Artemis: Defanging Software Supply Chain Attacks in Multi-repository Update Systems

Marina Moore, NYU
Use of multiple repositories

A software repository distributes packages containing software libraries or applications.

Software is downloaded by software installation tools.

Top 10 Linux distributions have average of 4.8 default repositories.
Articulated Trust

Allow software installation tools to specify trusted developers and repositories for each package

Selective trust in developers and repositories
Artemis

Security framework that implements articulated trust

Extends the functionality of role-based access control (RBAC) models
Limitations of existing uses of multiple repositories
Dependency Confusion

Many companies use both a public repository and a private, internal repository

Company downloads package foo from the private repository

Attack:

- Upload package named foo to the public repository
  - Version number greater than the internal version

Requirement: per-package prioritization of repositories
Only want some packages from a repository

May not want all packages on a public repository

- Malicious versions through hijacked accounts
- Undertested/lower quality code

Requirement: Defining a trusted subset
Fallback problem

If one repository is unavailable, the installation tool will fallback to other repositories

May want some packages from particular repositories

Requirement: terminate search for a package
Repository compromise

Repository compromise is common

Attacker can replace any package signed with keys on the repository

Requirement: Mitigate repository compromise
Maintainer compromise

Maintainer compromise and protestware happen frequently

These attacks can be recovered from by revoking compromised maintainers

But not prevented

Requirement: Mitigate role compromise
Real-world use

Requirement: Shareable configuration

Requirement: preserve backwards compatibility with existing systems

Requirement: mechanisms added must not significantly affect performance
Threat model

Attacker can:

- Respond to user requests
- Compromise one or more keys
- Use compromised keys to perform arbitrary software attacks
- Upload an arbitrary package to an unused name on a public repository

Goal:

- Do not install less-prefered or arbitrary package
- Compromise resilience
Artemis

Targets map file

Repository map file

User configuration

Repository A

Alice

targets

foo.img

bar.img

Repository B

Bob

testing

bar.img

Charlie

foo.img
Multi-role Delegations

- **targets**
  - Multi-role delegation
  - Signs hashes and length

- **Bob**

- **3 out of 5 keys testing**

- **Ubuntu-16.10.img**
Key pinning

```
"targets_rolename": Django,
"threshold": 1,
"keys": { A B C D }
```
Repository RBAC
Artemis
Implementation

Processing time:
- 210 ms
- 38% overhead

Storage:
- 10.3 KB
- 0.34% overhead
Analysis of past attacks

Attacks from CNCF Catalog of Supply Chain Compromises

- Repository Compromise
- Compromised developer key
- Compromised key and repository
- Compromised key of another trusted developer
- Redirect to attacker repository
- Malicious new developer
- Malicious existing developer
## Analysis of past attacks

<table>
<thead>
<tr>
<th>Attack Type</th>
<th>Count</th>
<th>GPG/SSL</th>
<th>Sigstore</th>
<th>TUF Online Targets</th>
<th>TUF Offline Targets</th>
<th>Artemis w/online targets</th>
<th>Artemis w/offline targets</th>
</tr>
</thead>
<tbody>
<tr>
<td>Repository compromise</td>
<td>13</td>
<td>✗</td>
<td>☐</td>
<td>○</td>
<td>●</td>
<td>■</td>
<td>■</td>
</tr>
<tr>
<td>Compromised key and repository</td>
<td>3</td>
<td>✗</td>
<td>☐</td>
<td>○</td>
<td>○</td>
<td>○</td>
<td>■</td>
</tr>
<tr>
<td>Compromised key</td>
<td>6</td>
<td>✗</td>
<td>☐</td>
<td>○</td>
<td>●</td>
<td>○</td>
<td>■</td>
</tr>
<tr>
<td>Compromised key for other trusted developer</td>
<td>2</td>
<td>✗</td>
<td>✗</td>
<td>●</td>
<td>●</td>
<td>●</td>
<td>●</td>
</tr>
<tr>
<td>Redirect to attacker repository</td>
<td>2</td>
<td>✗</td>
<td>●</td>
<td>●</td>
<td>●</td>
<td>●</td>
<td>●</td>
</tr>
<tr>
<td>Malicious new developer</td>
<td>1</td>
<td>✗</td>
<td>✗</td>
<td>●</td>
<td>●</td>
<td>●</td>
<td>●</td>
</tr>
<tr>
<td>Malicious existing developer</td>
<td>2</td>
<td>✗</td>
<td>✗</td>
<td>●</td>
<td>○</td>
<td>●</td>
<td>●</td>
</tr>
</tbody>
</table>
# Real-world Deployment

<table>
<thead>
<tr>
<th>Adoption requirement</th>
<th>Deployment</th>
<th>Artemis features</th>
<th>Configured by</th>
</tr>
</thead>
<tbody>
<tr>
<td>Define updates for each vehicle</td>
<td>Automotive</td>
<td></td>
<td>OEM</td>
</tr>
<tr>
<td>Protection from repository compromise</td>
<td>Automotive</td>
<td></td>
<td>OEM</td>
</tr>
<tr>
<td>Gather updates from multiple suppliers</td>
<td>Automotive</td>
<td></td>
<td>OEM</td>
</tr>
<tr>
<td>Using a third party container registry</td>
<td>Cloud</td>
<td>![icon]</td>
<td>Package manager</td>
</tr>
<tr>
<td>Store sensitive data on a private repository</td>
<td>Cloud</td>
<td>![icon] ![icon]</td>
<td>Company</td>
</tr>
<tr>
<td>Use software from a public repository</td>
<td>Cloud</td>
<td>![icon] ![icon] ![icon]</td>
<td>Package manager</td>
</tr>
<tr>
<td>Ensure updates are tested</td>
<td>Cloud</td>
<td>![icon]</td>
<td>Package manager</td>
</tr>
</tbody>
</table>

![icon] Repository thresholds
![icon] Per-package prioritization
![icon] Define a trusted subset
![icon] Role thresholds
![icon] Terminate search for a package
Conclusion

- Use of multiple software repositories has unique security challenges
- Articulated trust allows for selective trust in developers and repositories
- Implement articulated trust in Artemis
  - Multi-role delegations
  - Key pinning
  - Repository RBAC