CHAINIAC: Proactive Software-Update Transparency via Collectively Signed Skipchains and Verified Builds

Kirill Nikitin, Eleftherios Kokoris-Kogias, Philipp Jovanovic, Nicolas Gailly, and Linus Gasser, *École polytechnique fédérale de Lausanne* **(*EPFL*);** **Ismail Khoffi,** *University of Bonn;* **Justin Cappos,** *New York University;* **Bryan Ford,** *École polytechnique fédérale de Lausanne* **(*EPFL*)**

https://www.usenix.org/conference/usenixsecurity17/technical-sessions/presentation/nikitin

This paper is included in the Proceedings of the 26th USENIX Security Symposium

August 16–18, 2017 • Vancouver, BC, Canada

ISBN 978-1-931971-40-9

Open access to the Proceedings of the 26th USENIX Security Symposium is sponsored by USENIX

CHAINIAC: Proactive Software-Update Transparency via Collectively Signed Skipchains and Verified Builds

Kirill Nikitin[1], Eleftherios Kokoris-Kogias[1], Philipp Jovanovic[1], Linus Gasser[1], Nicolas Gailly[1], Ismail Khoffi[2], Justin Cappos[3], and Bryan Ford[1]

[1]École polytechnique fédérale de Lausanne (EPFL)
[2]University of Bonn
[3]New York University

Abstract

Software-update mechanisms are critical to the security of modern systems, but their typically centralized design presents a lucrative and frequently attacked target. In this work, we propose CHAINIAC, a decentralized software-update framework that eliminates single points of failure, enforces transparency, and provides efficient verifiability of integrity and authenticity for software-release processes. Independent *witness servers* collectively verify conformance of software updates to release policies, *build verifiers* validate the source-to-binary correspondence, and a tamper-proof release log stores collectively signed updates, thus ensuring that no release is accepted by clients before being widely disclosed and validated. The release log embodies a *skipchain*, a novel data structure, enabling arbitrarily out-of-date clients to efficiently validate updates and signing keys. Evaluation of our CHAINIAC prototype on reproducible Debian packages shows that the automated update process takes the average of 5 minutes per release for individual packages, and only 20 seconds for the aggregate timeline. We further evaluate the framework using real-world data from the PyPI package repository and show that it offers clients security comparable to verifying every single update themselves while consuming only one-fifth of the bandwidth and having a minimal computational overhead.

1 Introduction

Software updates are essential to the security of computerized systems as they enable the addition of new security features, the minimization of the delay to patch disclosed vulnerabilities and, in general, the improvement of their security posture. As software-update systems [17, 24, 34, 35, 48] are responsible for managing, distributing, and installing code that is eventually executed on end systems, they constitute valuable targets for attackers who might, *e.g.*, try to subvert the update infrastructure to inject malware. Furthermore, powerful adversaries might be able to compromise a fraction of the developers' machines or tamper with software-update centers. Therefore, securing update infrastructure requires addressing four main challenges:

First, the integrity and authenticity of updates traditionally depends on a single signing key, prone to loss [53] or theft [29, 32, 70]. Having proper protection for signing keys to defend against such single points of failure is therefore a top priority. Second, the lack of transparency mechanisms in the current infrastructure of software distribution leaves room for equivocation and stealthy backdooring of updates by compromised [15, 46], coerced [11, 28, 66], or malicious [36] software vendors and distributors. Recent work on reproducible software builds [49, 59] attempts to counteract this deficit by improving on the source-to-binary correspondence. However, it is unsuitable for widespread deployment in its current form, as rebuilding packages puts a high burden on end users (*e.g.*, building the Tor Browser bundle takes 32 hours on a modern laptop [60]). Third, attackers might execute a man-in-the-middle attack on the connections between users and update providers (*e.g.*, with DNS cache poisoning [67] or BGP hijacking [6]), thus enabling themselves to mount replay and freeze attacks [15] against their targets. To prevent attackers from exploiting unpatched security vulnerabilities as a consequence of being targeted by one of the above attacks [72], clients must be able to verify timeliness of updates. Finally, revoking and renewing signing keys (*e.g.*, in reaction to a compromise) and informing all their clients about these changes is usually cumbersome. Hence, modern software-update systems should provide efficient and secure means to evolve signing keys and should enable client notification in a timely manner.

To address these challenges, we propose CHAINIAC, a decentralized software-update framework that removes