Addressing IoT impact on software engineering

By Bill Graham, GrammaTech

Manufacturers need to carefully evaluate the cyber threats and the level of exposure of IoT devices. New levels of software integrity can only be achieved if teams can eliminate both accidental coding errors and intentional design-in vulnerabilities, through efficient analysis techniques suitable for the typical highly complex applications of today.

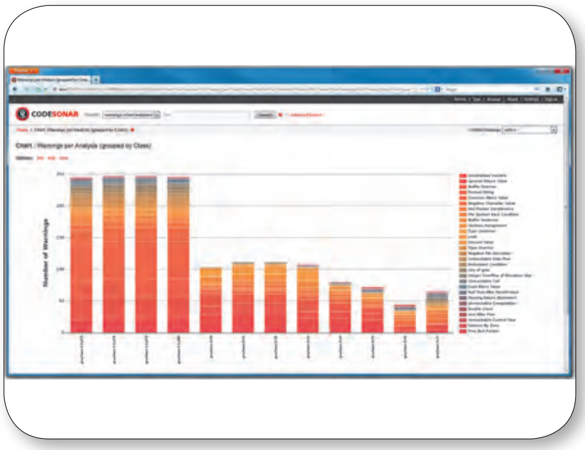

Figure 1. CodeSonar has been proven to provide the deepest static analysis, finding more critical defects than any other static analysis tool on the market

■ Powered by the forces of the cloud, connected endpoints, wireless technologies, and big data, the Internet of Things (IoT) evolution is forming a perfect storm for software engineering teams. This single, transformative force is bigger than anything in the history of tech industry, fueling an unparalleled consumer-oriented features race, expected to advance at an incredible rate over the next decade.

And why not? Vendors are racing to claim a piece of the predicted 8.9 trillion dollar IoT market by 2020, made up of more than 50 billion IoT devices spanning nearly all markets – automotive, energy/utilities, home appliance, consumer electronics, medical, education, manufacturing, and more. Although exciting to consumers and businesses alike, this race for IoT dominance also brings a significant dark side as embedded applications open communication to send and receive data between networks and other applications that may not always be known or friendly.

Current manufactures are still developing products using old and entrenched supply chain, engineering, and QA processes that weren't designed for the complexities of highly-connected smart devices today. Engineering teams are utilizing a progressively diverse set of suppliers and relying on third-party software to save time while trying to satisfy the business and market thirst for IoT demands. Unfortunately, many software development teams treat security in much the same way they have in the past, running only basic checks, if any, during their QA cycle.

This confluence of drivers – the lack of a security-first engineering philosophy, the increased use of third party software, and the continually growing time-to-market pressures from business executives complacent about IoT security – will continue to put software engineers in an ever-increasing tough spot, ripe for cyber criminals and nation states looking to exploit these connected devices and networks. These software vulnerabilities have already put consumer safety and privacy at risk, increasing corporate liabilities, eroding trust, and in some cases, shutting down critical public and industry services.

The fact of the matter is that today smart devices are anything but smart. One recent study found that 70% of the top 10 IoT smart devices are vulnerable to exploitation. The daily onslaught of news reports regarding new devices, appliances, and systems that have been hacked, includes stories that are quite terrifying, such as hackers remotely taking control of an automobile through its wireless hot spot connection and successfully commanding brakes and other critical systems. And unlike typical server applications that are housed in secured facilities with restricted access and controlled network connections, IoT devices (and their applications) are easily compromised. Allowing unprecedented time and access to isolate, reverse-engineer and repeatedly stress test for weaknesses. The reality of this is that if software engineers are unwilling or unable to find the cracks in their applications, someone else will. Possibly with bad intent.

So how do we evolve manufacturing processes to better protect our next-generation IoT devices? First, it starts with a sound plan that includes next-generation software assurance and a security first methodology. Teams need to rethink how they deliver software quickly – with security, safety, and quality in mind from design to deployment. However, rethinking should not be restarting. For example, trying to train every developer in the tactics hackers might use to exploit their software is not only impractical, but very time consuming. To do this successfully, teams should leverage the best tools available that help them analyze the software they are developing looking for problems that IoT presents – including both in-house source and third-party binary code. As IoT applications become more feature-rich, with additional elements of internet-connectivity and device intelligence, the risks of

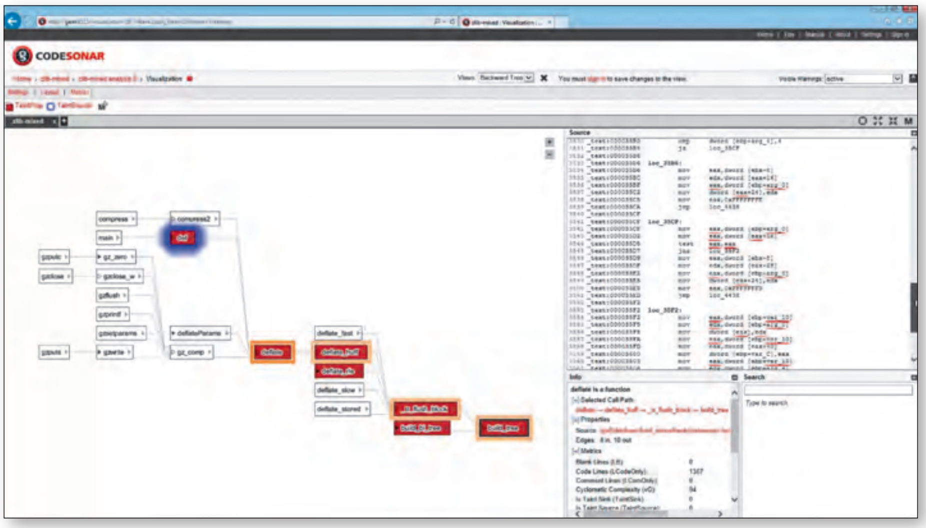

Figure 2. Taint Analysis allows to visualize notoriously hard-to-find tainted data pathways as an overlay on the code or superimposed on a graphical visualization of the program.

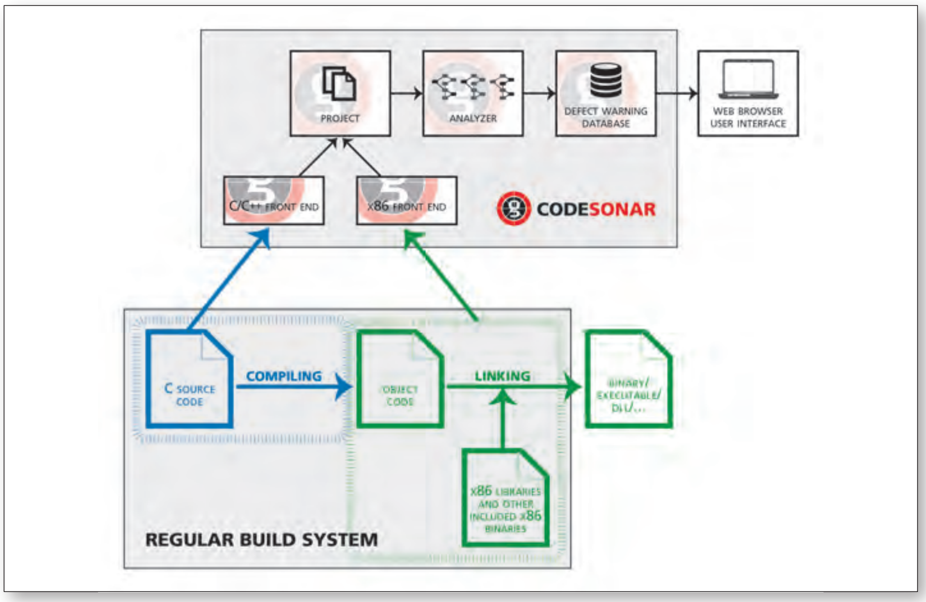

Figure 3. Due to the ever-rising amount of third-party code, strict check of third-party binaries is required. To realize this CodeSonar integrates binary code analysis.

built-in security vulnerabilities are increasing. Despite this trend, awareness of the risks associated with insecure code is still low among IoT developers and QA teams, and not a priority with most management teams.

Modern static-analysis tools are popular because they have proven to be effective, they are simple to introduce, and they can be used by development, QA, and security audit teams. Furthermore, in contrast to traditional dynamic testing, the code analyzed is never executed, so there is no additional test case development overhead and static analysis can be applied very early in the development process.

When programmers use static analysis as soon as code is written, bugs and security vulnerabilities can be found and eliminated even before the unit testing or integration testing phases begin. The earlier a defect is found, the cheaper it is to fix; this cost saving is a major advantage of automated static analysis.

Fortunately, static-analysis tools for source and binary have the ability to detect vulnerabilities before products are shipped, dramatically reducing security threats and corporate exposures that cost organizations several millions of dollars.

We've seen this numerous times in the recent news – with Toyota's unintended acceleration issue (estimated $3 billion in costs), in addition to the brand's first black eye; potential safety hazards with Jeep's recently-hacked Uconnect vulnerability, affecting over 470,000 vehicles; and the several SCADA systems recently hacked, most notably the Stuxnet exploitation, used to attack and destroy industrial equipment. It's simply unacceptable, and more importantly, easily avoidable, for development teams to not implement proper checks for security, reliability and hacking today. The added level of software assurance, which CodeSonar provides, can be easily deployed for the cost of a developer's morning coffee and a scone.

Over the last few years, third-party code has moved from a minor factor in software development to a dominant force in the industry. It is now used throughout software development in all applications, from highly sensitive government applications to security-intensive financial systems to safety-critical applications to consumer and mobile applications.

According to the latest report from VDC Research, the majority of software that runs on embedded devices is now developed by external sources, not in-house development teams. Some of this is open-source, but in embedded applications, nearly 30% of code is third-party commercial software – so the source is often unavailable. Such components include graphics toolkits, cryptography libraries, and communications middleware (network, USB, Bluetooth), which make up nearly 70% of the common embedded attack vectors.

GrammaTech, leveraging over 10 years of collaborative research, has developed a binary analysis capability to examine third-party code without requiring access to source code. This capability is fully integrated within our proven static analysis tool, CodeSonar, the first and only commercially-available binary analysis product.

CodeSonar's binary analysis technology provides developers with the ability to evaluate, check, and inspect third-party code, and provides businesses with more options within their supply chain, enabling them to utilize software from new, innovative companies that might not have an established reputation. When source code is available, you can use CodeSonar in mixed source/binary mode, analyzing your complete application.

The days of developing a standalone application are gone – the Internet of Everything has rapidly forced manufacturers to rethink how their products will support connected economy today, and changed the threat landscape forever. Nowadays reality is that there are educated attackers whose sole function is to break into IoT systems for many reasons, including fun, intellectual stimulation, profit, or worse, offensive attacks and terrorism. Today software development teams must adopt a robust secure design lifecycle, giving them the insights and capability to get it right first, to prevent these attackers from having a chance at breaking in. A general rule of thumb for teams to follow involves an end-to-end threat assessment (from a third-party audit team), security optimized designs, and security-scanning tools, of source and binaries. The Internet of Things is a paradigm that is impacting our daily life – for good and bad – today. IoT software needs security by design; for this reason, it must be a business imperative. Manufacturers must carefully evaluate the cyber threats and the level of exposure of IoT devices, implementing all the necessary design checks and countermeasures to respond to an accelerating set of menaces. GrammaTech was founded 26 years ago, with a firmly-grounded purposet o help organizations develop tomorrow's software. Given the ever-increasing dependence of software in today's connected world, our experts are focusing on solving the most challenging software issues through a thorough portfolio of software and security assurance solutions. IoT is here, it is our responsibility to ensure our software is ready for it. ■