# AURsec
# Detecting and preventing targeted attacks in the Arch User Repository: A blockchain-based approach

Lukas Krismer & Bennett Piater

October 10, 2017

Universität Innsbruck - QE - Christian Sillaber

2017-10-10

AURsec
Detecting and preventing targeted attacks in the Arch User Repository: A blockchain-based approach

AURsec
Detecting and preventing targeted attacks in the Arch User Repository: A blockchain-based approach

Lukas Krismer & Bennett Piater
October 10, 2017
Universität Innsbruck - QE - Christian Sillaber

# Outline

Background: AUR

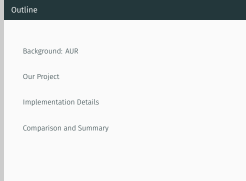
Our Project

Implementation Details

Comparison and Summary

2017-10-10

AURsec
Detecting and preventing targeted attacks in the Arch User Repository: A blockchain-based approach



Outline

Background: AUR

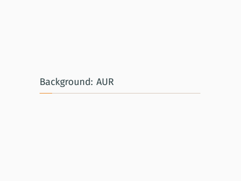
Our Project

Implementation Details

Comparison and Summary

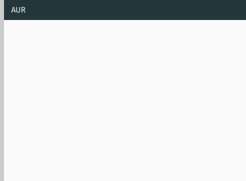
1 min L | Betreuer: Christian Sillaber - Quality Engineering

# Background: AUR



Background: AUR

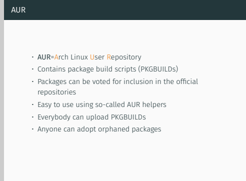
# AUR



3 min L | Arch has a active community -> packages to ftp://ftp.archlinux.org/income (long delay) -> Trusted User Repo -> AUR Comparable to Pypi npm | fulfill conditions

# AUR

- AUR=Arch Linux User Repository
- Contains package build scripts (PKGBUILDs)
- Packages can be voted for inclusion in the official repositories
- Easy to use using so-called AUR helpers
- Everybody can upload PKGBUILDs
- Anyone can adopt orphaned packages

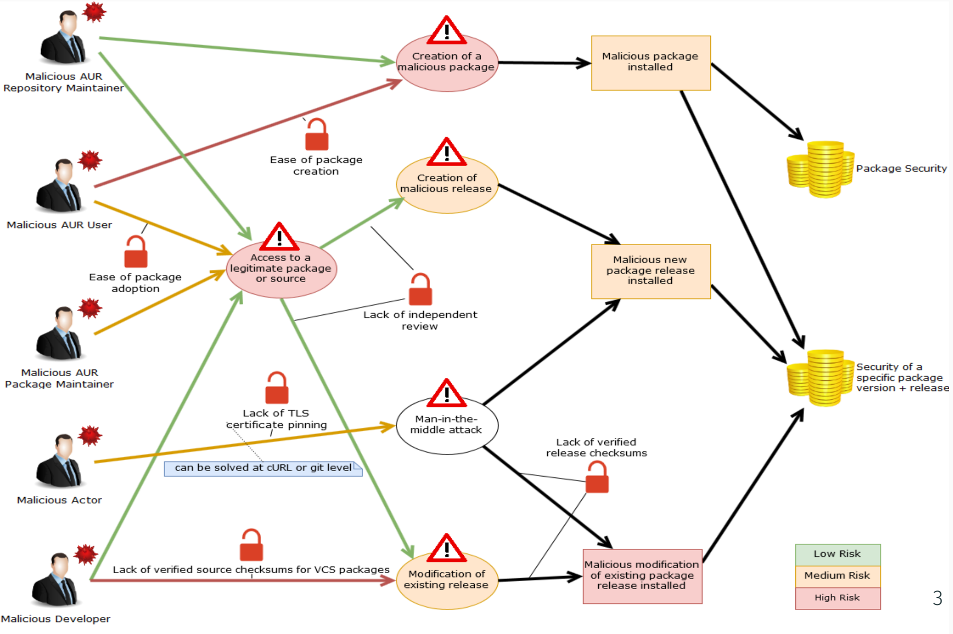


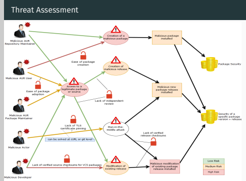
AUR

- AUR=Arch Linux User Repository
- Contains package build scripts (PKGBUILDs)
- Packages can be voted for inclusion in the official repositories
- Easy to use using so-called AUR helpers
- Everybody can upload PKGBUILDs
- Anyone can adopt orphaned packages

3 min L | Arch has a active community -> packages to ftp://ftp.archlinux.org/income (long delay) -> Trusted User Repo -> AUR Comparable to Pypi npm | fulfill conditions

# Threat Assessment


Malicious AUR Repository Maintainer
Malicious AUR User
Malicious AUR Package Maintainer
Malicious Actor
Malicious Developer
Creation of a malicious package
Ease of package creation
Creation of malicious release
Access to a legitimate package or source
Ease of package adoption
Lack of independent review
Lack of TLS certificate pinning
can be solved at cURL or git level
Man-in-the-middle attack
Lack of verified source checksums for VCS packages
Modification of existing release
Lack of verified release checksums
Malicious package installed
Malicious new package release installed
Malicious modification of existing package release installed
Package Security
Security of a specific package version + release
Low Risk
Medium Risk
High Risk

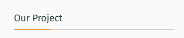


3.5 min B |

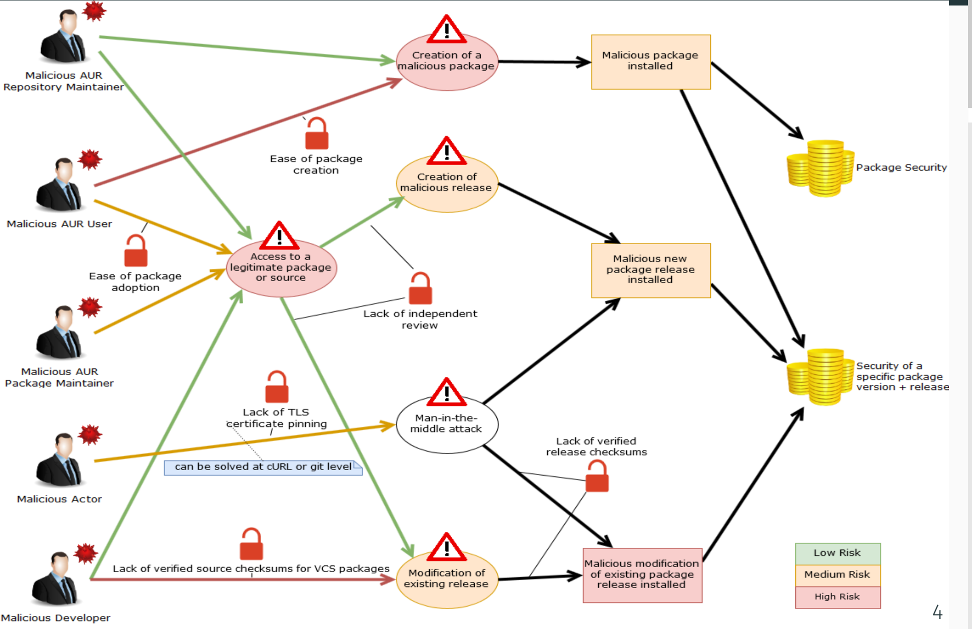
*Explain the main security issues!*

- The underlying problems of the AUR are not really solvable
- Too many people have access to build scripts and sources
- →: (automated) server-side signatures would only prevent MITM
- malicious packages, releases and modifications of releases are very easy to do

# Our Project


Our Project

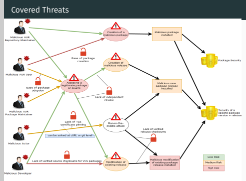
# Covered Threats



Malicious AUR Repository Maintainer
Malicious AUR User
Malicious AUR Package Maintainer
Malicious Actor
Malicious Developer
Ease of package creation
Ease of package adoption
Lack of independent review
Lack of TLS certificate pinning
can be solved at cURL or git level
Lack of verified source checksums for VCS packages
Lack of verified release checksums
Creation of a malicious package
Creation of malicious release
Access to a legitimate package or source
Man-in-the-middle attack
Modification of existing release
Malicious package installed
Malicious new package release installed
Malicious modification of existing package release installed
Package Security
Security of a specific package version + release
Low Risk
Medium Risk
High Risk

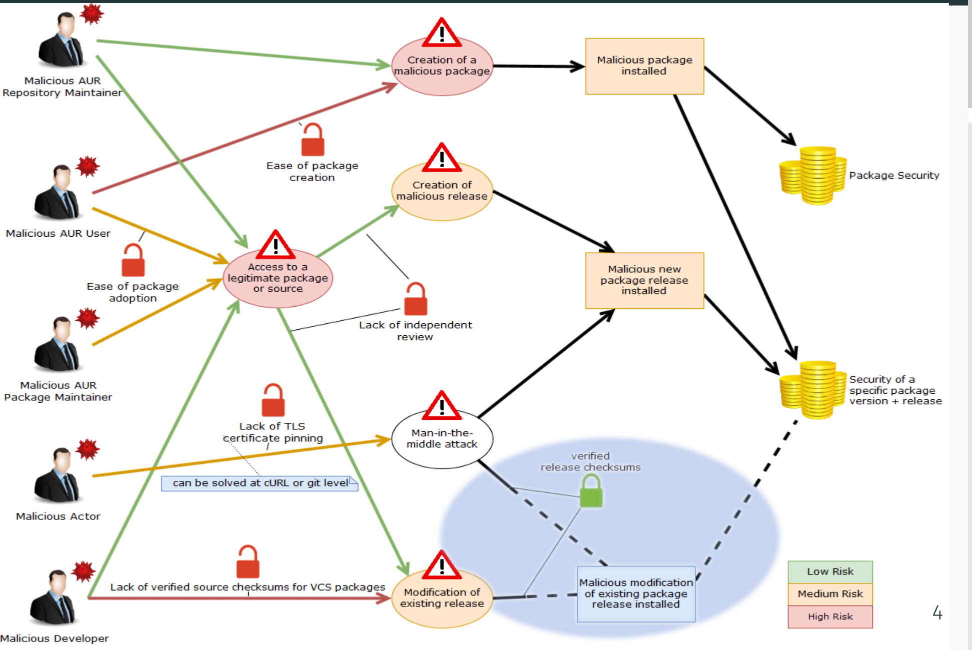
2017-10-10

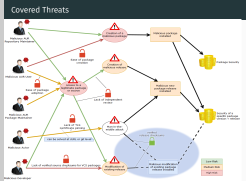
AURsec
Detecting and preventing targeted attacks in the Arch User Repository: A blockchain-based approach
└Our Project



2 min L |

# Covered Threats


Malicious AUR Repository Maintainer
Malicious AUR User
Malicious AUR Package Maintainer
Malicious Actor
Malicious Developer
Creation of a malicious package
Ease of package creation
Creation of malicious release
Access to a legitimate package or source
Ease of package adoption
Lack of independent review
Lack of TLS certificate pinning
can be solved at cURL or git level
Man-in-the-middle attack
Lack of verified source checksums for VCS packages
Modification of existing release
verified release checksums
Malicious modification of existing package release installed
Malicious package installed
Malicious new package release installed
Package Security
Security of a specific package version + release
Low Risk
Medium Risk
High Risk



2 min L |

# Live Demo

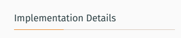
4 min BL | Wirklich live

```
git clone aur:aursec
aursec-hash -d aursec
aursec-hash aursec | aursec-verify-hashes
aursec -v aursec
echo var=val » aursec/PKGBUILD
aursec aursec
```

??

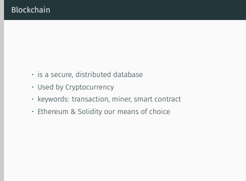
```
git clone aur:aursec-git
aursec -d aursec-git
```

# Implementation Details


Implementation Details

# Blockchain

- is a secure, distributed database
- Used by Cryptocurrency
- keywords: transaction, miner, smart contract
- Ethereum & Solidity our means of choice

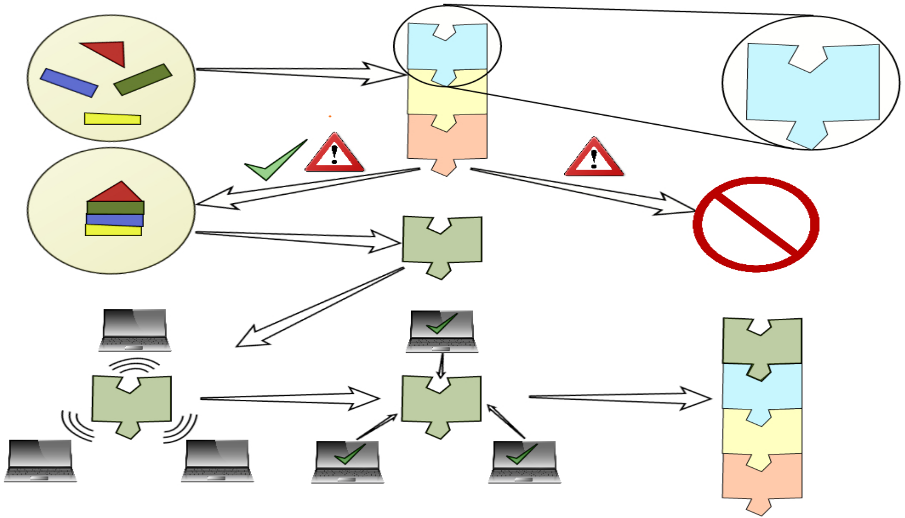


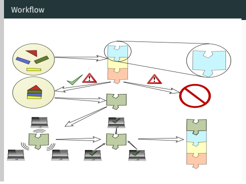
Blockchain

- is a secure, distributed database
- Used by Cryptocurrency
- keywords: transaction, miner, smart contract
- Ethereum & Solidity our means of choice

4 min L | Cryptocurrency

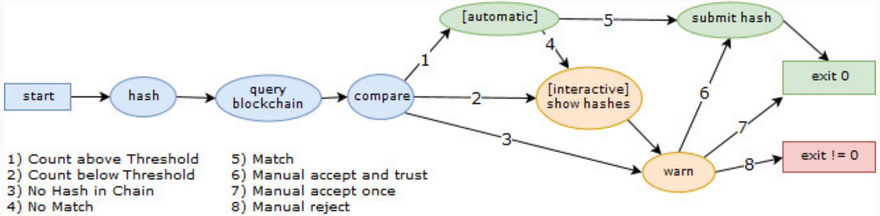
# Workflow





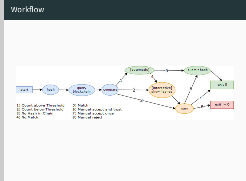
3 min B |

# Workflow



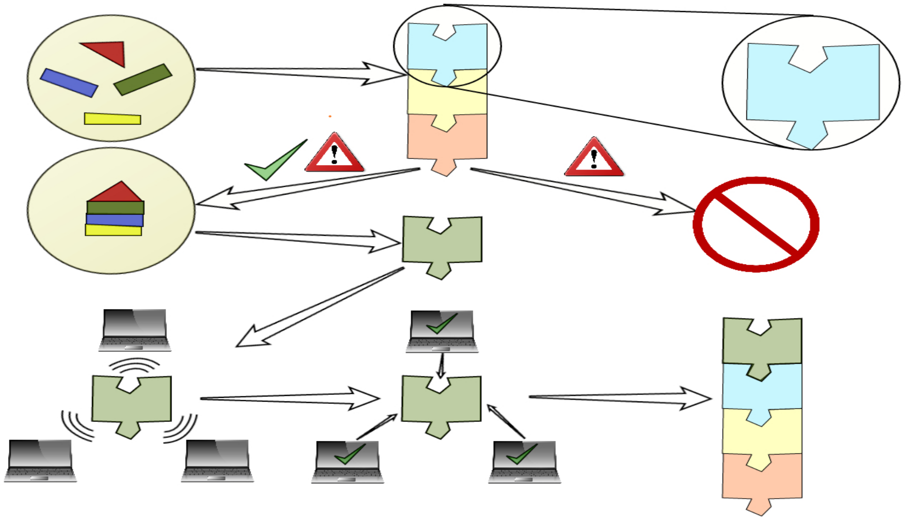
1) Count above Threshold
2) Count below Threshold
3) No Hash in Chain
4) No Match

5) Match
6) Manual accept and trust
7) Manual accept once
8) Manual reject



3 min B |

# Workflow

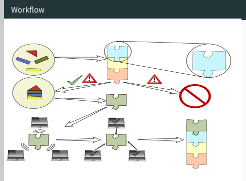


2017-10-10

AURsec

Detecting and preventing targeted attacks in the Arch User Repository: A blockchain-based approach

└Implementation Details



Workflow

3 min B |

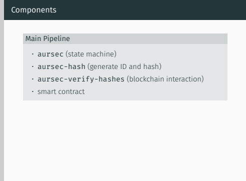
# Components

## Main Pipeline

- `aursec` (state machine)
- `aursec-hash` (generate ID and hash)
- `aursec-verify-hashes` (blockchain interaction)
- smart contract

2017-10-10

AURsec
Detecting and preventing targeted attacks in the Arch User Repository: A blockchain-based approach
└─Implementation Details



Components

Main Pipeline

- `aursec` (state machine)
- `aursec-hash` (generate ID and hash)
- `aursec-verify-hashes` (blockchain interaction)
- smart contract

4 min BL |

UNIX philosophy - small tools doing one thing well. Work on stdin/stdout with blocking I/O.

Good parallelism, straightforward to maintain and extend

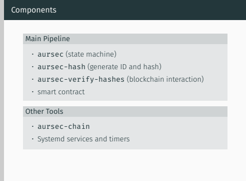
# Components

## Main Pipeline

- `aursec` (state machine)
- `aursec-hash` (generate ID and hash)
- `aursec-verify-hashes` (blockchain interaction)
- smart contract

## Other Tools

- `aursec-chain`
- Systemd services and timers



Components

Main Pipeline

- aursec (state machine)
- aursec-hash (generate ID and hash)
- aursec-verify-hashes (blockchain interaction)
- smart contract

Other Tools

- aursec-chain
- Systemd services and timers

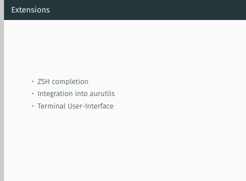
4 min BL |

UNIX philosophy - small tools doing one thing well. Work on stdin/stdout with blocking I/O.

Good parallelism, straightforward to maintain and extend

# Extensions

- ZSH completion
- Integration into aurutils
- Terminal User-Interface



Extensions

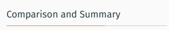
- ZSH completion
- Integration into aurutils
- Terminal User-Interface

2 min L | Live Demo

aursec-tui

aurutils integration

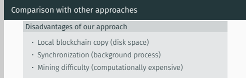
# Comparison and Summary


Comparison and Summary

# Comparison with other approaches

## Disadvantages of our approach

- Local blockchain copy (disk space)
- Synchronization (background process)
- Mining difficulty (computationally expensive)


Comparison with other approaches
Disadvantages of our approach
- Local blockchain copy (disk space)
- Synchronization (background process)
- Mining difficulty (computationally expensive)

2 min B |

Custom repositories make sense for private organizations – not for large scale

Redesigning the AUR is not an option – no one wants to sacrifice the ease of use

→ AURsec seems like overkill, but it's still the best solution available.

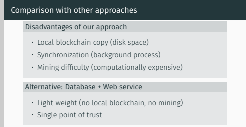
# Comparison with other approaches

## Disadvantages of our approach

- Local blockchain copy (disk space)
- Synchronization (background process)
- Mining difficulty (computationally expensive)

## Alternative: Database + Web service

- Light-weight (no local blockchain, no mining)
- Single point of trust


Comparison with other approaches
Disadvantages of our approach
- Local blockchain copy (disk space)
- Synchronization (background process)
- Mining difficulty (computationally expensive)
Alternative: Database + Web service
- Light-weight (no local blockchain, no mining)
- Single point of trust

2 min B |

Custom repositories make sense for private organizations – not for large scale

Redesigning the AUR is not an option – no one wants to sacrifice the ease of use

→ AURsec seems like overkill, but it’s still the best solution available.

# Comparison with other approaches

## Disadvantages of our approach

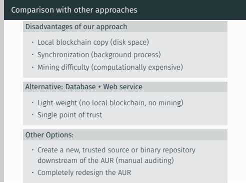
- Local blockchain copy (disk space)
- Synchronization (background process)
- Mining difficulty (computationally expensive)

## Alternative: Database + Web service

- Light-weight (no local blockchain, no mining)
- Single point of trust

## Other Options:

- Create a new, trusted source or binary repository downstream of the AUR (manual auditing)
- Completely redesign the AUR



Comparison with other approaches

Disadvantages of our approach

- Local blockchain copy (disk space)
- Synchronization (background process)
- Mining difficulty (computationally expensive)

Alternative: Database + Web service

- Light-weight (no local blockchain, no mining)
- Single point of trust

Other Options:

- Create a new, trusted source or binary repository downstream of the AUR (manual auditing)
- Completely redesign the AUR

2 min B |

Custom repositories make sense for private organizations – not for large scale

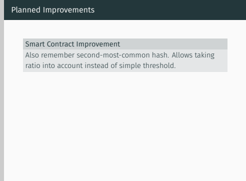
Redesigning the AUR is not an option – no one wants to sacrifice the ease of use

→ AURsec seems like overkill, but it's still the best solution available.

# Planned Improvements

**Smart Contract Improvement**

Also remember second-most-common hash. Allows taking ratio into account instead of simple threshold.



Planned Improvements

Smart Contract Improvement

Also remember second-most-common hash. Allows taking ratio into account instead of simple threshold.

2 min B |

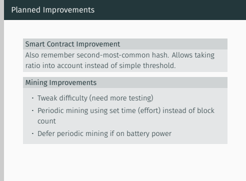
# Planned Improvements

## Smart Contract Improvement

Also remember second-most-common hash. Allows taking ratio into account instead of simple threshold.

## Mining Improvements

- Tweak difficulty (need more testing)
- Periodic mining using set time (effort) instead of block count
- Defer periodic mining if on battery power



Planned Improvements

Smart Contract Improvement

Also remember second-most-common hash. Allows taking ratio into account instead of simple threshold.

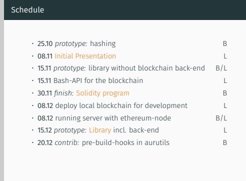
Mining Improvements

- Tweak difficulty (need more testing)
- Periodic mining using set time (effort) instead of block count
- Defer periodic mining if on battery power

2 min B |

# Schedule

| | |
|---|---|
| 25.10 *prototype:* hashing | B |
| 08.11 Initial Presentation | L |
| 15.11 *prototype:* library without blockchain back-end | B/L |
| 15.11 Bash-API for the blockchain | L |
| 30.11 *finish:* Solidity program | B |
| 08.12 deploy local blockchain for development | L |
| 08.12 running server with ethereum-node | B/L |
| 15.12 *prototype:* Library incl. back-end | L |
| 20.12 *contrib:* pre-build-hooks in aurutils | B |



Schedule

- 25.10 *prototype:* hashing B
- 08.11 Initial Presentation L
- 15.11 *prototype:* library without blockchain back-end B/L
- 15.11 Bash-API for the blockchain L
- 30.11 *finish:* Solidity program B
- 08.12 deploy local blockchain for development L
- 08.12 running server with ethereum-node B/L
- 15.12 *prototype:* Library incl. back-end L
- 20.12 *contrib:* pre-build-hooks in aurutils B

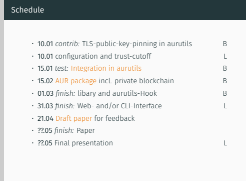
2 min B

Wir haben eine sehr **detaillierte Planung** ausgearbeitet. Einerseits benötigen wir sie, um effizient **kooperieren** zu können und zügig voran zu kommen; Andererseits soll sie uns auch ein Maximaltempo vergeben, denn wir tendieren beide eher dazu, uns zu **überarbeiten**.

- Solidity-program auf Blockchain
- Library-Prototyp
- Beiträge zum AUR-Helper aurutils über Weihnachten

## Schedule

- 10.01 *contrib:* TLS-public-key-pinning in aurutils B
- 10.01 configuration and trust-cutoff L
- 15.01 *test:* Integration in aurutils B
- 15.02 AUR package incl. private blockchain B
- 01.03 *finish:* libary and aurutils-Hook B
- 31.03 *finish:* Web- and/or CLI-Interface L
- 21.04 Draft paper for feedback
- ??.05 *finish:* Paper
- ??.05 Final presentation L



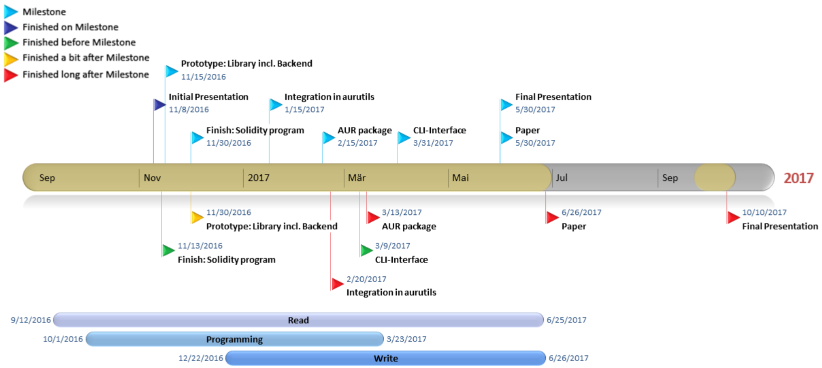
Schedule

- 10.01 *contrib:* TLS-public-key-pinning in aurutils B
- 10.01 configuration and trust-cutoff L
- 15.01 *test:* Integration in aurutils B
- 15.02 AUR package incl. private blockchain B
- 01.03 *finish:* libary and aurutils-Hook B
- 31.03 *finish:* Web- and/or CLI-Interface L
- 21.04 Draft paper for feedback
- ??.05 *finish:* Paper
- ??.05 Final presentation L

2 min B

- am 15.01 mit aurutils testbar
- AUR-Paket zur einfachen Verbreitung
- Programmierung endet am 31. März
- Meiste Schreibarbeit im April und besonders über Ostern
- Abgabe bequem for den Klausuren

# Schedule

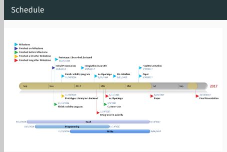


2017-10-10

AURsec

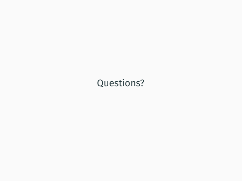
Detecting and preventing targeted attacks in the Arch User Repository: A blockchain-based approach

└─Comparison and Summary



2 min L |

# Questions?



Questions?