PROGRAM PLC\_PRG

VAR

CODE5 : Code5 ; //instance of Code5

WGUST: LREAL;

QGUST: LREAL;

END\_VAR

WGUST := 50;

QGUST := 50 ;

CODE5 (wGust:= WGUST, qGust := QGUST);

FUNCTION\_BLOCK Code5

VAR\_INPUT

wGust: LREAL;

qGust: LREAL;

END\_VAR

VAR\_OUTPUT

Out1: LREAL;

Out2: LREAL;

END\_VAR

VAR

//vulnerability variables

Index1 : LREAL ;

Index2 : DINT ;

Array\_ : ARRAY [1..100] OF LREAL:= [100(100)]; //Array\_ starts with 100 in all 100 spots

bActivateTimer : BOOL := 0 ; // checks if timer activated

fbTimer : TON ; // instance of function block TON

Index\_Time : TIME ;

END\_VAR

Index1 := ((wGust)\*(qGust));

Index2 := LREAL\_TO\_DINT (Index1);

IF Index2 < 0 THEN

Array\_[Index2] := Index1; //here vulnerability with data types as well

bActivateTimer := 1 ;

Index\_Time := DINT\_TO\_TIME (Index2) ;

fbTimer (IN := TRUE, PT := Index\_Time) ; //will return true after timer off

ELSIF Index2 > 0 AND Index2 < 1 THEN

Index2 := Index2 \* 100 ;

Array\_[Index2] := Index1; //here vulnerability with data types as well

bActivateTimer := 1 ;

Index\_Time := DINT\_TO\_TIME (Index2) ;

fbTimer (IN := TRUE, PT := Index\_Time) ;

ELSIF Index2 > 1 AND Index2 < 100 THEN

Index2 := (Index2)\* 1000;

Array\_[Index2] := Index1; //here vulnerability with data types as well

bActivateTimer := 1 ;

Index\_Time := DINT\_TO\_TIME (Index2) ;

fbTimer (IN := TRUE, PT := Index\_Time) ;

ELSIF Index2 > 100 AND Index2 < 10000 THEN

Index2 := Index2 \* Index2 ;

Array\_[Index2] := Index1; //here vulnerability with data types as well

bActivateTimer := 1 ;

Index\_Time := DINT\_TO\_TIME (Index2) ;

fbTimer (IN := TRUE, PT := Index\_Time) ;

ELSE

Index2 := Index2 \* Index2 \* Index2 ;

Array\_[Index2] := Index1; //here vulnerability with data types as well

bActivateTimer := 1 ;

Index\_Time := DINT\_TO\_TIME (Index2) ;

fbTimer (IN := TRUE, PT := Index\_Time) ;

END\_IF;

bActivateTimer := 0 ; //timer off

IF fbTimer.Q THEN //move to the rest of the code

(\* Outputs for Atomic SubSystem: '<Root>/Code 2' \*)

(\* Outport: '<Root>/Out1' incorporates:

\* Gain: '<S1>/Gain' \*)

Out1 := -0.6385 \* wGust;

(\* Outport: '<Root>/Out2' incorporates:

\* Gain: '<S1>/Gain1'

\* Gain: '<S1>/Gain2'

\* Sum: '<S1>/Sum' \*)

Out2 := (-0.00592 \* wGust) + (-0.6571 \* qGust);

(\* End of Outputs for SubSystem: '<Root>/Code 2' \*)

END\_IF ;