Review article
UDK 502.131.1:005.5
DOI 10.7251/BLCZR0126665S

LINUX ECOSYSTEM SECURITY AND THE RISKS OF USER REPOSITORIES: A CASE STUDY OF THE "ATOMIC ARCH" ATTACK ON THE ARCH USER REPOSITORY

Aleksandar Šijan[1]
Luka Ilić[2]
Dejan Viduka[3]

Keywords: *Linux, Arch Linux, AUR, supply chain, trust, malware, open source*

Abstract: *Open-source software and Linux form the backbone of modern server and development infrastructure, and the focus of security is increasingly shifting from individual software components to the supply chain through which software reaches its users. This paper reviews the security of the Linux ecosystem, with particular attention to package repositories as points of trust. After a general overview, it focuses on the Arch Linux distribution and its community-driven Arch User Repository (AUR), explaining how it works, its advantages, and its security weaknesses. As an illustration of these weaknesses, the paper analyses in detail the June 2026 "Atomic Arch" supply-chain attack, in which attackers compromised approximately 1,500 AUR packages by adopting orphaned packages and modifying their build scripts, delivering credential-stealing malware. The analysis, grounded in existing taxonomies of software supplychain attacks, shows that the essence of the attack was not a technical vulnerability but the abuse of a trust model based on package name and history. The paper also proposes multi-layered defence measures, concluding that the security of user repositories requires shifting trust from package identity to verification of the actual content before it is executed.*

BEZBJEDNOST LINUX EKOSISTEMA I RIZICI KORISNIČKIH REPOZITORIJUMA: STUDIJA SLUČAJA NAPADA „ATOMIC ARCH" NA ARCH USER REPOSITORY

Ključne riječi: *Linux, Arch Linux, AUR, lanac snabdijevanja, povjerenje, zlonamjerni softver, otvoreni kod*

Sažetak: *Softver otvorenog koda i Linux predstavljaju okosnicu savremene serverske i razvojne infrastrukture, a fokus bezbjednosti sve se više pomjera sa pojedinačnih softverskih komponenti*

1 Faculty of Applied Management, Economics and Finance, University Business Academy in Novi Sad, Belgrade, Serbia
2 Faculty of Applied Management, Economics and Finance, University Business Academy in Novi Sad, Belgrade, Serbia
3 Faculty of Information Technology, Alfa BK University, Belgrade, Serbia

na lanac snabdijevanja putem kojeg softver dolazi do korisnika. U radu se razmatra bezbjednost Linux ekosistema, s posebnim osvrtom na repozitorijume paketa kao tačke povjerenja. Nakon opšteg pregleda, pažnja je usmjerena na distribuciju Arch Linux i njen zajednički održavan Arch User Repository (AUR), uz objašnjenje načina njegovog funkcionisanja, prednosti i bezbjednosnih slabosti. Kao ilustracija tih slabosti, detaljno se analizira napad na lanac snabdijevanja „Atomic Arch" iz juna 2026. godine, u kojem su napadači kompromitovali približno 1.500 AUR paketa preuzimanjem napuštenih paketa i izmjenom njihovih skripti za izgradnju, čime je distribuiran zlonamjerni softver za krađu pristupnih podataka. Analiza, zasnovana na postojećim taksonomijama napada na softverski lanac snabdijevanja, pokazuje da suština napada nije bila tehnička ranjivost, već zloupotreba modela povjerenja zasnovanog na nazivu i istoriji paketa. Rad takođe predlaže višeslojne mjere odbrane i zaključuje da bezbjednost korisničkih repozitorijuma zahtijeva pomjeranje povjerenja sa identiteta paketa na provjeru stvarnog sadržaja prije njegovog izvršavanja.

1. Introduction

Linux and open source software form the backbone of modern digital infrastructure - from servers and the cloud to development and CI/CD environments. With that ubiquity the nature of security threats changes as well: the centre of gravity is shifting from vulnerabilities in the software itself to the supply chain, that is, to the paths by which software and its updates reach users (Aslan et al., 2023). The year 2026 has been marked by a series of supply-chain attacks aimed precisely at development ecosystems and package repositories (StepSecurity, 2026).

The scale of that dependency today is such that modern applications no longer consist for the most part of code written by their own authors, but of open source packages pulled from public repositories. The security of an individual system therefore depends increasingly on the integrity of the entire chain through which that code arrives. Isolated incidents such as the compromise of the SolarWinds Orion platform, downloaded by around 18,000 customers including government institutions and critical infrastructure operators, have demonstrated the reach and potential impact of supply-chain attacks (Ladisa et al., 2023). Their economic and infrastructural consequences single out supply-chain security as one of the central questions of contemporary cybersecurity.

Linux occupies a particular place in this context, since it dominates server and cloud environments and underpins most automated build and CI infrastructure (W3Techs, 2026), which has made developers and automated build environments an attractive target of attack.

Within the Linux ecosystem, package repositories and package managers hold the central role in that chain and constitute its main points of trust: the user believes that software obtained through an official mechanism is what it claims to be. This paper reviews the security of that model, focuses on the Arch Linux distribution and its user repository, the AUR, and uses the concrete example of the "Atomic Arch" attack of June 2026 to show how trust in a repository can be abused. It is conceived as a review paper with a case study and closes with a proposal of layered defensive measures.

2. Security of the Linux Ecosystem

Open-source software carries an assumption of transparency: because the source code is available, vulnerabilities and malicious modifications can be caught by inspection. In

practice, however, the assumption that "many eyes" will notice every malicious change is empirically limited - most packages, particularly those in repositories to which users publish directly, receive little or no independent review (Ohm et al., 2020). Trust is therefore transferred de facto to intermediaries: package maintainers and repositories. The package manager thereby becomes a critical security component, since it determines where software is fetched from, whether its authenticity is verified, and with what privileges it is installed.

Linux is in this respect a particularly valuable target: it dominates server and cloud environments, and its security is organized in layers - from the kernel, through system services, to the package layer. While the kernel and core services are subject to intensive auditing and hardening, the package distribution layer has become a relatively weaker link, because it rests on trust in sources and maintainers rather than on the technical correctness of the code alone.

Supply chain attacks are recognized as a distinct and systematized class of threat. Ladisa et al. (2023) propose an ecosystem-agnostic taxonomy covering 107 distinct attack vectors linked to 94 real-world incidents, identifying package repositories as one of the most frequently attacked surfaces. Unlike unintentional vulnerabilities, malicious packages are deliberately injected into the chain, so conventional vulnerability detection tools are not sufficient (Ohm et al., 2020).

By level of vetting, repositories divide broadly into official ones, in which packages are reviewed and signed by trusted maintainers, and user repositories, to which the community itself contributes content with limited or no formal validation. The dependency architecture introduces further risk of its own: package managers automatically resolve, download, and install large numbers of packages over the software lifecycle, multiplying the combined attack surface of a project (Ohm et al., 2020). The build and installation step is precisely the privileged moment at which code executes on the user's system, which makes it a natural target of attack (Ferdous et al., 2023).

Distributions also introduce technical integrity mechanisms (package signing and checksums) which confirm that a package originates from the expected repository and has not been altered in transit. These mechanisms do not, however, guarantee that the content of the repository is itself benign: if a legitimate channel is compromised, a signed malicious package will pass verification. The weight of trust thus returns to the human and governance layer, to the integrity of maintainers and of the process that approves what enters the repository, which is exactly the point exploited by the attack examined in this paper.

3. Arch Linux

Arch Linux is a distribution built on the principle of simplicity and user control, with a rolling release model that provides access to the latest software versions. Package management rests on the pacman package manager and on the official repositories (core, extra, and multilib) whose packages pass through a process of review and maintenance by trusted maintainers (Arch Linux, n.d.-b). Derived distributions such as Manjaro and EndeavourOS rest on the same model.

The rolling release model brings a security trade-off of its own: rapid delivery of patches narrows the window of exposure to known vulnerabilities, but constant package churn makes auditing harder and increases reliance on trust in sources. The official repositories

mitigate this through package signing and a controlled maintenance process, thereby securing the integrity and provenance of content. Alongside them, the Arch ecosystem is distinguished by the AUR user repository, whose markedly different security profile was at the centre of the attack examined here.

4. Arch User Repository (AUR)

The AUR is a repository maintained by the community that contains no ready-made binary packages, only build scripts (so-called PKGBUILD files) for software not available in the official repositories (Arch Linux, n.d.-a). Users, most often by way of AUR helpers such as yay or paru, fetch the PKGBUILD and build the package locally on their own system. This means that any logic contained in the build script executes directly on the user's machine, during build or installation, which is the defining security characteristic of this model.

A distinctive AUR mechanism is package adoption: when a maintainer abandons a package, another user may request to adopt it, inheriting its name, its history, and the reputation it has accumulated. This mechanism keeps useful software alive, but it also opens a channel for the quiet change of ownership of a trusted package.

Advantages

The main advantages of the AUR are an enormous selection of software (tens of thousands of packages, including many that would never reach the official repositories), fast access to the latest versions, a low barrier to contribution, and a strong community that keeps packages current. For many users the AUR is one of the most attractive features of the Arch ecosystem.

Security Weaknesses

The very properties that make the AUR useful also make it vulnerable. Packages are not formally validated and the Arch documentation itself warns explicitly that they are used at the user's own risk(Arch Linux, n.d.-a). Code executes at build time with the user's privileges, and trust rests in practice on the name and history of a package rather than on verification of the identity and intentions of its current maintainer. This makes the AUR fertile ground for supply-chain attacks, in which the software itself is not attacked but rather the trust placed in the channel through which it arrives.

Aspect	*Official repositories*	*AUR*
Content contribution	Trusted maintainers	Any user
Pre-publication vetting	Review and signing	No formal validation
Form of distribution	Ready-made binary packages	PKGBUILD build scripts
Code execution	Controlled environment	On the user's machine, at build time
Basis of trust	Maintainer identity	Package name and history
Supply-chain exposure	Lower	Higher

Table 1. Comparison of the Official Repositories and the AUR

These weaknesses are not merely theoretical. The AUR mechanism maps directly onto already catalogued attack vectors such as account takeover or the injection of a malicious

package into an ecosystem (Ladisa et al., 2023). A similar pattern - adoption of an abandoned package in order to exploit its accumulated trust was recorded on the AUR as early as 2018, on an abandoned PDF viewer, while in 2025 Arch was subjected to a multi-day denial of service attack as well as to compromised web browser packages carrying a remote access tool (The Hacker News, 2026).

5. Case Study: The "Atomic Arch" Attack (June 2026)

Discovery and Scope

The attack was discovered on 11 June 2026, when researchers at Sonatype and the Arch Linux community, via the aur-general mailing list, warned of mass malicious adoption and modification of packages in the AUR (Sonatype, 2026). The campaign, named "Atomic Arch," initially covered around 400 packages, expanding over the weekend to approximately 1,500, with some tracking lists citing more than 1,570 (The Hacker News, 2026). Sonatype tracked the campaign under the identifier Sonatype-2026-003775, with a severity rating of CVSS 8.7; no formal CVE was assigned, and the attack was not attributed to a specific actor.

Date	*Event*
11 June 2026	Discovery and disclosure (Sonatype, aur-general list); around 400 packages; npm atomic-lockfile
12 June 2026	Second wave (Bun paths, js-digest); scope grew to around 1,500 packages
12 June 2026	Official Arch announcement; account creation, updates, and package adoption restricted
Thereafter	Malicious commits reverted, accounts blocked, list of affected packages, detection tooling

Table 2. Timeline of the "Atomic Arch" Campaign Date Event

Attack Mechanism

The attackers did not exploit a technical vulnerability but the very mechanism for taking over abandoned packages. By adopting orphaned packages they gained control over their build scripts while retaining the original name, history, and reputation of each package. They then modified the PKGBUILD files and install hooks so that, at build time, these would fetch and run malicious npm/Bun packages (atomiclockfile, js-digest, and lockfile-js) whose install hook immediately executed a hidden binary (Sonatype, 2026; StepSecurity, 2026). Characteristically, the application code itself was never modified; only the build "recipe" was changed, so from the user's perspective the package looked entirely ordinary. The attackers even forged git commit metadata so that the changes would appear to originate from a longstanding maintainer, although the original account had not been compromised. A day later, on 12 June, a second wave followed that also used Bun-based installation paths (The Hacker News, 2026).

Payload

The main payload was an infostealer written in Rust, targeted specifically at development secrets - browser cookies and sessions, SSH keys, API tokens, and cloud access keys. The attack was thus deliberately aimed at developers and CI environments rather than at the data of incidental users.

Optionally, on systems with root privileges, the payload could also load an eBPF rootkit; it is worth emphasizing that this was not used to gain privileges but only for concealment - hiding processes, their names, and network descriptors, and blocking debugging (The Hacker News, 2026). Analyses also pointed to a possible cryptocurrency miner staged for later delivery.

Scope of Impact

The official Arch repositories (core, extra, and multilib) remained untouched, as they are subject to a stricter review process. Those affected were users of Arch and of distributions derived from it (such as Manjaro and EndeavourOS) who used the AUR, as well as self-hosted CI runners on those systems. The attack did not directly affect GitHub-hosted runners or distributions that do not use the AUR.

Response and Remediation

The Arch Linux team responded quickly: malicious commits were reverted, attacker accounts permanently blocked, and a list of affected packages published. At the same time, in order to halt the campaign, the creation of new accounts, the publication of updates, and package adoption were temporarily restricted (Arch Linux, 2026). Users were advised to list all AUR packages with pacman - Qm and compare them against the published list, to review the history of PKGBUILD changes between 10 and 12 June, and to rotate all credentials - passwords, SSH keys, and API tokens. Because the payload had rootkit capabilities, systems on which it had executed with root privileges were also advised to be reinstalled, since removing the package does not guarantee that the machine is clean (The Hacker News, 2026).

Wider Context

The "Atomic Arch" attack is not an isolated case but part of a broader wave of supply-chain attacks through development ecosystems. The structural risks of open repositories have been documented before: Ohm et al. (2020) analyse 174 malicious packages distributed via npm, PyPI, and RubyGems, while Zimmermann et al. (2019) show that the densely interwoven dependency network of the npm ecosystem allows the compromise of a single package to endanger thousands of others. "Atomic Arch" relied on precisely that network, using npm and Bun packages as payload carriers, and the same pattern recurs in other 2026 incidents targeting npm packages and CI/CD infrastructure (StepSecurity, 2026).

5. Defence Mechanisms and Recommendations

Defending against attacks such as "Atomic Arch" requires measures at several levels - from the individual user, through the repository itself, to technical mechanisms for chain

integrity. Table 3 maps the identified weaknesses of the AUR model onto the corresponding defensive measures.

Weakness	*Defensive measure*
Unvalidated content	Review PKGBUILD and install scripts before building; prefer the official repositories
Code execution at build time	Build in an isolated environment; automated script scanning
Trust by package name	Check the current maintainer; caution with recently adopted packages
Takeover of abandoned packages	Stricter vetting of adoption at the repository level
Lack of integrity verification	Signing and reproducible builds
Credential exposure	Routine credential rotation; separate development and CI environments

Table 3 Weaknesses of the AUR Model and Corresponding Defensive Measures

At the systemic level, supply-chain integrity is further strengthened by reproducible builds, which allow independent verification that a delivered artifact genuinely corresponds to its source code (Lamb & Zacchiroli, 2022), and by automated scanning of packages and scripts to detect malicious patterns (Ohm et al., 2020; Ladisa et al., 2023). A broader foundation is provided by the practice of maintaining a software bill of materials (SBOM) and by provenance verification, which make it possible to trace the origin of every embedded element and to establish exposure quickly in the event of an incident.

6. Discussion and Lessons

The essential lesson of "Atomic Arch" concerns the trust model itself. Unlike classic supply-chain attacks, which must first build trust (typosquatting through similar names, or brandjacking through false familiarity) this attack inherits it. The attackers did not create a new, suspicious package; they took over existing, already trusted ones, thereby removing precisely the warning signals users normally rely on (Sonatype, 2026). Viewed through existing taxonomies, "Atomic Arch" falls under attacks on distribution through the abuse of a legitimate channel, which confirms that established frameworks describe such novel incidents well (Ladisa et al., 2023).

From this follows the direction of defence: trust must shift from package identity (name and history) to verification of the content itself before execution. Recommended measures include reviewing PKGBUILD and install scripts before every build, particular caution toward recently adopted packages or those that suddenly acquire new install hooks, preference for the official repositories wherever possible, consistent credential rotation hygiene, and a tightening of the adoption process for abandoned packages at the repository level. Since incidents of this kind are not isolated but recur and spread across multiple ecosystems, a durable solution must be systemic rather than merely reactive.

7. Conclusion

The "Atomic Arch" case shows that the greatest strength of user repositories such as the AUR - openness, community, and reputation-based trust, is at the same time their greatest weakness. The security of such repositories is not a matter of an individual vulnerability but an enduring problem of trust governance, in which the build and installation step remains a critical privileged moment.

A durable solution entails a shift from identity-based trust to content-based trust: stricter vetting of package adoption, automated scanning of build scripts, reproducible builds, and raising user awareness that obtaining software from a "trusted" source is no substitute for verifying what is actually executed.

The security of the Linux ecosystem, like security in general, is thereby confirmed as a continuous process rather than a one-off state.

Literature

- Arch Linux. (n.d.-a). Arch User Repository. ArchWiki. Retrieved August 24, 2026, from https://wiki.archlinux.org/title/Arch_User_Repository Arch Linux. (n.d.-b). Official repositories. ArchWiki. Retrieved August 24, 2026, from https://wiki.archlinux.org/title/Official_repositories Arch Linux. (2026, June 12). Active AUR malicious packages incident. Arch Linux News. https://archlinux.org/news/active-aur-malicious-packages-incident/
- Aslan, Ö., Aktuğ, S. S., Ozkan-Okay, M., Yilmaz, A. A., & Akin, E. (2023). A comprehensive review of cyber security vulnerabilities, threats, attacks, and solutions. Electronics, 12(6), 1333. https://doi.org/10.3390/electronics12061333
- Ferdous, J., Islam, R., Mahboubi, A., & Islam, M. Z. (2023). A state-of-the-art review of malware attack trends and defense mechanisms. IEEE
- Access, 11, 121118-121141. https://doi.org/10.1109/ACCESS.2023.3328351
- Ladisa, P., Plate, H., Martinez, M., & Barais, O. (2023). SoK: Taxonomy of attacks on open-source software supply chains. In 2023 IEEE Symposium on Security and Privacy (SP) (pp. 1509-1526). IEEE. https://doi.org/10.1109/SP46215.2023.10179304
- Lamb, C., & Zacchiroli, S. (2022). Reproducible builds: Increasing the integrity of software supply chains. IEEE Software, 39(2), 62-70. https://doi.org/10.1109/MS.2021.3073045
- Ohm, M., Plate, H., Sykosch, A., & Meier, M. (2020). Backstabber's Knife Collection: A review of open source software supply chain attacks. In Detection of Intrusions and Malware, and Vulnerability Assessment (DIMVA 2020) (LNCS Vol. 12223, pp. 23-43). Springer. https://doi.org/10.1007/978-3-030-52683-2_2
- Sonatype. (2026, June 11). Atomic Arch: Attackers hijack trusted AUR packages to deliver malware. Sonatype Blog. https://www.sonatype.com/blog/atomic-arch-npm-campaign-adds-maliciousdependency
- StepSecurity. (2026, June). 400+ AUR packages hijacked: What the "Atomic Arch" campaign means for supply-chain security. StepSecurity Blog. https://www.stepsecurity.io/blog/400-aur-packageshijacked-atomic-arch-campaign The Hacker News. (2026, June 12). Over 400 Arch Linux AUR packages hijacked to deploy infostealer and eBPF rootkit. https://thehackernews.com/2026/06/over-400-arch-linux-aurpackages.html
- Zimmermann, M., Staicu, C.-A., Tenny, C., & Pradel, M. (2019). Small world with high risks: A study of security threats in the npm ecosystem. In 28th USENIX Security Symposium (USENIX Security 19) (pp. 995-1010). USENIX Association.