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

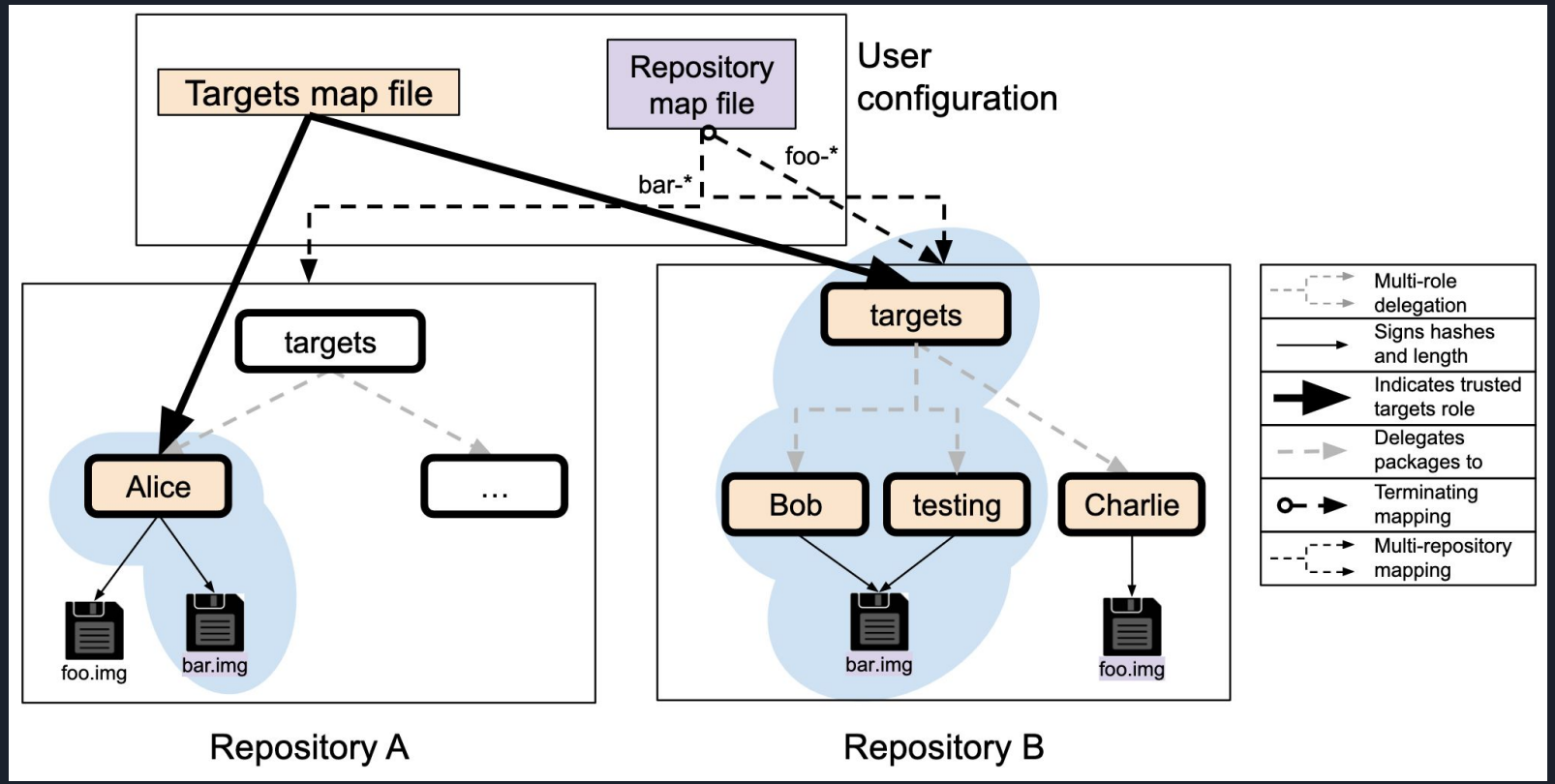
Attacker can:

- 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

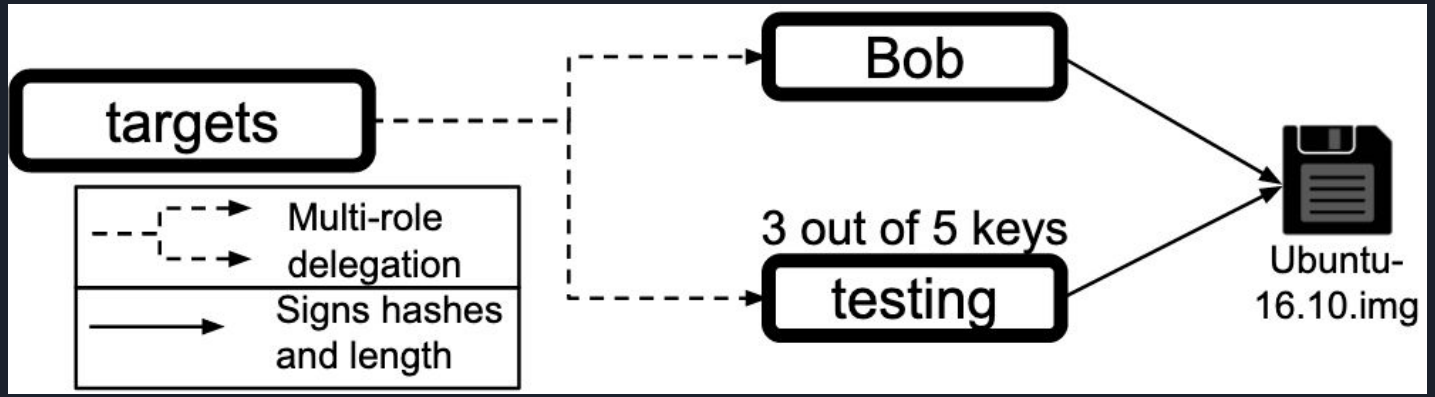
Goal:

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

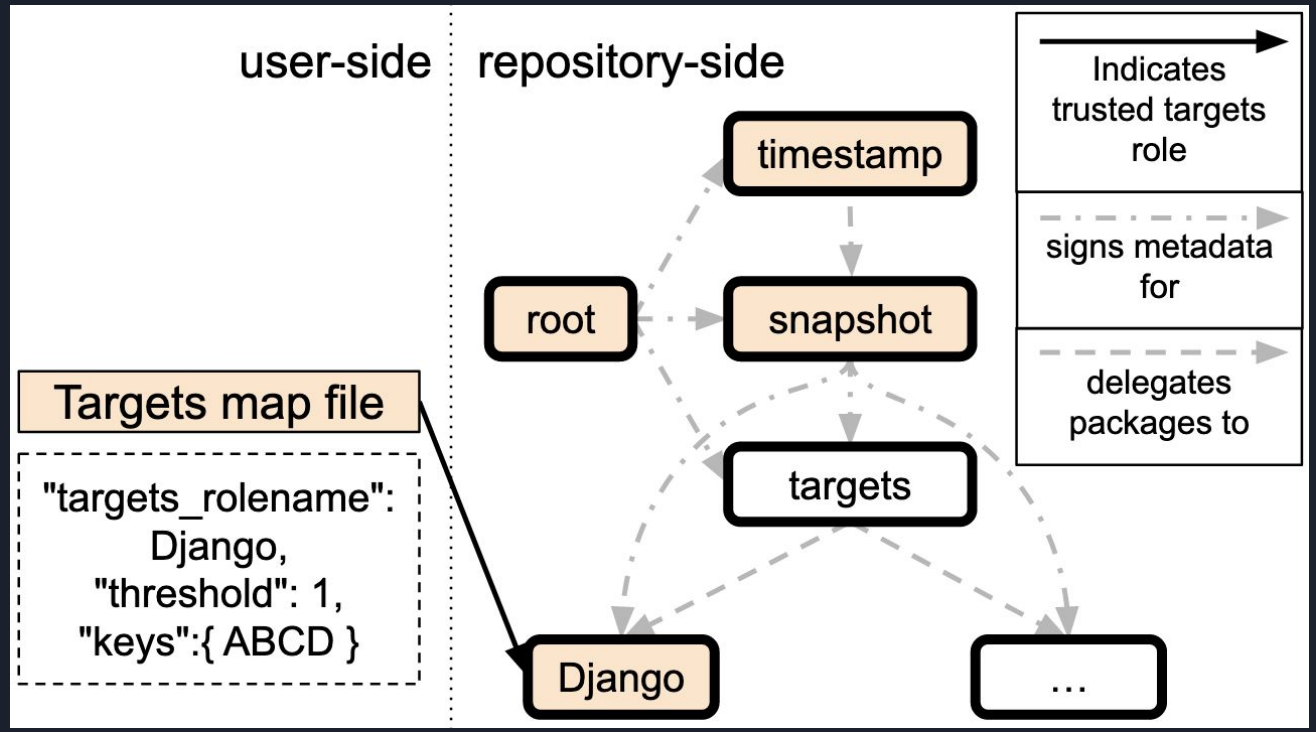
Artemis


Targets map file
Repository map file
User configuration
foo-*
bar-*
targets
Alice
...
foo.img
bar.img
Repository A
targets
Bob
testing
Charlie
bar.img
foo.img
Repository B
Multi-role delegation
Signs hashes and length
Indicates trusted targets role
Delegates packages to
Terminating mapping
Multi-repository mapping

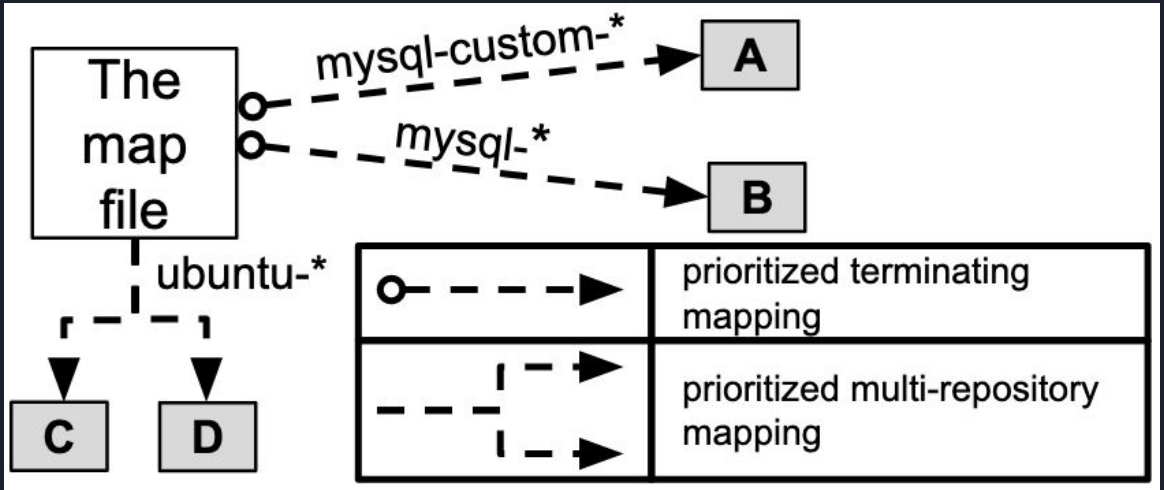
Multi-role Delegations



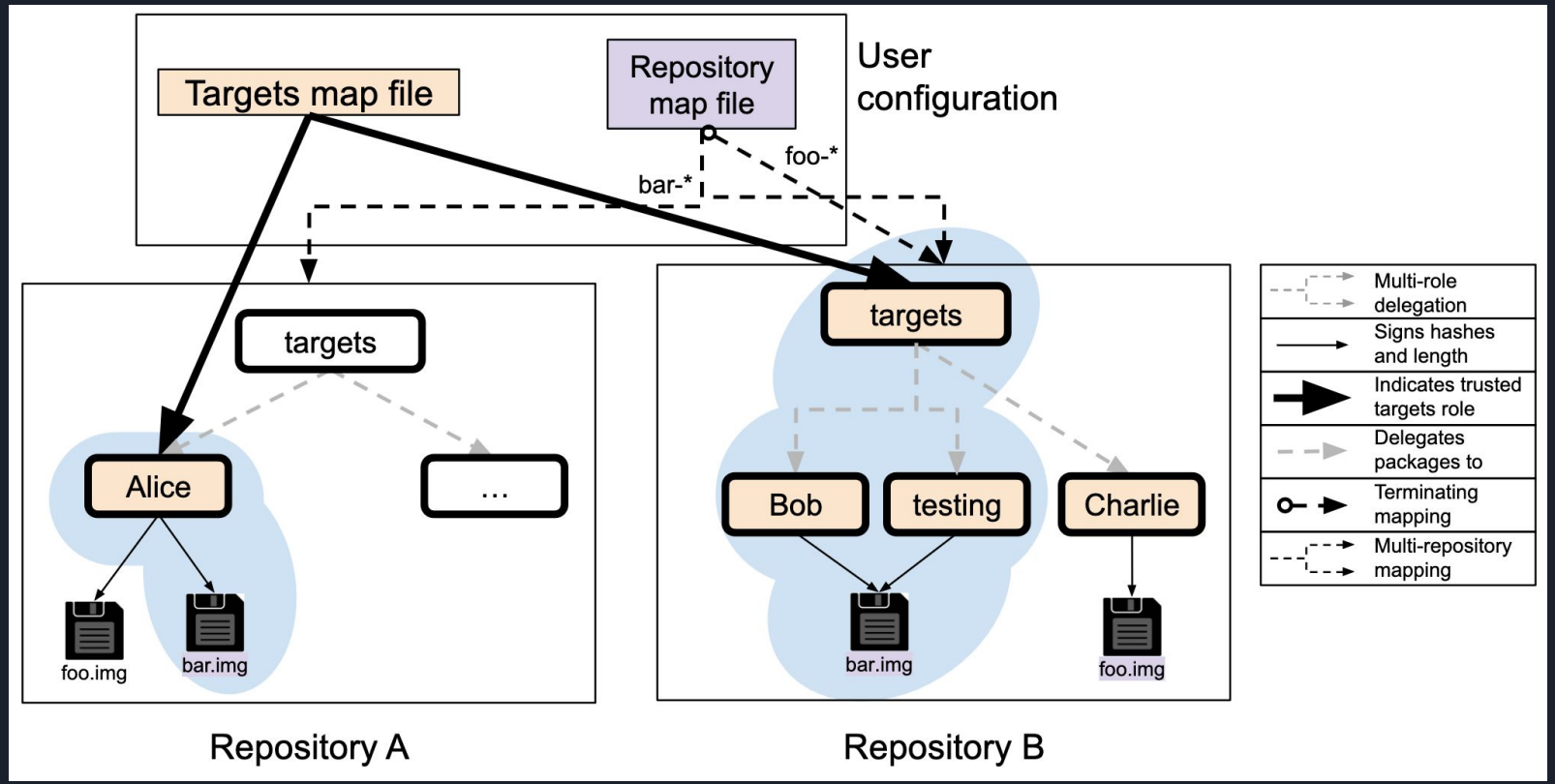
Key pinning


user-side
repository-side
Indicates trusted targets role
signs metadata for
delegates packages to
timestamp
root
snapshot
targets
Targets map file
"targets_rolename": Django,
"threshold": 1,
"keys":{ ABCD }
Django
...

Repository RBAC



Artemis


Targets map file
Repository map file
User configuration
foo-*
bar-*
targets
Alice
...
foo.img
bar.img
Repository A
targets
Bob
testing
Charlie
bar.img
foo.img
Repository B
Multi-role delegation
Signs hashes and length
Indicates trusted targets role
Delegates packages to
Terminating mapping
Multi-repository mapping

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

Attack Type	Count	GPG/ TLS	Sigstore	TUF		Artemis w/online targets			Artemis w/offline targets		
				Online targets	Offline targets	Key pinning	Multi-role delegations	Repository RBAC	Key pinning	Multi-role delegations	Repository RBAC
Repository compromise	13	✕	○	○	●	●	○	●	●	●	●
Compromised key and repository	3	✕	○	○	◐	◐	○	●	◐	●	●
Compromised key	6	✕	○	◐	◐	◐	●	◐	◐	●	◐
Compromised key for other trusted developer	2	✕	✕	●	●	●	●	●	●	●	●
Redirect to attacker repository	2	✕	●	●	●	●	●	●	●	●	●
Malicious new developer	1	✕	✕	◐	◐	●	●	◐	●	●	◐
Malicious existing developer	2	✕	✕	◐	◐	✕	●	✕	✕	●	✕

Real-world Deployment

Adoption requirement	Deployment	Artemis features	Configured by
Define updates for each vehicle	Automotive		OEM
Protection from repository compromise	Automotive		OEM
Gather updates from multiple suppliers	Automotive		OEM
Using a third party container registry	Cloud		Package manager
Store sensitive data on a private repository	Cloud		Company
Use software from a public repository	Cloud		Package manager
Ensure updates are tested	Cloud		Package manager

Repository thresholds · Per-package prioritization · Define a trusted subset · Role thresholds · 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