee that the output is perceived as intended.
- . . . *does* prevent software based attacks on i/o ressources.
- . . . can be applied to other i/o devices (card readers, graphical terminals,. . .)

Conclusions

The method we introduced. . .

- . . . does not help against hardware based attacks.
- . . . does not help against content based attacks.
- . . . does not guarantee that the output is perceived as intended.
- . . . *does* prevent software based attacks on i/o ressources.
- . . . can be applied to other i/o devices (card readers, graphical terminals,. . .)
- . . . requires special operating system functionality (locking of resources).

Summary and Future

Summary

- We gave a formalism for the description of text based input and output.
- We showed an effective counter-measure against certain types of man-in-the-middle attacks.

Summary and Future

Summary

- We gave a formalism for the description of text based input and output.
- We showed an effective counter-measure against certain types of man-in-the-middle attacks.

Future

- We will provide a methods for the formal specification of text based interactive applications, based on state charts.
- More email specific security issues will be addressed.