Barriers to the Use of Intrusion Detection Systems in Safety-Critical Applications

Chris W. Johnson,

School of Computing Science, University of Glasgow, Glasgow, G12 8RZ.

Johnson@dcs.gla.ac.uk,
http://www.dcs.gla.ac.uk/~johnson

Keywords: SCADA, Industrial Control Systems, Intrusion Detection, Safety, Cyber-Security.

Abstract. Intrusion detection systems (IDS) provide valuable tools to monitor for, and militate against, the impact of cyber-attacks. However, this paper identifies a range of theoretical and practical concerns when these systems are integrated into safety-critical systems. White-list approaches enumerate the processes that can legitimately exploit system resources and any other access requests are interpreted to indicate the presence of malware. They cannot easily be used in safety-related applications where the use of legacy applications and Intellectual Property (IP) barriers associated with the extensive use of sub-contracting can make it different to enumerate the resource requirements for all valid processes. In contrast, blacklist intrusion detection systems characterize the behavior of known malware. In order to be effective, blacklist IDS must be updated at regular intervals. This raises enormous concerns in safety-critical systems where extensive validation and verification requirements ensure that software updates must be rigorously tested. In other words, there is a concern that an IDS signature update might itself introduce bugs into a safety-related system. Isolation between an IDS and a safety related application can minimize this threat, for instance, using information diodes. However, further problems arise when IDS false positives compromise the reliability of safety-related applications.

1 Introduction

Intrusion detection systems (IDS) provide valuable tools for detecting potential cyber-attacks. Aldenstein [1] stresses the need for protective monitoring over Supervisory Control and Data Acquisition (SCADA) systems – based on the capability of operating systems to monitor running processes and to examine the raw memory of a machine. Sutherland et al [2], have also explored the use of protective monitoring tools to identi-

fy requirements for better access to information on memory, network and system activity to inform intrusion detection.

Most previous work in this area has focused on intrusion detection for UNIX and Windows platforms. However, problems arise when control systems are hosted as applications on top of these mass-market operating systems. Changes are often made by suppliers– for instance to customise file handling. This can frustrate attempts to use existing IDS when operating system modifications undermine existing authentication and security mechanisms. For instance, role-based file access mechanisms are typically used to implement security policies. Lockout mechanisms can then be used to ensure that unauthorized users cannot access a system while authorized users continue. However, many control systems use ad hoc or absolute permission techniques where system processes are not sufficiently distinguished to trace potential access violations – especially in legacy SCADA implementations.

This paper identifies a range of theoretical and practical concerns when intrusion detection systems are integrated into safety-critical applications. Whitelist approaches enumerate the processes that can legitimately exploit system resources. These cause problems in most safety-related applications because the costs of certification and development have led to a widespread reliance on sub-contracting and on legacy applications. It is, therefore, difficult for companies to obtain precise details of the systems they operate when contractual and IP barriers frustrate the disclosure of technical information that was not considered relevant during the acquisition process.

In contrast, blacklist intrusion detection systems rely on signatures that characterize the behaviour of known malware. In order to be effective, these signatures must be updated at regular intervals. Otherwise, the IDS will not identify the latest generation of threat. However, this raises enormous concerns in safety-critical systems where extensive validation and verification requirements ensure that software updates must be rigorously tested to ensure that they will not compromise application processes. In other words, there is a concern that an IDS signature update might introduce failure modes, including bugs, which would compromise

a safety-related system. This threat can be minimized by demonstrating the isolation between a blacklist IDS and safety related systems using information diodes.

Many regulators lack the technical insights necessary to approve such practices; in consequence very few IDS have been implemented within industrial control applications. Further problems stem from the manner in which conventional IDS only operate on conventional IP stacks rather than the lower level protocols used in many SCADA systems. More theoretical barriers include the sensitivity of IDS in safety-related applications. It is important not to miss positive instances of malware within a safety related application – however, if an IDS is too sensitive then it may trigger alarms even for nominal behaviour. In this sense, the IDS can itself trigger a denial of service. The closing sections focus on the issues that arise after an IDS has correctly identified the presence of malware in safety-critical systems. A host of concerns remain over our ability to reach a safe state following an attack, these are mirrored by uncertainty over appropriate forensic activities when – for example, preserving safety may overwrite the information needed to diagnose the causes of any incident.

2 Manual Approaches to Intrusion Detection

National and international organisations warn against an increasing threat from malware to safety-critical systems [3, 4]. However, it is hard to quantify the extent of this threat – given concerns over the disclosure of information about previous attacks. There are further technical and organisational barriers that frustrate attempts to detect potential intrusions. Malware is often identified as part of the normal fault detection processes that support the operation of complex infrastructures [5]. System logs and network-monitoring tools provide the evidence necessary to identify an intrusion [6]. However, a number of factors combine to undermine cyber-Situation Awareness in safety-critical systems:

- *Legacy systems.* Many safety-related applications combine layers of software that were gradually developed over many years. This complicates the manual detection of malware given that legacy systems

were seldom designed with cyber-security as a primary concern. Further problems arise because it can be difficult for engineers to distinguish normal behaviour from the symptoms of an attack when there is a limited understanding of the proprietary code that was written many decades before and often from companies that are no longer in business. There have also been cases where legacy systems contract legacy viruses – for instance through the use of obsolete but infected floppy drives, forcing engineers to reconstruct diagnosis techniques for legacy applications.

- *IPR concerns and out sourcing.* The manual detection of malware is complicated because many safety-critical industries now make extensive use of out-sourcing. This can include the provision of network services. Unfortunately, outsourcing a service does not outsource the risk to safety-related companies. In contrast, it can create vulnerabilities when malware propagates from sub-contractors that lack the security culture of the companies that employ them. Out-sourcing complicates the manual detection of malware using standard debugging techniques because the safety-critical organisation may identify the symptoms of an attack without the detailed understanding of the sub-contractors code that would be required to diagnose the cause or to unambiguously identify those symptoms as the result of an intrusion. In such circumstances, it is hard for end users who experience the safety-related consequences of a security breach to trace the technical causes of particular violations.

- *Failure of incident reporting.* There are many barriers to the reporting of security violations up the supply chain. Contracting companies have significant concerns about the legal and commercial implications of admitting cyber incidents on their future business. Other industries have responded by creating legal requirements to share information about cyber-attacks. Article 13a of the Telecoms Directive (2009/140/EC) requires network service providers to report significant security breaches and losses of integrity to competent national authorities. The United States Security and Exchange Commission also expects its members to file information about cyber incidents. It is for

this reason that the proposed EC Network and Information Security Directive (COM2013/48) extends the Article 13a reporting obligations across all European critical infrastructure providers. In the absence of such requirements, it can be difficult for safety-critical service providers to use information about previous incidents to guide the detection of future attacks.

- *Lack of competency.* Few safety-critical organisations have the technical capacity to diagnose and combat a broad range of attacks on their own infrastructures. It is for this reason that the US government has established their Industrial Control System Computer Emergency Response Team (ICS-CERT, https://ics-cert.us-cert.gov). They have the specialist expertise that is required to identify malware without undermining the safety of complex application processes. Many European industries lack this support and instead have to rely on external security service providers who have little understanding of the critical nature of the underlying software architectures.

- *Lack of appropriate guidance.* Both ENISA [3] and NIST [8] provide valuable guidance on how to improve the detection and reporting of potential incidents, neither considers the role of sub-contractors in gathering evidence about cyber attacks. Neither considers the complexities that can arise in safety-critical applications, for example when malware is potentially detected in software that has gone through a formal certification process. This is an important omission; lives may depend on the timely provision of information about the scope and extent of any violation.

- *Lack of forensic tools.* The manual detection of an intrusion is complicated in safety-related systems because many of these applications rely on devices that are very different from the office-based systems that are the focus of most forensic analysis. While there are significant monitoring and analysis tools available for conventional IP networks and devices, very few are available for Supervisory Control And Data Acquisition (SCADA) environments that employ protocols such as

HART for communication between field devices, including Programmable Logic Controllers (PLCs)

The factors that complicate the detection of malware using conventional network and systems analysis tools justify attempts to identify techniques that can be used to support automated intrusion detection in safety-critical systems.

3 Blacklist Approaches to Intrusion Detection

NIST [9] advocate the use of several different intrusion detection systems to automatically detect potential incidents. The intention is to increase cyber situation awareness. However, the use of diverse monitoring systems further complicates safety-critical software engineering. In order to obtain regulatory approval, companies must demonstrate the reliability of their code within its intended context of use. Companies must show that intrusion detection and prevention systems, antivirus software, and file integrity systems do not undermine the safety of application processes.

Blacklisting relies on detecting the characteristics of malware in contrast to whitelisting, discussed in subsequent sections, which compiles lists of approved code. All processes/requests are approved unless they are explicitly mentioned on the blacklist. In contrast, whitelist approaches to intrusion detection block everything by default unless they are explicitly approved. Greylist approaches enable the temporary suspension of access rights.

Blacklist techniques can be applied to a range of resources including email addresses – for example, to prevent phishing attacks. They can also be applied to IP addresses and DNS to throttle back denial of service attacks. Most typically, however, blacklists are used to record characteristics of malware including file names, types, sizes, content patterns etc. Control software is needed to implement the blacklist, blocking attempts to execute the files associated with known malware.

A number of further concerns limit the application of blacklist approaches in safety-related systems. Most IDS systems and their associated malware signatures focus on office based systems. This is justified because most attacks are focussed on these more general protocols. However, existing IDS cannot typically protect industrial automation protocols, such as HART. They will not detect attacks at this level. There have, however, been initial attempts to develop IDS for automation protocols, including Modbus, which have been embedded within commercial, open source tools [10]. Without further work to improve incident reporting and exchange, significant doubts remain about whether the signatures that are embedded within these tools can accurately characterise the range of threats being deployed against safety-critical applications [5].

Many control systems are distributed across a wide geographical area. Many components in process environments are not networked. This limits opportunities to use blacklist approaches. There is no easy way for system administrators to automatically update nodes with anti-virus definitions. This limitation can be addressed by manually uploading malware signatures – however, the more frequently these updates are applied then the greater the likelihood that the update process may itself lead to cross-contamination. There are also significant resource implications from implementing this policy across complex, distributed control systems.

In safety environments, there is a clear concern to ensure that a blacklist IDS does not result in critical software being erroneously blocked. Other concerns centre on the reliability of the IDS itself. Uploading a corrupted blacklist could cause the failure of a detection system with knock-on consequences for safety-related processes. In such circumstances, safety engineers would continually be engaged in a test and re-test cycle to ensure that new versions of protection and detection systems could safely be integrated into critical operating environments. There is a trade-off between the time required to verify that new malware signatures would not affect the reliability of an IDS and the imperative to quickly upload new definitions that might protect safety-related applications from a new potential threat.

4 Whitelist Approaches to Intrusion Detection

Whitelist approaches provide an alternative for intrusion detection in safety-critical applications. These profile 'normal behaviour' so that deviations can be reported. A deep knowledge of normal operation can be gained by reviewing logs and through the routine analysis of system behaviour. Whitelisting aims to ensure that only approved programs and software libraries can be executed. All others are denied system resources. In order to be successful, this approach relies on the following measures:

- identifying specific executables and software libraries which should be permitted to execute on a given system, his must not simply rely on file names or directory structures given that malware might then masquerade as a legitimate application;
- preventing any other executables and software libraries from functioning on that system. This can be implemented by creating a hash digest of all software applications. If the hash of an executable does not match what is contained in the list, it will run and trigger a security event.
- preventing unauthorised users from changing the lists indicating which files can be executed [11].

Application whitelisting depends on software that maintains the lists of approved executable and library files. It also depends on the maintenance of Access Control Lists that prevent unauthorised users from manipulating these lists. In a safety related environment, the software used to implement a whitelist approach must pass the relevant regulatory requirements. In addition, the approved lists may themselves be subject to safety assessments given the implications of denying resources to critical executable files. A host of commercial tools exist to support whitelisting – including Microsoft's AppLocker, the Bit9 Parity Suite, McAfee Application Control etc.

A small number of attempts have been made to extend this approach to safety-critical infrastructures [12]. The proponents of this technique argue that it has significant benefits over alternate approaches, including blacklisting. Whitelist approaches provide a degree of protection again zero-day exploits – even if the signature of an attack is unknown, the ma-

licious code will not be included on the approved hash list. Protection against zero-day exploits is extremely important for SCADA systems. The Human Machine Interfaces (HMIs) that control local processes are, typically, connected to geographically remote data historians and servers. The relative stability of the software running on these systems provides considerable opportunities for the use of whitelist techniques. Locking down the data historian and HMI can block zero-day exploits and notify the remote command and control center.

Whitelist IDS have significant benefits over blacklist approaches for safety-related systems when infrastructure components are isolated from standard network connections. Recall that there are significant resource implications when administrators have to manually update malware signatures across complex, distributed control systems. Whitelisting avoids many of these over-heads as the approved process list is likely to be more stable than the blacklist malware signatures for these 'air gapped' control systems.

There are some complications – for instance, if the same attack is launched across multiple instances of a control system then it may simultaneously lead to a large number of distributed security events. System administrators must periodically inspect system logs to identify patterns of attack across remote systems isolated by these air gaps. System administration is further complicated by the need to periodically update the hash tables that record the list of approved executable files, for example when an application needs to be patched. The whitelist software must be disabled without exposing the system to a synchronised attack. Recall that other application processes will still be running to support safety-related functions during the update process. Recalculating the hash tables for approved software can take several hours for even relatively simple control applications. Hybrid approaches use blacklist software to scan for malware both before and after the whitelist update process. This still creates vulnerabilities – for instance, from zero day exploits that would not be identified during the scan. Further concerns arise when an attacker obtains physical access to a control system – potentially enabling them to reboot the workstation without accessing the whitelist software.

There are further organisational concerns in implementing whitelist approaches to intrusion detection across complex safety-critical systems. It can be hard to coordinate the activities of many different sub-contractors to ensure that they do not trigger security events by installing unrecognised executable files. This creates particular concerns when the time required to re-computed the hash values might delay necessary safety updates. Further problems can arise in ensuring that engineering teams support the policy when they may have grown accustomed to making ad hoc updates to the systems they support. In consequence, whitelisting techniques may be restricted to a safety kernel within more complex applications.

Here we have focussed on what is termed 'application white listing'. However, there are alternative approaches that focus on the resource usage of application processes rather than executable file structures. In this approach, each recognised process on a whitelist is granted finite access to network, memory and processing resources. If the approved process exceeds these limits then a security event will be generated. Further concerns restrict the application of this resource-based whitelisting in safety-critical systems. Many safety-critical systems do not routinely have the level of monitoring implemented, for example by financial institutions. Networks that have experienced few operational problems will often not be analysed to any significant extent. There are numerous reasons for this. The most obvious is that safety-related engineering is guided by risk-based techniques – resources are focussed on those applications that are most likely to have a significant impact on safe and successful operation. Attention tends to focus on those areas that cause the greatest problems for operations rather than on areas that might be most vulnerable to cyber-attacks. Many companies also question the need to maintain logs which are very unlikely to be used given the relatively low reported frequency of cyber incidents, mentioned above. There is also a justified concern that the introduction of additional audit mechanisms will increase complexity and might undermine the resilience of safety-critical systems.

5 Information Diodes and the Threat from False Positives

Naedele [5] argues that IDS can undermine cyber situation awareness by increasing "confusion and operator stress in critical situations, for example if a malfunction in the plant causes a storm of alarm messages in the automation system which then again are interpreted as unusual by the IDS, causing additional alerts from the IDS". This is particularly important for whitelist approaches – where any unusual activity may be interpreted as a potential threat leading to a cascade of false positives. This 'alarm storm' is less of a concern for blacklist approaches; where degraded modes of behaviour are unlikely to match the signatures used to characterise existing forms of malware. As we have seen, however, blacklist approaches create significant concerns for validation and verification. In particular, it can be difficult to convince safety regulators that updates to a signature-based IDS will not introduce new failure modes or create vectors for the transmission of malware to isolated PLCs, smart controllers etc.

Information diodes provide an alternate approach. ISA 99 and IEC 62443 advocate the use of these devices to implement zoning of throughout Industrial Control Systems. One-way-diodes increase confidence that threats resident on the business systems cannot spread to the real time arena. However, information diodes can also be used to support intrusion detection. A one-way flow of data from the operational system is monitored for signs of malware using either a white or black list approach. The isolation of the IDS from the control system helps to reduce concerns that the detection system will have an adverse effect on safety-related processes and that the IDS updates may themselves provide an attack vector.

The use of these architectures does not provide a panacea for the security of SCADA and safety-related systems. Diodes cannot easily be deployed across the air gap architectures that have been described in previous sections. There are significant overheads in monitoring these applications and updating them for isolated PLCs and controllers distributed across production facilities. There are further concerns. Greater levels of monitoring may lead to an increasing number of false alarms, whereas in-

creased tolerances increase the potential for missed positives. Further work is urgently required to determine whether advanced visualisation techniques can be combined with, for instance, machine learning algorithms to ensure that IDS enhance rather than undermine the cyber-situation awareness of operators in complex safety-critical environments.

A host of remaining concerns remain to be addressed at the interface between safety and security. In particular, it is far from clear what measures should be taken one an intrusion has been detected. Existing guidance from the US Department of Justice and the UK Association of Chief Police Officers suggests that computational infrastructures should be treated as a crime scene [7]. Equipment should be switched off and no redundant or secondary systems should be enabled in case they destroy forensic evidence or extend an infection. In many industries this creates significant concerns that safety will be undermined in the aftermath of an incident. Further questions relate to the forensic analysis of SCADA systems, where extensive logs are not usually retained for thousands of isolated PLCs distributed across production facilities. In the same way that existing intrusion detection systems provide greatest support for office systems, there is a critical need to develop appropriate forensic tools that can be applied at the lower levels of many safety-critical infrastructures.

6 Conclusions

Intrusion detection systems help to detect malware in a range of software systems. White-list approaches enumerate the processes that can legitimately exploit system resources. Any other attempts to access those resources are interpreted to indicate the presence of malware. This paper has argued that whitelist techniques cannot easily be used in safety-related applications. Intellectual Property (IP) barriers associated with the extensive use of sub-contracting make it different to enumerate the resource requirements for all valid processes. Further concerns stem from the widespread use of legacy systems, where users have limited access either to the source code or to the original developers who can characterise the legitimate behaviour of their systems under a range of operating conditions. We have also identified practical and technical concerns over vulnerabilities that arise during the re-computation of

hash tables that implement whitelist techniques. We have also identified potential concerns over a loss of situation awareness when large numbers of unwarranted security alarms are triggered by degraded modes of operation.

In contrast, blacklist intrusion detection systems characterize the behaviour of known malware. In order to be effective, blacklist IDS must be updated at regular intervals. This raises enormous concerns in safety-critical systems where extensive validation and verification requirements ensure that software updates must be rigorously tested. In other words, there is a concern that an IDS signature update might itself introduce bugs into a safety-related system. Further concern stem from the difficulty of updating malware signatures in distributed control systems where an airgaps are often used to isolate low-level devices including PLCs.

Information diodes can be used to isolation an IDS from safety related applications. This reduces the likelihood that signature updates will cause the failure of a blacklist IDS or will cross-contaminate SCADA systems. Information diodes can also be used in whitelist approaches to ensure that the computational overhead of checking process permissions does not rob critical applications of much needed processing resources. However, further problems arise when IDS false positives compromise the reliability of safety-related applications. Greater levels of monitoring may lead to an increasing number of false alarms, whereas increased tolerances increase the potential for missed positives. These concerns can be addressed through, for example, machine learning techniques that adjust the tolerances of IDS to anomalous behaviour. It remains to be seen whether such techniques can meet regulatory requirements across a range of safety-related industries.

7 References

1. F. Adelstein, Live forensics: diagnosing your system without killing it first. Communications of the ACM, 49(2), 63–66, 2006.

2. I. Sutherland, J. Evans, T. Tryfonas and A. Blyth, Acquiring volatile operating system data tools and techniques. *SIGOPS Oper. Syst. Rev.*, *42*(3), 65–73, 2008.

3. European Network and Information Security Agency (ENISA), Technical Guidelines on Reporting Incidents: Article 13a Implementation, Heraklion, Greece, December 2011.

4. US Government Auditors Office, Information Security: FAA Needs to Address Weaknesses in Air Traffic Control Systems, GAO-15-221: Published: Jan 29, 2015.

5. M. Naedele, Addressing IT Security for Critical Control Systems, Proceedings of the 40th Hawaii International Conference on System Sciences, IEEE Computer Society, 2007.

6. C.W. Johnson, Anti-Social Networking: Crowdsourcing and the Cyber Defence of National Critical Infrastructures, Ergonomics, (57)3:419-433, 2014.

7. C.W. Johnson, Inadequate Legal, Regulatory and Technical Guidance for the Forensic Analysis of Cyber-Attacks on Safety-Critical Software. In Don Swallom (ed.), Proceedings of the 32nd International Systems Safety Society, Louisville, USA, International Systems Safety Society, Unionville, VA, USA, 2014.

8. U.S. National Institute of Standards and Technology (NIST), Computer Security Incident Handling Guide (Draft), Special Publication 800-61 Revision 2 (Draft), Gaithersburg, Maryland, 2012.

9. U.S. National Institute of Standards and Technology (NIST) (2006), Guide to Integrating Forensic Techniques into Incident Response, Special Publication 800-86, Gaithersburg, Maryland, 2006.

10. DigitalBond SCADA intrusion detection forum, last accessed March 2015. http://www.digitalbond.com/support-center/.

11. Australian Signals Directorate, *Application Whitelisting Explained.* Australian Government, Department of Defense. 2012.

12. D. Anderson and H. Khiabani, Protect Critical Infrastructure Computer Systems With Whitelisting, The SANS Institute, Bethesda, USA, July 2014.