# CASE STUDY: THE PROJECTIT

In this case study, we present an example software system described in Wright/c. An illustration of the verification process applied to the system is also given by elaborating each step of the process.

## Wright/c Description of the ProjectIT

The software system supports data exchange, in the context of a project called *ProjectIT*, between a customer and a consortium formed by two companies: a software vendor that develops an application software for the customer, and a hardware vendor that supplies the hardware platform for the software.

Throughout project life cycle, information of various sensitivity flows between the vendors and the customer. The vendors establish interfaces to the customer to exchange project-specific information that may also flow between the vendors. Additionally, the companies set up a communication path for transferring project-specific information not to be shared with the customer. The topology of the architecture is depicted in Figure 1.

An access control lattice model, shown in Figure 2, introduces the security labels of data that flow within the project. The clearance that dominates a security label is shown in parentheses.

As depicted in Figure 1, the connections between the customer and the vendors are bidirectional and *ProjectWide* labeled data flows on these connections. Such connection is also established between the vendors.

Vendor-related information is exchanged between the partner vendors using two unidirectional connectors. Software vendor sends *SWSpecific* data through *SwHwConn* connector while the hardware vendor sends HW*Specific* data on *HwSwConn* connector.

The description of the lattice model and Wright/c description of ProjectIT configuration is presented in Figure 3 and Figure 4, respectively.

As shown in the Wright/c description, the ports are assigned suitable clearance to send and receive data in compliance with the BLP principles. For example, on the *HwCustConn* connection, only *ProjectWide* labeled data transfer is allowed. For instance, sending a vendor-specific data (labeled *HwSpecific*) through that connection causes a violation of the 'no read up' principle at the receiving side of the connection.

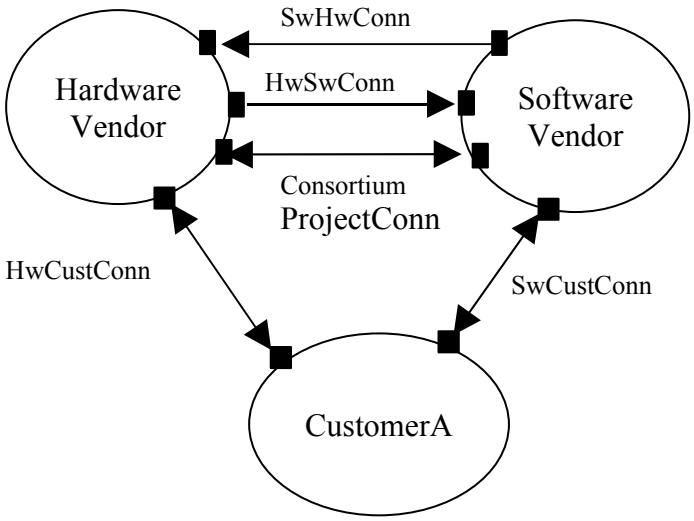


Figure 1: Topology of ProjectIT

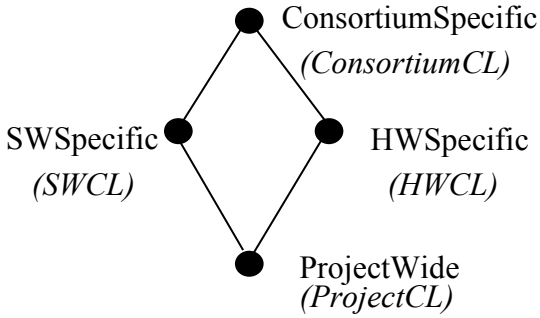


Figure 2: An access control lattice model for ProjectIT

```
Lattice PLM
  Security Labels
        ConsortiumSpecific, SWSpecific,
        HWSpecific, ProjectWide
  Ordering
       ProjectWide, SWSpecific, ConsortiumSpecific
       ProjectWide, HWSpecific, ConsortiumSpecific
  Clearance List
       ConsortiumCL  : ConsortiumSpecific
       HWCL          : HWSpecific
       SWCL          : SWSpecific
       ProjectCL     : ProjectWide
End Lattice
```

Figure 3: Wright/c description of the access control lattice model for ProjectIT

**Configuration ProjectIT**

**Import Lattice PLM "PITLattice.txt"**

**Component** Vendor($\tau$, $\mu$ : SecurityLabel) =

**Port** VendorSend = $\overline{\text{SendData! x}^{\tau}}$ $\rightarrow$ VendorSend

**Port** VendorReceive = ReceiveData?x $\rightarrow$ VendorReceive

**Port** VendorProject = $\overline{\text{SendData!x}^{\mu}} \rightarrow$ VendorProject

$\prod$ ReceiveData?x $\rightarrow$ VendorProject

**Port** CustomerProject= $\overline{\text{SendData!x}^{\mu}} \rightarrow$ CustomerProject

$\prod$ ReceiveData?x $\rightarrow$ CustomerProject

**Computation** = ( $\overline{\text{CustomerProject.SendData!x}^{\mu}} \rightarrow$ Computation

□ $\overline{\text{VendorSend.SendData!x}^{\tau}} \rightarrow$ Computation

□ $\overline{\text{VendorProject.SendData!x}^{\mu}} \rightarrow$ Computation)

$\prod$ CustomerProject. ReceiveData?x $\rightarrow$ Computation

$\prod$ VendorReceive. ReceiveData?x $\rightarrow$ Computation

$\prod$ VendorProject. ReceiveData?x $\rightarrow$ Computation

**Component** Customer(n:1..10; $\tau$ : SecurityLabel) =

**Port** $\text{VendorInterface}_{1..n}$ = ReceiveData?x $\rightarrow$ VendorInterface

$\prod$ $\overline{\text{SendData!x}^{\tau}} \rightarrow$ VendorInterface

**Computation** = ;i;(1..n) • ($\text{VendorInterface}_i$. ReceiveData?x $\rightarrow$

$\overline{\text{DoOwnJob}} \rightarrow$ Computation

□ $\overline{\text{VendorInterface}_i\text{.SendData!x}^{\tau}} \rightarrow$ Computation)

**Connector** BiDirectionalLink =

**Role** SideA = Receive!x $\rightarrow$ SideA $\prod$ $\overline{\text{Send?x}} \rightarrow$ SideA

**Role** SideB = Receive!x $\rightarrow$ SideB $\prod$ $\overline{\text{Send?x}} \rightarrow$ SideB

**Glue** = SideA. Receive!x $\rightarrow$ $\overline{\text{SideB.Send?x}} \rightarrow$ Glue

$\prod$ SideB. Receive!x $\rightarrow$ $\overline{\text{SideA.Send?x}} \rightarrow$ Glue

**Connector** UniDirectionalLink =

**Role** SideA = Receive!x $\rightarrow$ SideA

**Role** SideB = $\overline{\text{Send?x}} \rightarrow$ SideB

**Glue** = SideA. Receive!x $\rightarrow$ $\overline{\text{SideB.Send?x}} \rightarrow$ Glue

Figure 4: Wright/c description of ProjectIT configuration

**Instances**

SWVendor : Vendor(PLM.SWSpecific, PLM.ProjectWide)
HWVendor : Vendor(PLM.HWSpecific, PLM. ProjectWide)
CustomerA : Customer(2, PLM. ProjectWide)
SwHwConn, HwSwConn : UniDirectionalLink
HwCustomerConn, SwCustomerConn,
ConsortiumProjectConn : BiDirectionalLink

**Clearance**

SWVendor.VendorSend : SWCL
HWVendor.VendorReceive : ConsortiumCL
HWVendor.VendorSend : HWCL
SWVendor.VendorReceive : ConsortiumCL
SWVendor.CustomerProject, HWVendor.CustomerProject : ProjectCL
SWVendor.VendorProject, HWVendor.VendorProject : ProjectCL
CustomerA : ProjectCL // all ports of CustomerA

**Attachments**

SWVendor.VendorSend As SwHwConn.SideA
HWVendor.VendorReceiveAs SwHwConn.SideB
HWVendor.VendorSend As HwSwConn.SideA
SWVendor.VendorReceive As HwSwConn.SideB
SWVendor.VendorProject As ConsortiumProjectConn.SideA
HWVendor.VendorProject As ConsortiumProjectConn.SideB
SWVendor.CustomerProject As SwCustomerConn.SideA
CustomerA. $VendorInterface_1$ As SwCustomerConn.SideB
HWVendor.CustomerProject As HwCustomerConn.SideA
CustomerA. $VendorInterface_2$ As HwCustomerConn.SideB

**End Configuration**

Figure 4: Wright/c description of ProjectIT configuration (continued)

## Verification of the ProjectIT System

This section gives an illustration of the verification process applied to the ProjectIT system. The verification starts by invoking a call to the `verify` function in ML run-time environment as:

- `verify(configuration,style);`

The output of the function is the report including the information for each port of every component instance in the configuration. The contents of the *rlsa*, the *slsa*, and the labels refused due to violation prevention are displayed after each iteration step of the verification process. In the report, the labels are represented in the form of sublattices. Moreover, for each component instance that lowers security labels of its input data in its computation, a warning message is produced saying that the component must be trusted.

The following is the output of the verification process after the iteration 0 and iteration 1 where the stable state is reached. The warning messages for trustworthiness of `SWVendor` and `HWVendor` are reported in the first iteration since all security labels are offered to the ports of these components but, for example, `ConsortiumSpecific` labeled data are not output.

```
      Warning !...SWVendor MUST BE TRUSTED!...

      Warning !...HWVendor MUST BE TRUSTED!...
 ****************************
Iteration No: 0
RECEIVED LIST (0)
Component.Port: SWVendor.VendorSend  clearance:SWCL
   Allowed Security Labels (min,max): (ProjectWide,ConsortiumSpecific)
   Refused Security Labels (min,max):

Component.Port: SWVendor.VendorReceive  clearance:ConsortiumCL
   Allowed Security Labels (min,max): (ProjectWide,ConsortiumSpecific)
   Refused Security Labels (min,max):

Component.Port: SWVendor.VendorProject  clearance:ProjectCL
   Allowed Security Labels (min,max): (ProjectWide,ConsortiumSpecific)
   Refused Security Labels (min,max):

Component.Port: SWVendor.CustomerProject  clearance:ProjectCL
   Allowed Security Labels (min,max): (ProjectWide,ConsortiumSpecific)
   Refused Security Labels (min,max):

Component.Port: HWVendor.VendorSend  clearance:HWCL
   Allowed Security Labels (min,max): (ProjectWide,ConsortiumSpecific)
   Refused Security Labels (min,max):

Component.Port: HWVendor.VendorReceive  clearance:ConsortiumCL
   Allowed Security Labels (min,max): (ProjectWide,ConsortiumSpecific)
   Refused Security Labels (min,max):

Component.Port: HWVendor.VendorProject  clearance:ProjectCL
   Allowed Security Labels (min,max): (ProjectWide,ConsortiumSpecific)
   Refused Security Labels (min,max):

Component.Port: HWVendor.CustomerProject  clearance:ProjectCL
   Allowed Security Labels (min,max): (ProjectWide,ConsortiumSpecific)
   Refused Security Labels (min,max):

Component.Port: CustomerA.VendorInterface_1  clearance:ProjectCL
   Allowed Security Labels (min,max): (ProjectWide,ConsortiumSpecific)
   Refused Security Labels (min,max):
```

```
Component.Port: CustomerA.VendorInterface_2  clearance:ProjectCL
   Allowed Security Labels (min,max): (ProjectWide,ConsortiumSpecific)
   Refused Security Labels (min,max):

----------------------------------------------------------------------

SENT LIST (0)
Component.Port: SWVendor.VendorSend  type: OUTPUT_PORT  clearance:SWCL
   Allowed Security Labels (min,max): (SWSpecific,SWSpecific)
   Refused Security Labels (min,max):

Component.Port: SWVendor.VendorReceive  type: INPUT_PORT
clearance:ConsortiumCL

Component.Port: SWVendor.VendorProject  type: INPUTOUTPUT_PORT
clearance:ProjectCL
   Allowed Security Labels (min,max): (ProjectWide,ProjectWide)
   Refused Security Labels (min,max):

Component.Port: SWVendor.CustomerProject  type: INPUTOUTPUT_PORT
clearance:ProjectCL
   Allowed Security Labels (min,max): (ProjectWide,ProjectWide)
   Refused Security Labels (min,max):

Component.Port: HWVendor.VendorSend  type: OUTPUT_PORT  clearance:HWCL
   Allowed Security Labels (min,max): (HWSpecific,HWSpecific)
   Refused Security Labels (min,max):

Component.Port: HWVendor.VendorReceive  type: INPUT_PORT
clearance:ConsortiumCL

Component.Port: HWVendor.VendorProject  type: INPUTOUTPUT_PORT
clearance:ProjectCL
   Allowed Security Labels (min,max): (ProjectWide,ProjectWide)
   Refused Security Labels (min,max):

Component.Port: HWVendor.CustomerProject  type: INPUTOUTPUT_PORT
clearance:ProjectCL
   Allowed Security Labels (min,max): (ProjectWide,ProjectWide)
   Refused Security Labels (min,max):

Component.Port: CustomerA.VendorInterface_1  type: INPUTOUTPUT_PORT
clearance:ProjectCL
   Allowed Security Labels (min,max): (ProjectWide,ProjectWide)
   Refused Security Labels (min,max):

Component.Port: CustomerA.VendorInterface_2  type: INPUTOUTPUT_PORT
clearance:ProjectCL
   Allowed Security Labels (min,max): (ProjectWide,ProjectWide)
   Refused Security Labels (min,max):

**************************************************************
Iteration No: 1
RECEIVED LIST (1)
Component.Port: SWVendor.VendorSend  clearance:SWCL

Component.Port: SWVendor.VendorReceive  clearance:ConsortiumCL
   Allowed Security Labels (min,max): (HWSpecific,HWSpecific)
   Refused Security Labels (min,max):

Component.Port: SWVendor.VendorProject  clearance:ProjectCL
   Allowed Security Labels (min,max): (ProjectWide,ProjectWide)
   Refused Security Labels (min,max):

Component.Port: SWVendor.CustomerProject  clearance:ProjectCL
   Allowed Security Labels (min,max): (ProjectWide,ProjectWide)
   Refused Security Labels (min,max):
```

```
Component.Port: HWVendor.VendorSend  clearance:HWCL

Component.Port: HWVendor.VendorReceive  clearance:ConsortiumCL
   Allowed Security Labels (min,max): (SWSpecific,SWSpecific)
   Refused Security Labels (min,max):

Component.Port: HWVendor.VendorProject  clearance:ProjectCL
   Allowed Security Labels (min,max): (ProjectWide,ProjectWide)
   Refused Security Labels (min,max):

Component.Port: HWVendor.CustomerProject  clearance:ProjectCL
   Allowed Security Labels (min,max): (ProjectWide,ProjectWide)
   Refused Security Labels (min,max):

Component.Port: CustomerA.VendorInterface_1  clearance:ProjectCL
   Allowed Security Labels (min,max): (ProjectWide,ProjectWide)
   Refused Security Labels (min,max):

Component.Port: CustomerA.VendorInterface_2  clearance:ProjectCL
   Allowed Security Labels (min,max): (ProjectWide,ProjectWide)
   Refused Security Labels (min,max):

---------------------------------------------------------------------

SENT LIST (1)
Component.Port: SWVendor.VendorSend  type: OUTPUT_PORT  clearance:SWCL
   Allowed Security Labels (min,max): (SWSpecific,SWSpecific)
   Refused Security Labels (min,max):

Component.Port: SWVendor.VendorReceive  type: INPUT_PORT
clearance:ConsortiumCL

Component.Port: SWVendor.VendorProject  type: INPUTOUTPUT_PORT
clearance:ProjectCL
   Allowed Security Labels (min,max): (ProjectWide,ProjectWide)
   Refused Security Labels (min,max):

Component.Port: SWVendor.CustomerProject  type: INPUTOUTPUT_PORT
clearance:ProjectCL
   Allowed Security Labels (min,max): (ProjectWide,ProjectWide)
   Refused Security Labels (min,max):

Component.Port: HWVendor.VendorSend  type: OUTPUT_PORT  clearance:HWCL
   Allowed Security Labels (min,max): (HWSpecific,HWSpecific)
   Refused Security Labels (min,max):

Component.Port: HWVendor.VendorReceive  type: INPUT_PORT
clearance:ConsortiumCL

Component.Port: HWVendor.VendorProject  type: INPUTOUTPUT_PORT
clearance:ProjectCL
   Allowed Security Labels (min,max): (ProjectWide,ProjectWide)
   Refused Security Labels (min,max):

Component.Port: HWVendor.CustomerProject  type: INPUTOUTPUT_PORT
clearance:ProjectCL
   Allowed Security Labels (min,max): (ProjectWide,ProjectWide)
   Refused Security Labels (min,max):

Component.Port: CustomerA.VendorInterface_1  type: INPUTOUTPUT_PORT
clearance:ProjectCL
   Allowed Security Labels (min,max): (ProjectWide,ProjectWide)
   Refused Security Labels (min,max):

Component.Port: CustomerA.VendorInterface_2  type: INPUTOUTPUT_PORT
clearance:ProjectCL
   Allowed Security Labels (min,max): (ProjectWide,ProjectWide)
```

```
   Refused Security Labels (min,max):

****************************************************************
```

Next, the stable *rlsa* and *slsa* are reported including the same type of information as follows:

```
STABLE RECEIVED LIST (1)
Component.Port: SWVendor.VendorSend  clearance:SWCL

Component.Port: SWVendor.VendorReceive  clearance:ConsortiumCL
   Allowed Security Labels (min,max): (HWSpecific,HWSpecific)
   Refused Security Labels (min,max):

Component.Port: SWVendor.VendorProject  clearance:ProjectCL
   Allowed Security Labels (min,max): (ProjectWide,ProjectWide)
   Refused Security Labels (min,max):

Component.Port: SWVendor.CustomerProject  clearance:ProjectCL
   Allowed Security Labels (min,max): (ProjectWide,ProjectWide)
   Refused Security Labels (min,max):

Component.Port: HWVendor.VendorSend  clearance:HWCL

Component.Port: HWVendor.VendorReceive  clearance:ConsortiumCL
   Allowed Security Labels (min,max): (SWSpecific,SWSpecific)
   Refused Security Labels (min,max):

Component.Port: HWVendor.VendorProject  clearance:ProjectCL
   Allowed Security Labels (min,max): (ProjectWide,ProjectWide)
   Refused Security Labels (min,max):

Component.Port: HWVendor.CustomerProject  clearance:ProjectCL
   Allowed Security Labels (min,max): (ProjectWide,ProjectWide)
   Refused Security Labels (min,max):

Component.Port: CustomerA.VendorInterface_1  clearance:ProjectCL
   Allowed Security Labels (min,max): (ProjectWide,ProjectWide)
   Refused Security Labels (min,max):

Component.Port: CustomerA.VendorInterface_2  clearance:ProjectCL
   Allowed Security Labels (min,max): (ProjectWide,ProjectWide)
   Refused Security Labels (min,max):

----------------------------------------------------------------------

STABLE SENT LIST (1)
Component.Port: SWVendor.VendorSend  type: OUTPUT_PORT  clearance:SWCL
   Allowed Security Labels (min,max): (SWSpecific,SWSpecific)
   Refused Security Labels (min,max):

Component.Port: SWVendor.VendorReceive  type: INPUT_PORT
clearance:ConsortiumCL

Component.Port: SWVendor.VendorProject  type: INPUTOUTPUT_PORT
clearance:ProjectCL
   Allowed Security Labels (min,max): (ProjectWide,ProjectWide)
   Refused Security Labels (min,max):

Component.Port: SWVendor.CustomerProject  type: INPUTOUTPUT_PORT
clearance:ProjectCL
   Allowed Security Labels (min,max): (ProjectWide,ProjectWide)
   Refused Security Labels (min,max):

Component.Port: HWVendor.VendorSend  type: OUTPUT_PORT  clearance:HWCL
   Allowed Security Labels (min,max): (HWSpecific,HWSpecific)
```

```
   Refused Security Labels (min,max):

Component.Port: HWVendor.VendorReceive  type: INPUT_PORT
clearance:ConsortiumCL

Component.Port: HWVendor.VendorProject  type: INPUTOUTPUT_PORT
clearance:ProjectCL
   Allowed Security Labels (min,max): (ProjectWide,ProjectWide)
   Refused Security Labels (min,max):

Component.Port: HWVendor.CustomerProject  type: INPUTOUTPUT_PORT
clearance:ProjectCL
   Allowed Security Labels (min,max): (ProjectWide,ProjectWide)
   Refused Security Labels (min,max):

Component.Port: CustomerA.VendorInterface_1  type: INPUTOUTPUT_PORT
clearance:ProjectCL
   Allowed Security Labels (min,max): (ProjectWide,ProjectWide)
   Refused Security Labels (min,max):

Component.Port: CustomerA.VendorInterface_2  type: INPUTOUTPUT_PORT
clearance:ProjectCL
   Allowed Security Labels (min,max): (ProjectWide,ProjectWide)
   Refused Security Labels (min,max):
```

Then, a verification report combining the *rlsa* and *slsa* information is output. For each port of every component instance, the type of the port (INPUT, OUTPUT, INPUTOUTPUT), its clearance, and potentially input and output data security labels are included in the report. The potentially input (output) data security labels and the potentially input data security labels will be '`No data...`' when the port is an output (input) port. If refused security label list has some entries in the stable *rlsa* or *slsa*, a potential anomaly notification is also produced. If no such anomalies are detected, a success notification message is displayed as shown below.

```
VERIFICATION REPORT
************************************************************
Component.Port: SWVendor.VendorSend  type :OUTPUT_PORT  clearance:SWCL
  potentially output data security labels: SWSpecific
  potentially input  data security labels:  No data...

Component.Port: SWVendor.VendorReceive  type :INPUT_PORT
clearance:ConsortiumCL

  potentially output data security labels:  No data...
  potentially input  data security labels: HWSpecific

Component.Port: SWVendor.VendorProject  type :INPUTOUTPUT_PORT
clearance:ProjectCL
  potentially output data security labels: ProjectWide
  potentially input  data security labels: ProjectWide

Component.Port: SWVendor.CustomerProject  type :INPUTOUTPUT_PORT
clearance:ProjectCL
  potentially output data security labels: ProjectWide
  potentially input  data security labels: ProjectWide

Component.Port: HWVendor.VendorSend  type :OUTPUT_PORT  clearance:HWCL
  potentially output data security labels: HWSpecific
```

```
  potentially input  data security labels:  No data...

Component.Port: HWVendor.VendorReceive  type :INPUT_PORT
clearance:ConsortiumCL

  potentially output data security labels:  No data...
  potentially input  data security labels: SWSpecific

Component.Port: HWVendor.VendorProject  type :INPUTOUTPUT_PORT
clearance:ProjectCL
  potentially output data security labels: ProjectWide
  potentially input  data security labels: ProjectWide

Component.Port: HWVendor.CustomerProject  type :INPUTOUTPUT_PORT
clearance:ProjectCL
  potentially output data security labels: ProjectWide
  potentially input  data security labels: ProjectWide

Component.Port: CustomerA.VendorInterface_1  type :INPUTOUTPUT_PORT
clearance:ProjectCL
  potentially output data security labels: ProjectWide
  potentially input  data security labels: ProjectWide

Component.Port: CustomerA.VendorInterface_2  type :INPUTOUTPUT_PORT
clearance:ProjectCL
  potentially output data security labels: ProjectWide
  potentially input  data security labels: ProjectWide

************ The verification is SUCCESSFUL ************
```

Lastly, the report displays excess privileges if any is detected during the verification process, and recommends a new (minimum) clearance in place of currently assigned. In our case, the `HWCL` and `SWCL` the `VendorReceive` ports of `SWVendor` and `HWVendor` are recommended, respectively, although `ConsortiumCL` was originally assigned to both of them.

```
 EXCESS PRIVILEGES
 *****************

Excess privilege for SWVendor.VendorReceive found:
   Current: ConsortiumCL Recommended: HWCL
Excess privilege for HWVendor.VendorReceive found:
   Current: ConsortiumCL Recommended: SWCL

WARNING : Some excessive privileges are associated with ports as given
above!..

 Please check them and revise your system configuration...
```