Original Article

ISSN: 2349 - 4891

International
Journal of Recent Research and Applied Studies
(Multidisciplinary Open Access Refereed e-Journal)

A Research on Efficient Processor Design Structure with Reduced Memory Gap

Dr.S.R.Boselin Prabhu[1], Dr.E.Gajendran[2] & Balakumar, N[3]

[1]*Associate Professor, Department of Electronics and Communication Engineering, V.S.B College of Engineering–Technical Campus, Coimbatore, India.*
[2]*Professor, Department of Information Technology, St.Martin's Engineering College, Hyderabad, India.*
[3]*Assistant Professor, Department of Electrical and Electronics Engineering, Tamilnadu College of Engineering, Coimbatore, India.*

Received 29th October 2016, Accepted 1st December 2016

Abstract

Vigorous approaches for security are dominant in wireless embedded systems due to the characteristic vulnerability of the fundamental shared medium and the absence of any monitoring structure. To promise security and reliability, most of the current schemes execute periodic reestablishment of authentication authorizations and share secrets among numerous participating nodes. However, the present approaches fail to offer appropriate protection from packet sniffing or eavesdropping attacks. In addition, these methods are energy intensive and fail to scale well in energy-constrained situations. Instead of placing the entire matching pattern on the chip, our solution is the parallel intrusion detection system that works by combining extracting as much of the important filtering information as possible onto a chip and infrequently accessing off chip data to make the matching mechanism suitable for large pattern set. Testing is also introduced to clear the bug which is presented in the network application software. It will improve the coverage of the test, clearly saving in cost and development time. The virus detection and testing processor also protect the multi core systems from Real Time attacks and will provide the formal model–based test for multi core system. With the model a test suite can be extracted from the test case generator and a test program generator will generate test programs automatically. Both generators are assisted with model checking on the formal model. In this paper a detailed research on efficient processor design structure with reduced memory gap has been elaborated.

Keywords: Covert Channel, Embedded Systems, Media Access Control, Physical Layer, Soft Security, Embedded Processor, Memory Gap, Model-based test.

Introduction

As embedded systems transfuse numerous aspects of our lives, they are often obligatory to deal with delicate information or to achieve critical functions, building security an important worry in embedded system design. Security has been the topic of widespread research in the setting of general-purpose computing and communications systems, leading to countless advances such as cryptographic algorithms, security protocols, etc. While such functional security measures deliver a strong basis for safeguarding embedded systems, recent trends insecurity attacks have made it plentifully clear that most attacks target faintness in a system's implementation. It is now well acknowledged that a secure system implementation is as serious to a system's complete security as the strength of the theoretical security measures engaged. Accordingly, recent years have seen cumulative awareness that security needs to be deliberated at numerous stages of the embedded system design process, including system architecture and hardware/software application. A distinguished trend in embedded systems has been an extreme increase in embedded software content in order to provision increasing end-user functionality and performance. It is therefore not astonishing that the most common security attacks on embedded systems are software-based, or deed weaknesses in embedded software.

A multi-core processor [1] combines two or more independent cores into a single package composed of a single integrated circuit. It represents a major evolution in hardware technology. The majority of microprocessors contain more number of cores on die. It will provide more processing power from the hardware perspective for the network security application RTOS [2] (Real Time operating System).So testing in multi-core RTOS and the embedded network is very important for the system to work without any interruption. Currently, conventional test [3] is the main method for detecting bugs and these tests have several problems; like ensuring the perfect coverage of the test is impossible, particularly if the core has severe timing and it should be carried out with different configuration. Time consumption is high. We proposes the new type of testing which will improve the quality, cut cost and

Correspondence
Dr.S.R.Boselin Prabhu.
E-mail: eben4uever@gmail.com

reduce the time of test. The model-based test aims at a more complete test suite as well as the automation of the test. This paper gives the review on the new virus detection and correction processor which will combine the shift signature table and the trie skip mechanism to improve the performance and the blow of impact on memory gap for the two-phase architecture. The new table not only maintains the shift value of properties but also avoids reducing the filter rate for the large scale of pattern. The testing processor will increase the coverage of the test by eliminating the bug in the embedded network and multi-core RTOS. The rest of this paper is structured is as follows: Section II reviews several related work. Section III describes the types of attack text available. Section IV briefly describes blocks in virus detection processor for the embedded network security. Section V describes testing processor for the embedded network and multi-core RTOS. Section VI describes the complete model for virus detection and testing processor for secure embedded network. Section VII concludes with the brief discussion about conclusion and the performance comparison of the processors.

Related Works

Olatunji Ruwase, Phillip B. Gibbonssuggests “Parallelizing Dynamic Information Flow Tracking “, assesses a novel form of parallel dynamic information flow tracking (DIFT) that customs inheritance tables to track emblematically the net effects of segments that are treated in parallel by worker threads, and relaxes DIFT such that data flow needs to be tracked only through unary processes. The results validate speedups of multiple worker threads for numerous bench-marks. It also recommends a relaxed DIFT which bids substantial performance compensations over the original DIFT in numerous cases by reducing the work related with tracking information flow through binary operations [4-6].

Haibo Chen, Xi Wu, Liwei Yuan Presents “From Speculation to Security: Practical and Efficient Information Flow Tracking Using Speculative Hardware”, a system SHIFT, a low-overhead dynamic information flow tracking system for software security. SHIFT influences present hardware support for delayed exception tracking to lower the runtime overhead. It also outfits a prototype to defend applications running on Itanium processors. The assessment shows that SHIFT can overthrow a set of real world attacks with no known false positives. Performance measurements also specify that the performance strike in SHIFT is modest. Quantitative measurements display that minor architectural enhancements cannot significantly reduce the performance slowdown.

Lap Chung Lam, Tzi-cker Chiueh Presents “A General Dynamic Information Flow Tracking Framework for Security Applications”, the strategy and employment of a universal dynamic information-flow tracking structure for C programs which delivers an interface for designers to require their own tag initialization, propagation, and processing purposes. The proposed system also mechanically propagates these tags through an application program beside data dependencies and control dependencies that essentially take place at run time. The elasticity of the structure permits it to provision a wide range of applications. It also provisions flexible sandboxing strategies that the fundamental sandboxing system can alter by providing callback functions. The results also displays that the CPU overhead is less than 170% for computation-intensive solicitations, and is less than 20% for non-computation demanding applications [7-9].

G. Edward Suh, Charles W. O’Donnell Proposes “AEGIS: A Single-chip Secure Processor”, architecture which can be used to shape computing schemes that are secure against both software and physical outbreaks. It clarifies the reputation of physical random functions role and the arrangement delivers a way to generate, protect, and share secrets without the use of on-chip nonvolatile Memory. The four modes of secure execution, which AEGIS delivers enable new ways of creating applications, particularly with the use of a postponed secure mode to reduce the trust base without foregoing physical and software security. An embedded AEGIS architecture implementation also displays that performance overheads are negligible given typical applications.

Michael Dalton, Hari Kannan Presents “Raksha: A Flexible Information Flow Architecture for Software Security Novel Information Flow Architecture for Software Security”, delivers a framework that syndicates the best of both hardware and software DIFT. Hardware support offers transparent, fine-grain management of security tags at little performance overhead for user code, OS code, and data that marks multiple processes. Software provides the litheness and robustness necessary to contract with a wide range of attacks. It also provisions flexible and programmable security strategies. Software can set the security strategy to offer protection against a new type of attack or identify a corner case in the communication of existing security policies and deployed software. It provisions multiple active policies that permit concurrent protection against both high-level and low-level vulnerabilities and supports a user-level exception handling that permits for fast security handlers that execute in the same address space as the possibly malicious application [10-12].

G. Edward Suh, Jaewook Lee Presents "Secure Program Execution via Dynamic Information Flow Tracking", a hardware mechanism to track dynamic information flow which averts malicious software attacks. This structure enables the operating system identifies false input channels, and a processor tracks the false information flow from those channels. It labels a security policy concerning the use of the false data is enforced by the processor. The experimental outcomes prove that this method is effective in involuntary detection and defense of security attacks, and very effectual in terms of space and performance overheads.

Meltem Ozsoy, Dmitry Ponomarev Presented "SIFT: A Low-Overhead Dynamic Information Flow Tracking Architecture for SMT Processors", architecture delivers a secure computing model which shoulders that only the on-chip environment is secure from physical attacks, which can be realized on single-chip secure processor architecture. The new security modules added to the Open SPARC system experience little extra hardware costs and performance overhead. By prototyping the secure Open SPARCT1 processor, the arrangement also proves that the Open SPARC FPGA platform has many compensations such as ability to modify real hardware, ease of modification due to the emulated cache, capability to run commodity OS and benchmarks, the availability of an open source hypervisor.

Rajiv Gupta, Neelam Gupta Presents "Scalable Dynamic Information Flow Tracking and its Applications", SIFT- a novel architectural application of Dynamic Information Flow Tracking which uses an isolated thread to perform taint propagation and policy enforcement where thread is implemented in a spare context of a processor. Successfully, SIFT provides run-time taint instruction generation and hardware acceleration for a software-based DIFT structure but necessitates no compiler support or support for instrumentation of the source code or the program binary. SIFT experiences lower performance loss because of hardware-accelerated generation of checking instructions and also because of the execution of checking instructions in a distinct SMT context. In distinction to architectural DIFT solutions, SIFT conserves the design of all major data path components [13-15].

Crispin Cowan Presents "Buffer Overflows: Attacks and Defenses for the Vulnerability of the Decade", a combined approach for detecting and ending a wide range of usually reported attacks that abuse software implementation errors. The planned structure is based on a fully involuntary and efficient taint analysis procedure that can track the flow of distrusted data through a program at the granularity of bytes. Obtainability of fine-grained taint information allows simple and accurate strategies to be developed that can dependably stop many classes of attacks. The results also displays that this system can be applied to dissimilar types of applications written in multiple programming languages, and displays efficiency in detecting attacks without creating false positives.

Tadeusz Pietraszek, Chris Vanden Berghe "Defending against Injection Attacks through Context-Sensitive String Evaluation", designates a sole and specific application for web application security, where information flow is straight and immediately applicable. It also displays that the system can automatically find a large number of security susceptibilities in real-life web applications through the synergism of anew language for unfolding information flow, context-sensitive pointer alias analysis, dynamic monitoring and model examination.

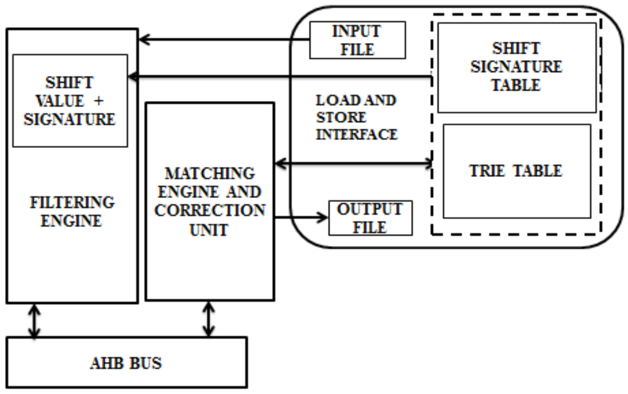

Figure I. Virus Detection/Correction Processor Architecture

Types of Attack Texts

Deep-Search Attack is the most common and the easiest to build. This type of attack texts regularly causes the filtering engine to launch alarms and require verification of the exact-matching engine. Two candidate positions cause exact-matching to take a long time to traverse its data structure. The Sub-Pattern attack provides a higher density text. If the given pattern contains the prefix of the other pattern, the attach text provide more candidate positions in the same length. The patterns "beach" and "each" are an example. Sub-Pattern attack is constituted by a series of character "d" and this can be considered to be a special case that combines by the first and second types and causes exact-matching for every position of itself. This extreme case dramatically lowers the performance. Although it can be avoided by choosing the pattern set well.

Virus Detection and Correction Processor

This architecture comprising the filtering engine and the exact matching engine. Filtering engine is responsible for filtering out the secure data efficiently and indicating to candidate positions that patterns possibly exist at the first stage. The filtering engine is otherwise called front end module. Exact matching engine is responsible for verifying the alarms caused by the filtering engine. It is otherwise called back end module. Figure 1 shows the virus detection and correction processor which has fast front end and slower back end. In this case we use 8MB memory to store the shift signature table and trie table. For storing shift signature value we use 4KB BiTCAM in the filtering engine. The exact-matching engine also supports data prefetching and caching techniques to hide the access latency of the memory by allocating its data structure well.

Model Based Testing Processor

The testing processor will include the testing, verification and validation for the embedded network and the multi-core RTOS. The model based testing processor is used to analyse the bug in the system. LTL is used to describe the model checking process. Figure 2 shows the model based testing processor. With a configuration [16], a concrete model for a system is created. Using the test case generator, an exhaustive test suite is extracted from the model.

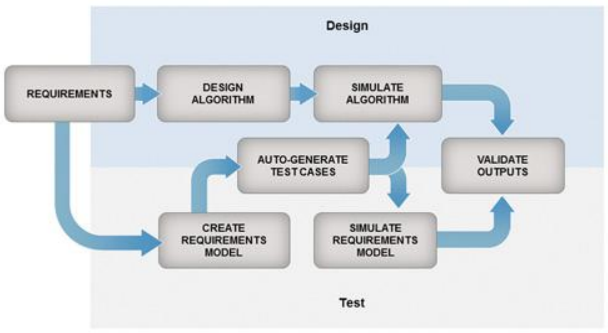

Figure II. Model-Based Testing Processor

Consequently, an entire and feasible test suite can be obtained after filtering with the coverage criteria. Finally, with the test program generator, the optimal test sequence for every test case is calculated, and then translated to execution program. If a bug exists, analysis is easy because it can be carried out step-by-step along the execution path provided from the model checker.

Review of Proposed Detection and Testing Processor

This virus detection and testing processor for secure embedded network and multi-core RTOS will provide complete security against attack such as virus and spam. The testing processor will use the same memory which is resided in the virus detection processor. Fig. 4 shows the block diagram of the complete processor which will increase the throughput compare to the original processor. This structure is mainly used to detect and correct a virus which is present in the embedded network and also it is used to test the software which is in the same network. The AHB Bus is used to transmit the date quickly between the filtering engine and the matching engine.

Bi-TCAM [17] unit is act as a filtering engine in this process and it is used to storethe shift signature value. When dealing with a large number of virus patterns, these design need a low area and power. Bi-TCAM is implemented in the filtering engine to detect the unsafe data efficiently.

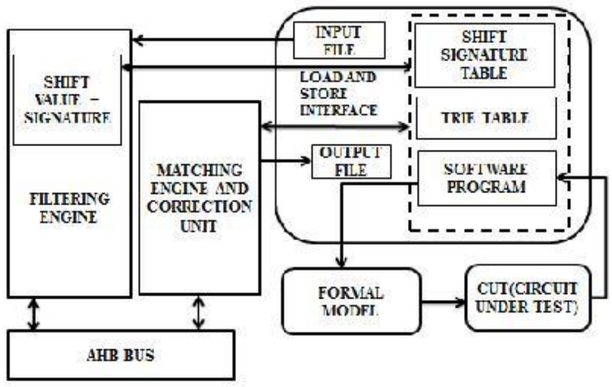

Figure III. Virus detection and Testing Processor architecture

The algorithm that works with the filtering engine is the modified shift signature algorithm which will store the shift signature table in the external memory to reduce the memory usage. The table that is used for this algorithm is included in the load/store interface to provide the excellent performance. The modified algorithm proposes the new shift signature table which will have only the shift value in the filtering engine. The modified shift-signature table has the same size as the original shift table, as its width and length are the same as the original shift table [18]. The carry field has two types of data: a shift value and a signature. These two data types are used by two different algorithms. The filtering engine can then filter the text using a different algorithm while providing a higher filter rate. The method used to merge these two tables is described as follows. First, the algorithm generates two tables, a shift table and signature table, at the pre-processing stage. The generation of the shift table is the same as in the Wu-Manber algorithm [8]. The shift table is used as the primary filter and its re-encodes the value. The signature table could be considered a set of the bit vector of the Bloom filter, and it is used for the second-level filtering. Generated signatures are mapped onto the signature table and indexed by bad-characters, which have shift values of zero in the shift table. After the shift table and signature table are generated, the algorithm re-encodes the shift value into two fields: an S-Flag and a carry in the shift-signature table. To merge these two tables, the algorithm maps each entry in the shift table and signature table onto the shift-signature table.

The proposed architecture storing the shift signature table in the external memory of 8MB. Once the shift value and the signature value are generated it will give those values to the filtering engine Bi-TCAM unit and it will give out the values to the matching engine for further process. This architecture also allows the parallel NIDS(Network Intrusion Detection System)[19, 20] to inspect high speed links with no packet loss and no negative impact on the accuracy of the traffic analysis that remain reliable and stateful. This parallel NIDS will reduce the computation time.

The matching engine verifies an alarm triggered by the filtering engine. Here the exact matching engine uses the compact trie structure. Trie skip mechanism is used to remove the redundancies. The compact tries generating the pattern with two points, sibling and child. It will check only the sibling and child to reduce the memory requirement. The correction unit was been introduced to remove the viruses from the network. Once the filtering engine alerts, the exact matching engine get the piece of the input text and create the root address. Now the engine reads the root node from the memory, compares it with the input text. The exact matching engine compares the child node, but it mismatches the child node. It compares the sibling node which is corresponding to the child node. It continuously checks the sibling node and the child node. Now it returns to the filtering engine to find the next candidate position. Once all the candidate position the output was given to the correction unit. The correction unit will correct the text and given output file which is in the load/store interface [22-24].The AHB Bus act as a high performance system backbone bus. It will provide efficient connection of processors, filtering engine and the exact matching engine.

Load/Store interface is otherwise called as Input/output interface. An I/O interface is required whenever the I/O device is driven by the processor. The interface must have necessary logic to interpret the device address generated by the processor. If different data formats are being exchanged, the interface must be able to convert serial data to parallel form and vice-versa. There must be provision for generating interrupts and the corresponding type numbers for further processing by the processor if required. The memory is included in this interface. The same memory is used for storing the shift value, trie table and embedded program. This system proposes 8MB of external SDRAM memory to store the shift signature table. The program of the network security application is stored in the memory of the load/store interface. The formal model is created from the program. The model creation [19], however, is one of the most difficult parts of the whole development process. Model abstraction is essential, both the performance and for scalability reasons. Corresponding the component based designed software architecture, we construct an abstract model of the basic software layer, and then create a concrete model with the runtime environment configuration. Formal model consist of test case generator and test program generator.

Testing is a disciplined process that consists of evaluating the application behavior, performance, and robustness-usually against expected criteria. These V&V(Verification and Validation) activities can be sub-classified as preventative, detective, or corrective measures of quality. It focuses on both the quality of the software product and of the engineering process. While testing is most often regarded as a detective measure of quality, it is closely related to corrective measures such as debugging.

Conclusion

A wide range of methods has been projected to augment software security in the background of general-purpose computing systems. Most ofthem discourse problems such as verifying the individuality of the provider of a program, inspection of the integrity of program binaries, safeguarding isolation between different programs running on a system, etc. Many previous designs have provide high performance, but the memory gap created by using the external memory decreases performance because of the increase size of virus database. In this paper a new method is introduced to protect the embedded processor and the multi-core RTOS which will reduce the memory usage and provide the reasonable solution. The testing processor is also integrated into the same memory will reduce the memory cost. The analysis for the bug is easy within the trace from the model checker. The power consumption and computation time is very low compared to the previous virus detection processor. A detailed review of a review of efficient processor design structure with reduced memory gap has been elaborated in this paper.

References

1. P. Paulin, F. Karim, P. Bromley, "Network Processors: A perspective on Market Requirements, Processor Architectures and Embedded S/W Tools," In Proceedings on Design, automation and test in Europe, IEEE Press, 2001, pp 420-429.
2. Balakumar N., "Evaluation of Quality in Network and Interoperable Connectivity between IP Networks", International Journal of Current Engineering and Scientific Research, Volume 3, Issue 9, pp. 81-85.
3. Aho and M. Corasick, "Efficient string matching: An aid to bibliographic search," Communications of the ACM, vol. 18, no. 6, June 1975, pp. 333-343.
4. L. Tan and T. Sherwood, "A high throughput string matching architecture for intrusion detection and prevention," in Proc. 32nd Annu. Int. Symp. Comput. Arch., 2005, pp. 112-122.

5. R. S. Boyer and J. S. Moore, “A fast string searching algorithm,” Communications of the ACM, vol. 20, no. 10, Oct. 1977, pp. 762-772.
6. Boselin Prabhu S.R. and Sophia S., “Environmental monitoring and greenhouse control by distributed sensor Network”, International Journal of Advanced Networking and Applications, 5(5), 2014.
7. Boselin Prabhu S.R. and Sophia S., “Greenhouse control using wireless sensor network”, Scholars Journal of Engineering and Technology, 2(4), 2014.
8. Boselin Prabhu S.R. and Sophia S., ‘Modern cluster integration of advanced weapon system and wireless sensor based combat system’, Scholars Journal of Engineering and Technology, 2(6A), 2014.
9. Boselin Prabhu S.R. and Sophia S., ‘A review of efficient information delivery and clustering for drip irrigation management using WSN’, International Journal of Computer Science and Business Informatics, 14(3), 2014.
10. Boselin Prabhu S.R. and Sophia S., ‘Mobility assisted dynamic routing for mobile wireless sensor networks’, International Journal of Advanced Information Technology, 3(3), 2013.
11. Boselin Prabhu S.R. and Sophia S., ‘A review of energy efficient clustering algorithm for connecting wireless sensor network fields’, International Journal of Engineering Research and Technology, 2(4), 2013.
12. Boselin Prabhu S.R. and Sophia S., ‘Variable power energy efficient clustering for wireless sensor networks’, Australian Journal of Basic and Applied Sciences, 7(7), 2013.
13. Boselin Prabhu S.R. and Sophia S., ‘Capacity based clustering model for dense wireless sensor networks’, International Journal of Computer Science and Business Informatics, 5(1), 2013.
14. Boselin Prabhu S.R. and Sophia S., ‘An integrated distributed clustering algorithm for dense WSNs’, International Journal of Computer Science and Business Informatics, 8(1), 2013.
15. Boselin Prabhu S.R. and Sophia S., ‘A research on decentralized clustering algorithms for dense wireless sensor networks’, International Journal of Computer Applications, 57(20), 2012.
16. Boselin Prabhu S.R. and Sophia S., ‘Hierarchical distributed clustering algorithm for energy efficient wireless sensor networks’, International Journal of Research in Information Technology, 1(12), 2013.
17. Boselin Prabhu S.R. and Sophia S., ‘Real-world applications of distributed clustering mechanism in dense wireless sensor networks’, International Journal of Computing Communications and Networking, 2(4), 2013.
18. S. Wu and U. Manber, “A fast algorithm for multi-pattern searching,” Univ. Arizona, Tuscon, Report TR-94-17, 1994.
19. H. Bloom, “Space/time trade-offs in hash coding with allowable errors,” Commun. ACM, vol. 13, pp. 762-772, 1977.
20. R.Gerth, D. Peled, M. Y. Vardi, and P. Wolper, “SimpleOn-The-Fly Automatic Verification of Linear Temporal Logic,” Proc. IFIP/WG6.1 Symp. Protocol Specification, Testing, and Verification (PSTV95), pp. 3-18, Warsaw, Poland, Chapman & Hall, June 1995.
21. P. E. Ammann, P. E. Black, and W. Majurski: Using Model Checking to Generate Tests from Specifications. Proceedings of 2nd IEEE International Conference on Formal Engineering Methods (ICFEM’98), IEEE Computer Society, pp. 46-54, 1998.
22. H. S. Hong, I. Lee, O. Sokolsky, and S. D. Cha: Automatic Test Generation from State Charts Using Model Checking. In Proceedings of FATES’01, Workshop on Formal Approaches to Testing of Software, vol. NS-01-4 of BRICS Notes Series, pp. 15-30, 2001.
23. Hamid Ali Abed Al-Asadi, “Temperature dependence of the lasing characteristics of vertical cavity surface emitting lasers," Engineering Journal of Technology University, Vol. 145, 1994.
24. Hamid Ali Abed Al-Asadi, “Theoretical investigation of spectral linewidth properties of double fused 1.3 um MQW-VCA in reflection and transition modes," Tikrit Journal for Pure Science, vol. 8, No. 2, 2002.