Preprints of the 2013 IFAC Conference on Manufacturing Modelling, Management, and Control, Saint Petersburg State University and Saint Petersburg National Research University of Information Technologies, Mechanics, and Optics, Saint Petersburg, Russia, June 19-21, 2013

A Preliminary Report on Static Analysis of C Code for Nuclear Reactor Protection System

Jong-Hoon Lee* , Eui-Sub Kim* , Junbeom Yoo* , Jang-Soo Lee**

**Konkuk Univeristy*
1 Hwayang-dong Gwangjin-gu Republic of Korea (e-mail: {kirdess, atang34, jbyoo}@konkuk.ac.kr)
***Korea Atomic Energy Research Institute (KAERI)*
989-111 Deadeok-daero Yuseong-gu Daejeon Republic of Korea (e-mail: jslee@kaeri.re.kr)

Abstract: Cybersecurity regulations require new I&C (Instrumentation & Control) systems in nuclear power plants to develop software in accordance with secure software development methodology to prevent the digital systems from cyber attacks. One of the common aspects of various secure software development methodologies is that widely-accepted practices should be followed throughout programming. As PLC (Programmable Logic Controller) is used to implement digital I&Cs, C programs are often translated automatically from design specifications such as FBD programs. This paper tries to analyze a part of preliminary version of C codes of a Korean I&C system with a static source code analysis tool of Microsoft. It shows that the automatic translator from FBD to C had a few critical defects, not concerned with security directly. It also recommends to select appropriate analysis tools and rule sets to check best practices in secure programming, even if the C code is produced mechanically.

Keywords: Nuclear Plant Protection System , I&C , PLC software , FBC-to-C translator , Static analysis ,

1. INTRODUCTION

Cybersecurity (International Electrotechnical Commission 2011) in nuclear I&C systems seeks to prevent unauthorized malicious accesses to information, software and data. It specifically recommends that designers and developers of I&C systems shall have established and verified secure development methodologies in place throughout the development lifecycle of a system. US NRC's cybersecurity regulations (U.S. Nuclear Regulatory Commission 2010) also require each nuclear power plant to submit a cybersecurity plan and implementation schedule (Andrewa 2012).

An I&C system in nuclear power plants consists of various safety and non-safety components. This paper tries to apply a secure software development methodology into RPS (Reactor Protection System), which is the most safety-critical component. As RPS makes decisions for emergent reactor shutdown, its correctness and functional safety should be verified throughout entire software/hardware development life-cycles.

RPS is typically implemented with safety-level PLCs, while RPS software is developed according to software development and verification processes. RPS software is first modeled with IEC-61131 (International Electrotechnical Commission 2013) FBD (Function Block Diagram) (International Electrotechnical Commission 2013) in design phase. In implementation, the FBD programs are translated into C programs and then compiled into executable machine code for the RPS hardware - PLC. Compiler expert companies typically provide C compilers which functional correctness was thoroughly verified and demonstrated. On the other hand, translators from FBDs to C programs are usually developed by PLC vendors own. They should demonstrate correctness and functional safety (Commission 2000) of the so-called '*FBD-to-C*' translator, sufficiently.

From the aspect of safety, safety-level PLC vendors have successfully demonstrated functional safety of RPS software. However, there has been no explicit consideration on security. As recommended by regulations and guidelines, we need to apply secure software development methodology into the RPS software too. In the area of secure software development, a number of methodologies (Ramsbrock 2010) have been proposed and used successfully. They all have own features and advantages, and it must be an interesting research to select one specific methodology appropriate for the RPS software development.

One of the common aspects of several secure software development methodologies is that widely-accepted practices should be followed throughout programming. Use of static and dynamic code analysis tools is also highly recommended as explained in (SAFECode 2008). Security issues in C code in the process of the RPS software development is especially important and crucial, since C code is mechanically translated from FBD/LD designs, not manually. Static code analysis on the mechanically-translated C code has often been skipped from the aspect of function safety and correctness. However, we need to check it against security issues as well as safety, additionally.

This paper tries to use a static code analysis tool of Microsoft (Microsoft n.d.) against a part of C code, which was generated mechanically from a preliminary version of FBD programs for APR-1400 RPS in Korea (Korea Atomic Energy Research Institute 2006). The static code analysis found a few critical defects in the premature C code (*i.e.,* not the official and finial version), but not directly related with security. Nevertheless we could conclude that more considerate application of static code analysis on the mechanically translated C codes is highly recommended.

The paper is organized as follows: Section 2 shares background of this paper, such as international standards and regulations. Section 3 explains a typical PLC-based RPS software development process in detail. Section 4 shows the result of static analysis, and Section 5 concludes the paper and remarks on future research expansion.

2. BACKGROUND

Cyber attacks in the nuclear industry has already launched in Bushehr nuclear power plant in Iran. The Stuxnet worm virus (Stuxnet worm hits Iran nuclear plant staff computers 2010) infiltrated PLCs in the power plant, and the virus tried to reprogram code of PLCs to get control of them. This case made the concerns for cybersecurity of nuclear industry greatly increasing, and the guidance documents and regulations have provided various requirements concerning cybersecurity.

US NRC regulatory guide 5.71 (U.S. Nuclear Regulatory Commission 2010) provides guidance to applicants and licensees on satisfying the requirements of 10 CFR 73.54 - "Protection of Digital Computer and Communication Systems and Networks" (U.S. Nuclear Regulatory Commission 2009). The CFR requires nuclear power palnt licenses in US to protect digital computer and communications systems and networks associated with following categories of functions, from cyber attacks: safety-related and important-to-safety functions, security functions, emergency preparedness functions, including offsite communications, and support systems and equipment which, if compromised, would adversely impact safety, security, or emergency preparedness functions.

IEC 61513, "Nuclear Power Plants – Instrumentation and control important to safety – General requirements for systems" provides general requirements for I&C systems and equipment that are used to perform functions important to safety in nuclear power plants. It is top-level document of the IEC SC45A standard series. IEC 62645, "Nuclear Power Plants – Instrumentation and control systems – Requirements for security programmes for computer-based systems" (International Electrotechnical Commission 2011) is the second level IEC SC45A document tackling the generic issue of cybersecurity. IEC 62645 standard specifically focuses on the issue of requirements for computer security programs and system development processes to prevent and/or minimize the impact of cyber attacks against computer-based systems.

IEC SC45A/894/NP is a proposal for a new standard, "Nuclear Power Plants – Instrumentation and control systems – Requirements for coordinating safety and cybersecurity". It is a new work item within IEC SC45A standard series, and related with IEC 62645. It focuses on a rigorous framework for I&C systems, to master the interactions and potential side-effects when safety and cybersecurity provisions converge on the same I&C systems.

Regulatory Guide 1.152, Revision 3 (U.S. Nuclear Regulatory Commission 2011) contains regulatory criteria on the establishment of a secure development and operational environment for digital safety systems. The establishment of a Secure Development and Operational Environment (SDOE) refers to: (1) measures and controls taken to establish a secure environment for development of the digital safety system against undocumented, unneeded and unwanted modifications and (2) protective actions taken against a predictable set of undesirable acts (e.g., inadvertent operator actions or the undesirable behavior of connected systems) that could challenge the integrity, reliability, or functionality of a digital safety system during operations. These SDOE actions may include adoption of protective design features into the digital safety system design to preclude inadvertent access to the system and/or protection against undesirable behavior from connected systems when operational.

IEC 60880, "Nuclear Power Plants – Instrumentation and control systems important to safety – Software aspects for computer-based systems performing category A functions" provides requirements for I&C software that are used to perform functions to safety in nuclear power plants. It also concerns about security issue, and reference to IEC 61513.

3. THE RPS SOFTWARE DEVELOPMENT PROCESS

An RPS is a real-time embedded system implemented on a number of PLCs. The RPS software is designed in FBD/LD languages and then translated into C programs which will be compiled and loaded on PLCs. Fig. explains a typical software development process for RPSs based on IEC 60880 software safety lifecycle (International Electrotechnical Commission 2006).

SRS (Software Requirements Specification) is first written in natural languages or formal specification languages (Labaw 1996) (Yoo, et al. 2005). Experts on PLC programming languages then translate the requirements specification into design models programmed in FBD or LD, manually. PLC vendors such as *AREVA*, *invensys* and *POSCO ICT* provide their own automatic translators from the FBD/LD programs into ANSI C programs (ISO/IEC 2011), while typically use COTS (Commercial Off-the-Shelf) software such as `*TMS320C55x*' of *Texas Instruments* (TEXAS INSTRUMENTS 2003) for the C compilers. The COTS compilers were well verified and certified enough to be used without additional verification effort. However, the vendor-provided automatic translators should demonstrate functional correctness rigorously.

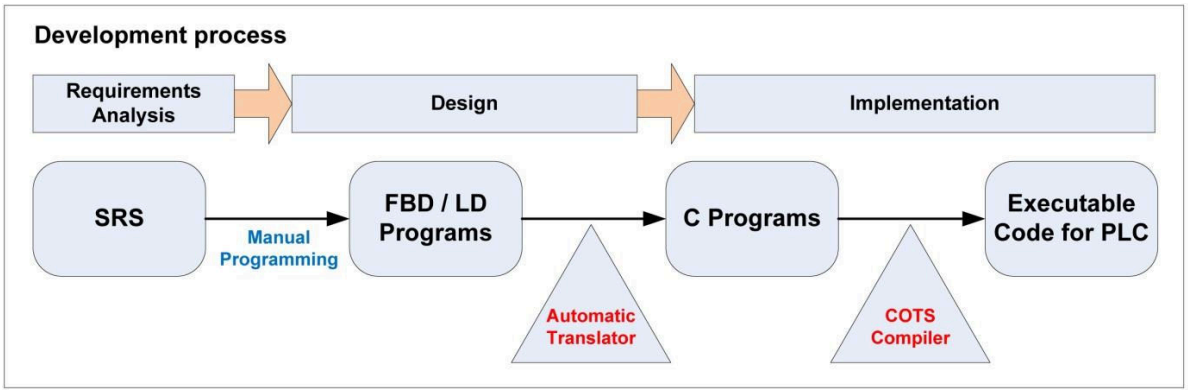

Fig. 1 A typical RPS software development process using PLCs

`*SPACE*' (SIEMENS 1996) is a software engineering tool-set for *AREVA*'s PLC `*TELEPERM XS*'. It stores FBD programs into a database `*INGRES*' and generates ANSI C programs to perform code-based testing and simulation (`*TXS SIVAT*' (RichterS, WittigJ. 2003)). *ISTec GmbH* (http://www.istec.de) also had developed a reverse engineering tool `*RETRANS*' (ISTec 1997) for checking consistency between FBD programs and generated C programs. The mechanical translator in `*SPACE*' has been validated in such ways, and the software engineering tool-sets have been used successfully for more than a decade.

PLCs of *invensys* also have been widely used. `*TriStation 1131*' (invensys n.d.) is its software engineering tool-set. It provides enhanced emulation-based testing and real-time simulation of FBDs, but does not include a translator into C programs yet. *KNICS* and *POSCO ICT* in Korea have recently developed a safety-level PLC `*POSAFE-Q*' and its software engineering tool-set `*pSET*' (Cho, et al. 2007). The tool-set provides a graphical editor for FBD and LD programming languages and generates ANSI-C programs mechanically. However, sufficient demonstration of correctness and functional safety of the so-called `*FBD-to-C*' translator is still in progress. It must be one of the most critical obstacles that should pass through to get permissions for the export of the new Korean nuclear power plant (Nuclear Power in South Korea n.d.) as a whole, *i.e.,* including control software - I&C.

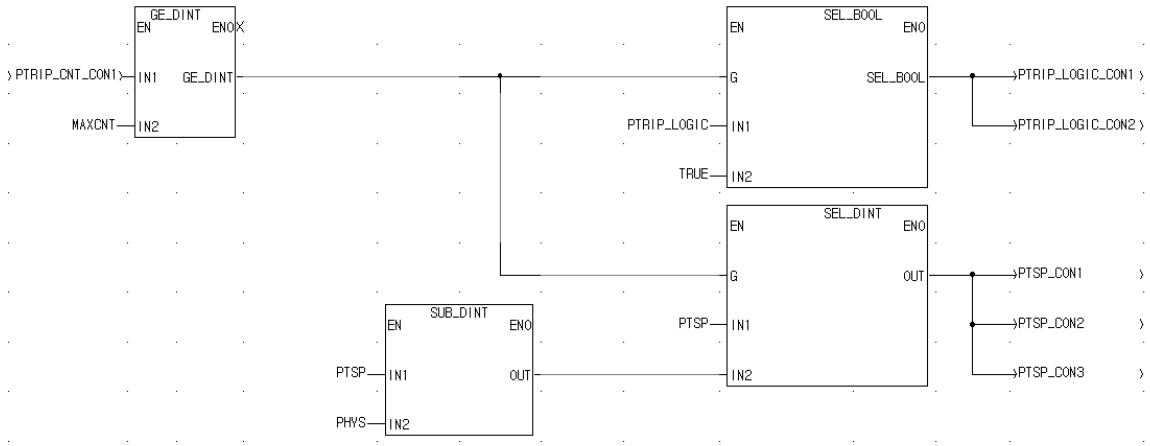

Fig. 2 A part of FBD for fixed set-point rising trip logic

4. STATIC CODE ANALYSIS

The objective of this case study is to check whether the mechanically translated C code is well-programmed from the viewpoint of security as well as functional safety and correctness. Since the C code is mechanically translated from FBD programs by PLC software engineering tools, static code analysis on the C code has been often skipped, and it is worth analyzing security and safety together.

We used a part of preliminary version of FBD programs '*fixed set-point rising trip logic*' for KNCIS APR-1400 RPS in Korea, as shown in

Fig. . It is a basic shutdown logic of RPSs. It consists of 23 function blocks and uses 4 inputs and 6 outputs. From the FBD programs, we produced C code mechanically using a translator provided by a PLC vendor. It is worth to note that the translator is also a preliminary version and we cannot access to the up-to-date version, unfortunately.

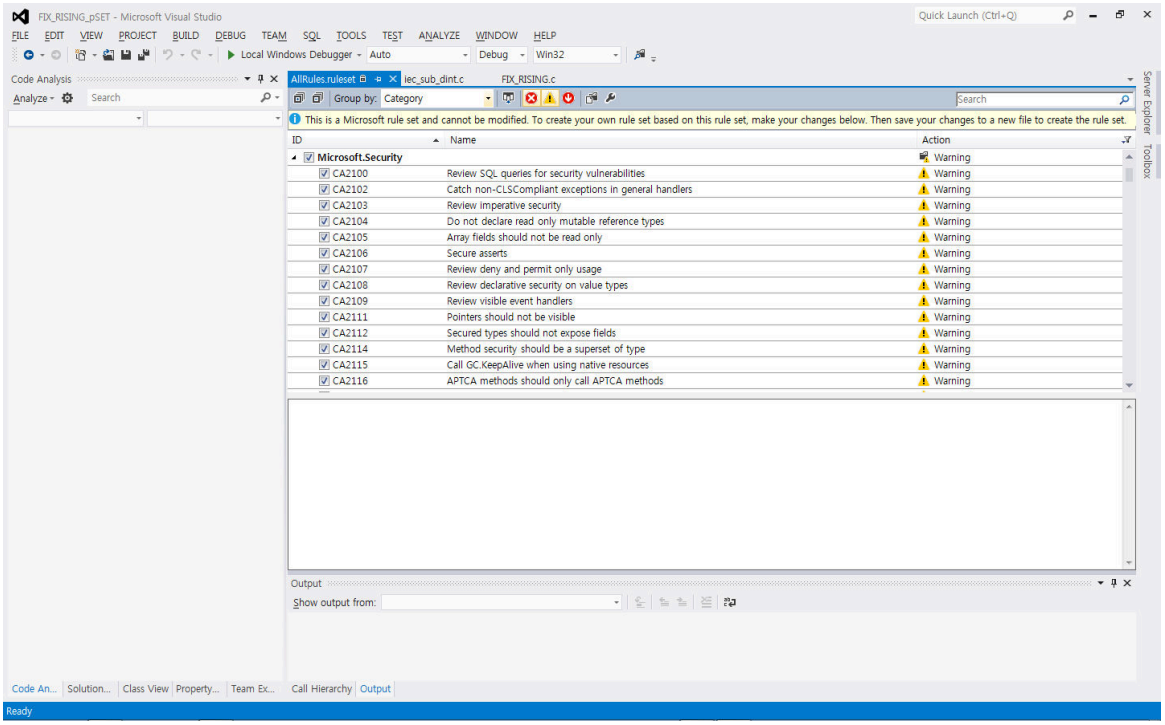

Fig. 3 A static analysis result for the security category

We used the static C code analysis tool included in *Microsoft Visual Studio 2012*. It includes 596 rules of 12 categories such as *Design*, *Globalization*, *Security* and *Reliability*. MSDL also provides detailed explanations on each rule. We can select some or all of them. Fig. shows the result of static code analysis for the *security* category. It detected no error but warning only. We, therefore, can conclude that the part of FBD has no security-related defect yet.

We also performed a full-scale static analysis on the C code and found 5 critical errors such as **C6001 Using uninitialized memory** and **C6281 Bitwise relation precedence**. C6001 reports that the code uses the variable *_tmp* without initialization as highlighted in Fig.4 below.

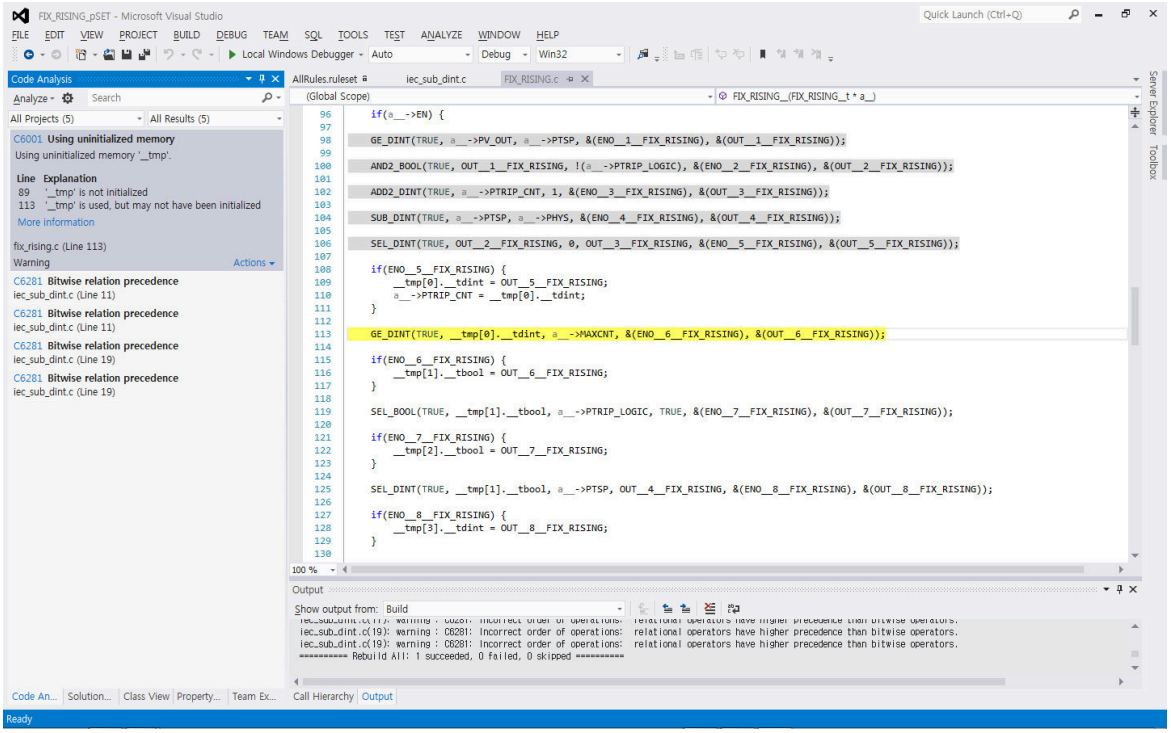

Fig. 4 The C6001 error

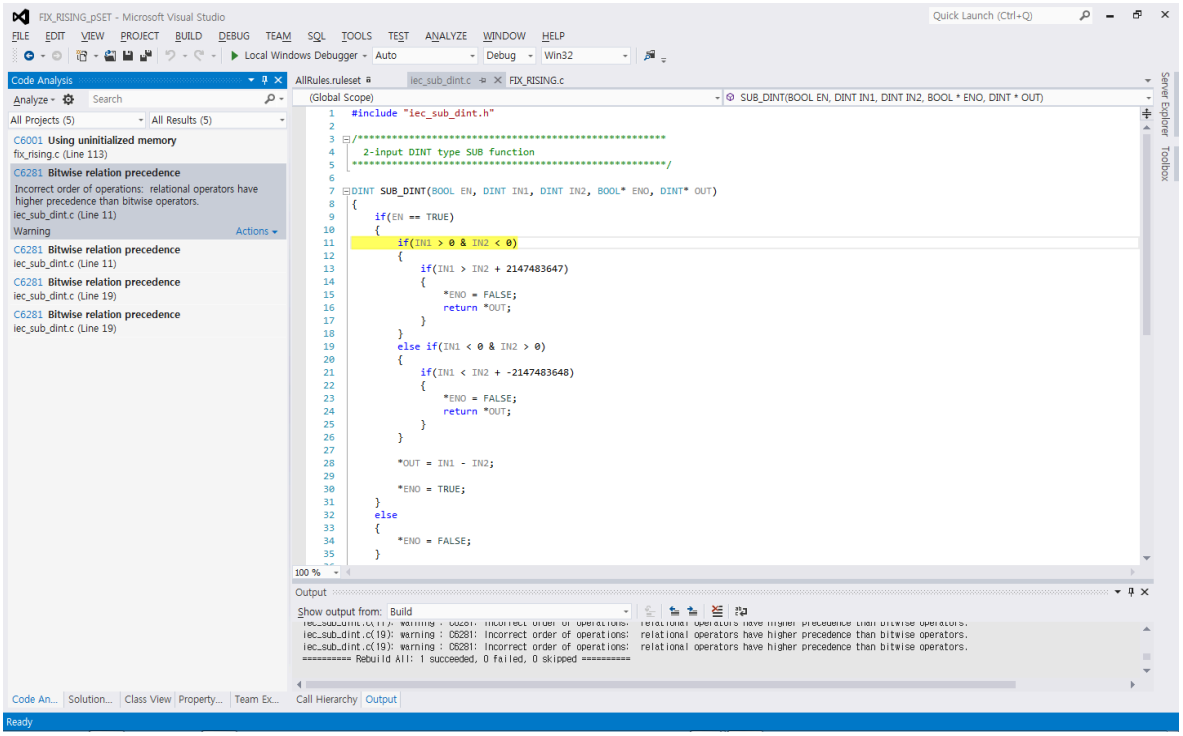

Fig. 5 The C6281 error

Fig. is a screen-dump of the C6281 error. It found an incorrect order of operation such that relational operators have higher precedence than bitwise operators used in *lec_sub_dint.c* file, which corresponds to SUB_DINT function block (subtraction with two decimal integer variables) used in Fig.2.

In the code above, the condition statement if (IN1 > 0 & IN2 < 0) may result in an error, since it uses not the logical operator && but the bitwise operator &. Since this C code corresponds to SUB_DINT function block, which used by all other FBD programs, the C code translated form the whole FBD program, which calls the SUB_DINT function block a number of times, have the same defect and a potential risk too. This defect should be fixed and translation rules from FBD to C also be modified, too.

5. CONCLUSIONS AND FUTURE WORK

This paper uses the static code analysis tool of Microsoft Visual Studio 2012 to check the C code, which had been translated mechanically from FBD programs by a PLC software engineering tool, against security and safety properties. Static code analysis on C code in the PLC software development has been looked down, since PLC vendors has the responsibility for demonstrating functional safety and correctness of the translator. While cybersecurity issues are highly recommended to be checked by automatic static analysis tool, this paper tries to check the translated C code against security as well as functional safety.

```
$ splint FIX_RISING.c
Splint 3.1.2 --- 28 Mar 2013

psettype.h:4:22: Type BOOL is probably meant as a boolean
type, but the boolean
                    type name is not set. Use -booltype BO
OL to set it.
  Use the -booltype, -boolfalse and -booltrue flags to cha
nge the name of the
  default boolean type. (Use -likelybool to inhibit warnin
g)
FIX_RISING.c: (in function FIX_RISING__)
FIX_RISING.c:96:5: Test expression for if not boolean, typ
e BOOL: a__->EN
  Test expression type is not boolean or int. (Use -predbo
olint to inhibit
  warning)
FIX_RISING.c:98:2: Return value (type BOOL) ignored: GE_DI
NT(TRUE, a_...
  Result returned by function call is not used. If this is
 intended, can cast
  result to (void) to eliminate message. (Use -retvalother
 to inhibit warning)
                          ...

FIX_RISING.c:61:6: Variable exported but not used outside
FIX_RISING:
                     OUT__24__FIX_RISING
FIX_RISING.c:62:6: Variable exported but not used outside
FIX_RISING:
                     ENO__25__FIX_RISING
FIX_RISING.c:63:6: Variable exported but not used outside
FIX_RISING:
                     OUT__25__FIX_RISING
FIX_RISING.c:64:6: Variable exported but not used outside
FIX_RISING:
                     ENO__26__FIX_RISING
FIX_RISING.c:65:6: Variable exported but not used outside
FIX_RISING:
                     OUT__26__FIX_RISING

Finished checking --- 103 code warnings
```

Fig. 6 A static analysis result from Splint

We used only a small part of FBD and C programs, but found some critical errors concerning functional correctness of the so called FBD-to-C translator. Even though we could not find security-related defects, the static analysis showed the potential of security defects possibly residing in the mechanically translated C code. Additionally, we also used other tool such as *Splint* and found more than 100 errors not categorized as depicted in Fig. 6. However, these static analysis tools and its rule sets are generally not focused on compiler for industrial computers and operating systems such as '*TMS320C55x*' C compiler for the PLC. Therefore we have to consider dissimilarity of systems, and need to select an appropriate static analysis tool and rule sets concerning security as well as safety in the process of secure software development methodology.

ACKNOWLEDGEMENT

This research was supported by the MKE (The Ministry of Knowledge Economy), Korea, under the Development of Performance Improvement Technology for Engineering Tool of Safety PLC (Programmable Logic Controller) program supervised by the KETEP (Korea Institute of Energy Technology Evaluation And Planning)" (KETEP-2010-T1001-01038). It was also supported, in part, by a grant from the Korea Ministry of Strategy, under the development of the integrated framework of I&C conformity assessment, sustainable monitoring, and emergency response for nuclear facilities.

REFERENCES

Andrewa, Mark Holt and Anthony. *Nuclear Power Plant Security and Vulnerabilities.* Congressional Research Service, 2012.

BBC. "Stuxnet worm hits Iran nuclear plant staff computers." September 26, 2010.

Cho, S., Koo K., You B., T.-W Kim, T. Shim, and J. Lee. "Development of the loader software for PLC programming." *Conference of the Institute of Electronics Engineers of Korea* 30 (2007): 959–960.

Commission, International Electrotechnical. *Functional safety of electrical, electronic and programmable electronic (E/E/PE) safety-related systems.* IEC, 2000.

International Electrotechnical Commission. *CD2 IEC 62645 Ed. 1.0.* IEC, 2011.

International Electrotechnical Commission. "Instrumentation and control systems important to safety - Software aspects for computer-based systems perfoming category A functions." IEC, 2006.

International Electrotechnical Commission. *International standard for programmable controllers: Programming languages, Part 3.* IEC, 2013.

invensys. *Safety software suite, TriStation 1131 (TS1131).* n.d. http://iom.invensys.com/.

ISO/IEC. "Information Technology -- Programming languages -- C." 2011.

ISTec. *RETRANS, reverse engineering tool for FBD programming of Teleperm XS PLC.* Technical Report, ISTec, 1997.

Korea Atomic Energy Research Institute. "Software Design Specification for Reactor Protection System." 2006.

Labaw, C.L. Heitmeyer and R.D. Jeffords and B.G. "Automated consistency checking of requirements specifications." (IEEE Transactions on Software Engineering) 5, no. 3 (1996): 231-261.

Microsoft. *Detecting and Correcting C/C++ Code Defects.* n.d. http://msdn.microsoft.com/en-us/library/ms182025.aspx.

"Nuclear Power in South Korea." *WIKIPEDIA.* n.d. http://en.wikipedia.org/wiki/Nuclear power in South Korea.

Ramsbrock, Daniel. *Secure Development Methodologies Overview.* 2010. http://blogs.captechconsulting.com/blog/daniel-ramsbrock/secure-development-methodologies-overview.

RichterS, WittigJ. "Verification and validation process for safety I&C systems." "Nuclear Plant Journal", 2003: 36-40.

SAFECode. *Fundamental ractices for Secure Software Development.* SAFECode, 2008.

SIEMENS. *Space, engineering system of Teleperm XS PLC.* SIEMENS (Germany), 1996.

TEXAS INSTRUMENTS. "TMS320C55x optimizing c/c++ compiler users guide." TEXAS INSTRUMENTS, 2003.

U.S. Nuclear Regulatory Commission. "Criteria for Use of Computers in Safety Systems of Nuclear Power Plants." US NRC, 2011.

U.S. Nuclear Regulatory Commission. *Cyber Security Porgrams for Nuclear Facilities.* US NRC, 2010.

U.S. Nuclear Regulatory Commission. "Protection of Digital Computer and Communication Systems and Networks." 2009.

Yoo, Junbeom, Taihyo Kim, Sungdeok Cha, and Han Sung Son. "A Formal Software Requirements Specification Method for Digital Nuclear Plants Protection Systems." *Journal of Systems and Software* 74, no. 1 (2005): 73-83.