Volume No:2, Issue No:1 (June-2016) ISSN No : 2454-423X (Online)

International Journal of Research in Advanced Computer Science Engineering
A Peer Reviewed Open Access International Journal
www.ijracse.com

Network Intrusion Detection on Cyber Medical Systems

E.Rupa
M.Tech(CSE),
Jawaharlal Nehru Institute of Technology,
Hyderabad.

K.Shalini
Associate Professor,
Jawaharlal Nehru Institute of Technology,
Hyderabad.

Abstract:

We propose and analyze a behavior-rule specification-based technique for intrusion detection of medical devices embedded in a medical cyber physical system (MCPS) in which the patient's safety is of the utmost importance. We propose a methodology to transform behavior rules to a state machine, so that a device that is being monitored for its behavior can easily be checked against the transformed state machine for deviation from its behavior specification. Using vital sign monitor medical devices as an example; we demonstrate that our intrusion detection technique can effectively trade false positives off for a high detection probability to cope with more sophisticated and hidden attackers to support ultra safe and secure MCPS applications. Moreover, through a comparative analysis, we demonstrate that our behavior-rule specification-based IDS technique outperforms two existing anomaly-based techniques for detecting abnormal patient behaviors in pervasive healthcare applications.

Key Words:

Intrusion detection, sensor actuator networks, medical cyber physical systems, healthcare, security, safety.

Introduction:

Internet services and applications have become an inextricable part of daily life, enabling communication and the management of personal information from anywhere. To accommodate this increase in application and data complexity, web services have moved to a multi-tiered design wherein the webserver runs the application front-end logic and data are outsourced to a database or file server. DoubleGuard differs from this type of approach that correlates alerts from independent IDSs. Rather, DoubleGuard operates on multiple feeds of network traffic using single IDS that looks across sessions to produce an alert without correlating or summarizing the alerts produced by other independent IDSs.

This system used to detect attacks in multi-tiered web services. Our approach can create normality models of isolated user sessions that include both the web front-end (HTTP) and back-end (File or SQL) network transactions. For websites that do not permit content modification from users, there is a direct causal relationship between the requests received by the front-end webserver and those generated for the database back end. No prior knowledge of the source code or the application logic of web services deployed on the webserver. Virtualization is used to isolate objects and enhance security performance.

Architecture:

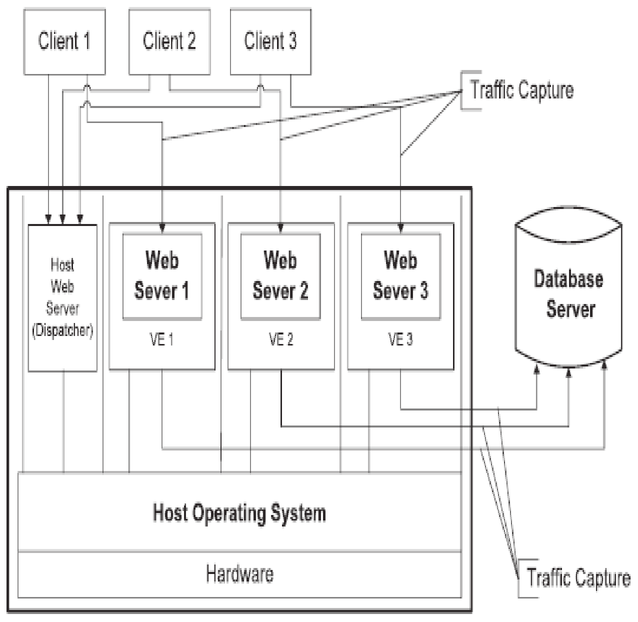

Fig: The Overall Architecture of our Prototype

Proposed System:

•It employs a lightweight virtualization technique to assign each user's web session to a dedicated container, an isolated virtual computing environment.

Volume No:2, Issue No:1 (June-2016) ISSN No : 2454-423X (Online)

International Journal of Research in Advanced Computer Science Engineering
A Peer Reviewed Open Access International Journal
www.ijracse.com

•It uses the container ID to accurately associate the web request with the subsequent DB queries. Thus, Double-Guard can build a causal mapping profile by taking both the webserver and DB traffic into account.
•In addition to this static website case, there are web services that permit persistent back-end data modifications. These services, which we call dynamic, allow HTTP requests to include parameters that are variable and depend on user input.

List of Modules:

•Web request collection
•Container creation
•Virtualization
•Detection engine

Web request collection:

websites that do not permit content modification from users, there is a direct causal relationship between the requests received by the front-end web server and those generated for the database back end. real-world network traffic obtained from the web and database requests. These services, which we call dynamic, allow HTTP requests to include parameters that are variable and depend on user input. Therefore, our ability to model the causal relationship between the front end and back end is not always deterministic and depends primarily upon the application logic.

Container creation:

When the request rate is moderate (e.g., under 110 requests per second), there is almost no overhead in comparison to an unprotected system. Even in a worst case scenario when the server was already overloaded, we observed only 26% performance overhead. The container-based web architecture not only fosters the profiling of causal mapping, but it also provides an isolation that prevents future session-hijacking attacks. Within a lightweight virtualization environment, we ran many copies of the web server instances in different containers so that each one was isolated from the rest. we assigned each client session a dedicated container so that, even when an attacker may be able to compromise a single session, the damage is confined to the compromised session; other user sessions remain unaffected by it.

Virtualization:

Virtualization is used to isolate objects and enhance security performance. Full virtualization and Para-virtualization are not the only approaches being taken. An alternative is a lightweight virtualization. virtualization techniques are commonly used for isolation and containment of attacks. However, in our Double Guard, we utilized the container ID to separate session traffic as a way of extracting and identifying causal relationships between web server requests and database query events. Double Guard focuses on modelling the mapping patterns between HTTP requests and DB queries to detect malicious user sessions.

Detection engine:

A single physical webserver runs many containers, each one an exact copy of the original webserver. Our approach dynamically generates new containers and recycles used ones. As a result, a single physical server can run continuously and serve all web requests. However, from a logical perspective, each session is assigned to a dedicated webserver and isolated from other sessions. Since we initialize each virtualized container using a read-only clean template, we can guarantee that each session will be served with a clean webserver instance at initialization. We choose to separate communications at the session level so that a single user always deals with the same webserver. Sessions can represent different users to some extent, and we expect the communication of a single user to go to the same dedicated webserver, thereby allowing us to identify suspect behaviour by both session and user. In our system, an attacker can only stay within the webserver containers that he/she is connected to, with no knowledge of the existence of other session communications.

We can thus ensure that legitimate sessions will not be compromised directly by an attacker. Both the web request and the database queries within each session should be in accordance with the model. If there exists any request or query that violates the normality model within a session, then the session will be treated as a possible attack. The attacker visits the website as a normal user aiming to compromise the webserver process or exploit vulnerabilities to bypass authentication. At that point, the attacker issues a set of privileged (e.g., admin-level) DB queries to retrieve sensitive information.

We log and process both legitimate web requests and database queries in the session traffic, but there are no mappings among them. DoubleGuard separates the traffic by sessions. If it is a user session, then the requests and queries should all belong to normal users and match structurally. Using the mapping model that we created during the training phase, DoubleGuard can capture the unmatched cases. We establish the mappings between HTTP requests and database queries, clearly defining which requests should trigger which queries. For an SQL injection attack to be successful, it must change the structure (or the semantics) of the query, which our approach can readily detect. First of all, according to our mapping model, DB queries will not have any matching web requests during this type of attack. On the other hand, as this traffic will not go through any containers, it will be captured as it appears to differ from the legitimate traffic that goes through the containers. DoubleGuard is designed to mitigate DDoS attacks. These attacks can occur in the server architecture without the back-end database.

Attack Scenarios:

Our system is effective at capturing the following types of attacks:

•Escalation Attack

•Hijack Future Session Attack

•Injection Attack

•Privilege Direct DB Attack

Privilege Escalation Attack:

Let's assume that the website serves both regular users and administrators. For a regular user, the web request ru will trigger the set of SQL queries Qu; for an administrator, the request ra will trigger the set of admin level queries Qa. Now suppose that an attacker logs into the web server as a normal user, upgrades his/her privileges, and triggers admin queries so as to obtain an administrator's data. This attack can never be detected by either the web server IDS or the database IDS since both ru and Qa are legitimate requests and queries. Our approach, however, can detect this type of attack since the DB query Qa does not match the request ru, according to our mapping model.

Hijack Future Session Attack:

This class of attacks is mainly aimed at the web server side. An attacker usually takes over the webserver and therefore hijacks all subsequent legitimate user sessions to launch attacks. For instance, by hijacking other user sessions, the attacker can eavesdrop, send spoofed replies, and/or drop user requests. A session-hijacking attack can be further categorized as a Spoofing/Man-in-the-Middle attack, an Exfiltration Attack, a Denial-of-Service/Packet Drop attack, or a Replay attack. According to the mapping model, the web request should invoke some database queries (e.g., a Deterministic Mapping then the abnormal situation can be detected. However, neither a conventional webserver IDS nor a database IDS can detect such an attack by itself. Fortunately, the isolation property of our container based webserver architecture can also prevent this type of attack. As each user's web requests are isolated into a separate container, an attacker can never break into other users' sessions.

Injection Attack:

Attacks such as SQL injection do not require compromising the webserver. Attackers can use existing vulnerabilities in the webserver logic to inject the data or string content that contains the exploits and then use the webserver to relay these exploits to attack the back-end database. Since our approach provides two-tier detection, even if the exploits are accepted by the webserver, the relayed contents to the DB server would not be able to take on the expected structure for the given webserver request. For instance, since the SQL injection attack changes the structure of the SQL queries, even if the injected data were to go through the webserver side, it would generate SQL queries in a different structure that could be detected as a deviation from the SQL query structure that would normally follow such a web request

Direct DB Attack:

It is possible for an attacker to bypass the webserver or firewalls and connect directly to the database. An attacker could also have already taken over the webserver and be submitting such queries from the webserver without sending web requests. Without matched web requests for such queries, a webserver IDS could detect neither. Furthermore, if these DB queries were within the set of allowed queries, then the database IDS it would not detect it either. However, this type of attack can be caught with our approach since we cannot match any web requests with these queries.

Anomaly detection:

Anomaly detection also referred to as outlier detection refers to detecting patterns in a given data set that do not conform to an established normal behaviour. The patterns thus detected are called anomalies and often translate to critical and actionable information in several application domains. Anomalies are also referred to as outliers, change, deviation, surprise, aberrant, peculiarity, intrusion, etc. In particular in the context of abuse and network intrusion detection, the interesting objects are often not rare objects, but unexpected bursts in activity. This pattern does not adhere to the common statistical definition of an outlier as a rare object, and many outlier detection methods (in particular unsupervised methods) will fail on such data, unless it has been aggregated appropriately. Instead, a cluster analysis algorithm may be able to detect the micro clusters formed by these patterns.Three broad categories of anomaly detection techniques exist.

Unsupervised anomaly detection:

Techniques detect anomalies in an unlabeled test data set under the assumption that the majority of the instances in the data set are normal by looking for instances that seem to fit least to the remainder of the data set.

Supervised anomaly detection:

Techniques require a data set that has been labelled as “normal” and “abnormal” and involves training a classifier (the key difference to many other statistical classification problems is the inherent unbalanced nature of outlier detection).

Semi-supervised anomaly detection:

Techniques construct a model representing normal behaviour from a given normal training data set, and then testing the likelihood of a test instance to be generated by the learnt model.

Conclusion

For safety-critical MCPSs, being able to detect attackers while limiting the false alarm probability to protect the welfare of patients is of utmost importance. In this paper we proposed a behavior-rule specification-based IDS technique for intrusion detection of medical devices embedded in a MCPS. We exemplified the utility with VSMs and demonstrated that the detection probability of the medical device approaches one (that is, we can always catch the attacker without false negatives) while bounding the false alarm probability to below 5 percent for reckless attackers and below 25 percent for random and opportunistic attackers over a wide range of environment noise levels. Through a comparative analysis, we demonstrated that our behavior rule specification-based IDS technique outperforms existing techniques based on anomaly intrusion detection.

References:

[1] H. Al-Hamadi and I. R. Chen, “Redundancy management of multipath routing for intrusion tolerance in heterogeneous wireless sensor networks,” IEEE Trans. Netw. Service Manage., vol. 10, no. 2, pp. 189–203, Jun. 2013.

[2] M. Anand, E. Cronin, M. Sherr, M. Blaze, Z. Ives, and I. Lee, “Security challenges in next generation cyber physical systems,” Beyond SCADA: Netw. Embedded Control for Cyber Phys. Syst., Pittsburgh, PA, USA, Nov. 2006.

[3] B. Asfaw, D. Bekele, B. Eshete, A. Villafiorita, and K. Weldemariam, “Host-based anomaly detection for pervasive medical systems,” in Proc. 5th Int. Conf. Risks Security Internet Syst., Oct. 2010, pp. 1–8.

[4] F. Bao, I. Chen, M. Chang, and J.H. Cho, “Trust-based intrusion detection in wireless sensor networks,” in Proc. IEEE Int. Conf. Commun., Jun. 2011, pp. 1–6.

[5] F. Bao, I. R. Chen, M. Chang, and J. H. Cho, “Hierarchical trust management for wireless sensor networks and its applications to trust-based routing and intrusion detection,” IEEE Trans. Netw. Service Manage., vol. 9, no. 2, pp. 169–183, Jun. 2012.

[6] F. B. Bastani, I. R. Chen, and T. W. Tsao, “Reliability of systems with fuzzy-failure criterion,” in Proc. Annu. Rel. Maintainability Symp., Anaheim, CA, USA, Jan. 1994, pp. 442–448.

[7] A. Carcano, A. Coletta, M. Guglielmi, M. Masera, I. Fovino, and A. Trombetta, “A multidimensional critical state analysis for detecting intrusions in SCADA systems,” IEEE Trans. Ind. Inf., vol. 7, no. 2, pp. 179–186, May 2011.

[8] A. C_ardenas, S. Amin, B. Sinopoli, A. Giani, A. Perrig, and S. Sastry, “Challenges for securing cyber physical systems,” in Proc. 1st Workshop Cyber-Phys. Syst. Security DHS, 2009, pp. 1–4.

[9] I. R.ChenandF. B. Bastani, “Effect of artificial-intelligence planning procedures on system reliability,” IEEE Trans. Rel., vol. 40, no. 3, pp.364–369,Aug.1991.

[10] I. R. Chen, F. B. Bastani, and T. W. Tsao, “On the reliability of AI planning software in real-time applications,” IEEE Trans. Knowl. Data Eng., vol. 7, no. 1, pp. 4–13, Feb. 1995.

[11] I. R. Chen and T. H. Hsi, “Performance analysis of admission control algorithms based on reward optimization for real-time multimedia servers,” Perform. Eval., vol. 33, no. 2, pp. 89–112, 1998.

[12] I. R. Chen, A. P. Speer, and M. Eltoweissy, “Adaptive fault tolerant QOS control algorithms for maximizing system lifetime of query-based wireless sensor networks,” IEEE Trans. Dependable Secure Comput., vol. 8, no. 2, pp. 161–176, Mar./Apr. 2011.