TRUSTED CI
THE NSF CYBERSECURITY CENTER OF EXCELLENCE

Blog for Trusted CI.

Tuesday, August 3, 2021

Initial Findings of the 2021 Trusted CI Annual Challenge on Software Assurance

In 2021, Trusted CI is conducting our focused "annual challenge" on the assurance of software used by scientific computing and cyberinfrastructure. The goal of this year-long project, involving seven Trusted CI members, is to broadly improve the robustness of software used in scientific computing with respect to security. The Annual Challenge team spent the first half of the 2021 calendar year engaging with developers of scientific software to understand the range of software development practices used and identifying opportunities to improve practices and code implementation to minimize the risk of vulnerabilities. In this blog post, the 2021 Trusted CI Annual Challenge team gives a high-level description of some of its more important findings during the past six months.

Later this year, the team will be leveraging its insights from open-science developer engagements to develop a guide specifically aimed at the scientific software community that covers software assurance in a way most appropriate to that community. Trusted CI will be reaching back out to the community sometime in the Fall for feedback on draft versions of that guide before the final version is published late in 2021.

In support of this effort, Trusted CI gratefully acknowledges the input from the following teams who contributed to this effort: FABRIC, the Galaxy Project, High Performance SSH/SCP (HPN-SSH) by the Pittsburgh Supercomputing Center (PSC), Open OnDemand by the Ohio Supercomputer Center, Rolling Deck to Repository (R2R) by Columbia University, and the Vera C. Rubin Observatory.

At a high level, the team identified challenges that developers face with robust policy and process documentation; difficulties in identifying and staffing security leads, and ensuring strong lines of security responsibilities among developers; difficulties in effective use of code analysis tools; confusion about when, where, and how to find effective security training; and challenges with controlling source code developed and external libraries used, to ensure strong supply chain security. We now describe our examination process and findings in greater detail.

Goals and Approach

The motivation for this year's Annual Challenge is that Trusted CI has reviewed many projects in its history and found significant anecdotal evidence that there are worrisome gaps in software assurance practices in scientific computing. We determined that if some common themes could be identified and paired with the proportional remediations, the state of software assurance in science might be significantly improved.

Trusted CI has observed that currently available software development resources often do not match well with the needs of scientific projects; the backgrounds of the developers, the available resources, and the way the software is used do not necessarily map to existing resources available for software assurance. Hence, Trusted CI put together a team including a range of security expertise with backgrounds in the field from academic researchers to operational expertise. That team then examined several software projects covering a range of sizes, applications, and NSF directorate funding sources, looking for commonalities among them related to software security. Our focus was on both procedures and practical application of security measures and tools.

In preparing our examinations of these individual software projects, the Annual Challenge team enumerated several details that it felt would shed light on the software security challenges faced by

About Trusted CI

The mission of Trusted CI is to improve the cybersecurity of NSF computational science and engineering projects, while allowing those projects to focus on their science endeavors.

This mission is accomplished through one-on-one engagements with projects to solve their specific problems, broad education, outreach and training to raise the practice-of-security across the community, and looking for opportunities for improvement to bring in research to raise the state-of-practice.

For more information about what Trusted CI does, how it can help your project, the advances it is making in cybersecurity and resources for cybersecurity professionals, please see the Trusted CI website.

Tweets from @TrustedCI

Trusted CI Retweeted

REN-ISAC
@ren... · Sep 26

A voided lawsuit from a cyber insurance carrier claiming its customer misled it on its insurance application could potentially pave the way to change how underwriters evaluate self-attestation claims on insurance applications.

scientific software developers, some of the most successful ways in which existing teams are addressing those challenges, and observations from developers about the way that they wish things might be different in the future, or if they were able to do things over again from the beginning.

Findings

The Annual Challenge team's findings are generally aligned with one of five categories: process, organization/mission, tools, training, and code storage.

Process: The team found several common threads of challenges facing developers, most notably related to policy process documentation, including policies relating to onboarding, offboarding, code commits and pull requests, coding standards, design, communication about vulnerabilities with user communities, patching methodologies, and auditing practices. One cause for this finding is often that software projects start small and do not plan to grow or be used widely. And when the software does grow and starts to be used broadly, it can be hard to develop formal policies after developers are used to working in an informal, ad hoc manner. In addition, organizations do not budget for security. Further, where policy documentation does exist, it can easily go stale -- "documentation rot." As a result, it would be helpful for Trusted CI to develop guides for and examples of such policies that could be used and implemented even at early stages by the scientific software development community.

Organization and Mission: Most projects faced difficulties in identifying, staffing, or funding a security lead and/or project manager. The few projects that had at least one of these roles filled had an advantage in regards to DevSecOps. In terms of acquiring broader security skills, some projects attempted to use institutional "audit services" but found mixed results. Several projects struggled with the challenge of integrating security knowledge among different teams or individuals. Strong lines of responsibility can create valuable modularity but can also create points of weakness when interfaces between different authors or repositories are not fully evaluated for security issues. Developers can ease this tension by using processes for developing security policies around software, ensuring ongoing management support and enforcement of policies, and helping development teams understand the assignment of key responsibilities. These topics will be addressed in the software assurance guide that Trusted CI is developing.

Tools: Static analysis tools are commonly employed in continuous integration (CI) workflows to help detect security flaws, poor coding style, and potential errors in the project. A primary attribute of a static analysis tool is the set of language-specific rules and patterns it uses to search for style, correctness, and security issues in a project. One major issue with static analysis tools is that they report a high number of false positives, which, as the Trusted CI team found, can cause developers to avoid using them. It was determined that it would be helpful for Trusted CI to develop tutorials that are appropriate for the developers in the scientific software community to learn how to properly use these tools and overcome their traditional weaknesses without being buried in useless results.

The Trusted CI team found that dependency checking tools were commonly employed, particularly given some of the automation and analysis features built into GitHub. Such tools are useful to ensure the continued security of a project's dependencies as new vulnerabilities are found over time. Thus, the Trusted CI team will explore developing (or referencing existing) materials to ensure that the application of dependency tracking is effective for the audience and application in question. It should be noted that tools in general could give a false sense of security if they are not carefully used.

Training: Projects shared that developers of scientific software received almost no specific training on security or secure software development. A few of the projects that attempted to find online training resources reported finding themselves lost in a quagmire of tutorials. In some cases, developers had computer science backgrounds and relied on what they learned early in their careers, sometimes decades ago. In other cases, professional training was explored but found to be at the wrong level of detail to be useful, had little emphasis on security specifically, or was extremely expensive. In yet other cases, institutional training was leveraged. We found that any kind of ongoing training tended to be seen by developers as not worth the time and/or expense. To address this, Trusted CI should identify training resources appropriate for the specific needs, interests, and budgets of the scientific software community.

Code Storage: Although most projects were using version control in external repositories, the access controls methods governing pull requests and commits were not sufficiently restricted to maintain a secure posture. Many projects leverage GitHub's dependency checking software; however, that tool is limited to only checking libraries within GitHub's domain. A few projects developed their own software in an attempt to navigate a dependency nightmare. Further, there was often little ability or attempt to vet external libraries; these were often accepted without inspection mainly because there is no straightforward mechanism in place to vet these packages. In the Trusted CI software assurance guide, it would be useful to describe processes for leveraging two-factor authentication and developing policies governing access controls, commits, pull requests, and vetting of external libraries.

darkreading.com
Cyber Insurers Clamp Down on Clients' Self-Attestation of ...

Blog Archive

Next Steps

The findings derived from our examination of several representative scientific software development projects will direct our efforts towards addressing what new content we believe is most needed by the scientific software development community.

Over the next six months, the Trusted CI team will be developing a guide consisting of this material, targeted toward anyone who is either planning or has an ongoing software project that needs a security plan in place. While we hope that the guide will be broadly usable, a particular focus of the guide will be on projects that provide a user-facing front end exposed to the Internet because such software is most likely to be attacked.

This guide is meant as a “best practices” approach to the software lifecycle. We will recommend various resources that should be leveraged in scientific software, including the types of tools to run to expose vulnerabilities, best practices in coding, and some procedures that should be followed when engaged in a large collaborative effort and how to share the code safely. Ultimately, we hope the guide will support scientific discovery itself by providing guidance around how to minimize possible risks incurred in creating and using scientific software.

Posted by Sean Peisert at 11:37 AM

Labels: annual challenge, software assurance

Newer Post Home Older Post

► 2016 (41)

► 2015 (13)

► 2014 (32)

► 2013 (18)

► 2012 (4)

Search This Blog

Search

Labels

webinar (85) engagements (84) events (42) NSF Summit (41) iam (32) Trusted CI (27) vulnerabilities (23) framework (21) cybersecurity programs (19) compliance (18) situational-awareness (16) software assurance (16) TTP (15) large facilities (15) major facilities (14) Fellows (13) trustworthy data (13) CyberCheckup (11) PEARC (11) presentations (11) project-news (11) reports (11) science gateways (11) success story (10) CUI (9) Internet2 (9) oscrp (9) annual challenge (8) engagement-cfp (8) identity federation (8) incident response (8) COVID-19 (7) Survey (7) cybertraining (7) ransomware (7) secure coding (7) solicitations (7) CMMC (6) ESnet (6) NSF-cybersecurity-guide (6) students (6) OSG (5) ResearchSOC (5) authentication (5) incommon (5) open source software (5) openssl (5) software sustainability (5) tutorial (5) working group (5) BD Hubs (4) Cloud-computing (4) Cybersecurity (4) DKIST (4) FABRIC (4) Jupyter (4) benchmarking (4) cici (4) cyberinfrastructure (4) data assurance (4) idm (4) jobs (4) network (4) news (4) ARF (3) CERN (3) CPP (3) HPC (3) LSST (3) NCSA (3) Pegasus (3) REED+ (3) advisory committee (3) blockchain (3) controls (3) educause (3) epoc (3) law and policy (3) office hours (3) ren-isac (3) video conferencing (3) xsede (3) AMNH (2) AoT (2) EDI (2) Gemini Observatory (2) GenApp (2) Globus (2) NEON (2) NRAO (2) NSF Summit Survey (2) OSiRIS (2) SLATE (2) Science DMZs (2) Skim Reaper (2) TransPac (2) Trusted CI Vision (2) UC Berkeley (2) UNH-RCC (2) USAP (2) WISE (2) ask@trustedci.org (2) cilogon (2) cyber-physical systems (2) higher education (2) ligo (2) log analysis (2) operational technology (2) racial inequities (2) research computing (2) risk (2) trust community (2) 2021 Summit report (1) 2022 Jean-Claude Laprie Award in Dependable Computing (1) Bart Miller (1) CENIC (1) CI CoE (1) CI Compass (1) EPSCoR (1) IEEE/IFIP International Conference on Dependable Systems and Networks (1) JASON (1) NOIRLab (1) NSPM-33 (1) Ocean Sciences (1) White House (1) cohort (1) cpe (1) cyberattacks (1) cybercrime (1) cybersecurity program (1) engagement (1) research (1)

Awesome Inc. theme. Powered by Blogger.