

Cargo Sherlock: An SMT-Based Checker for Software Trust Costs

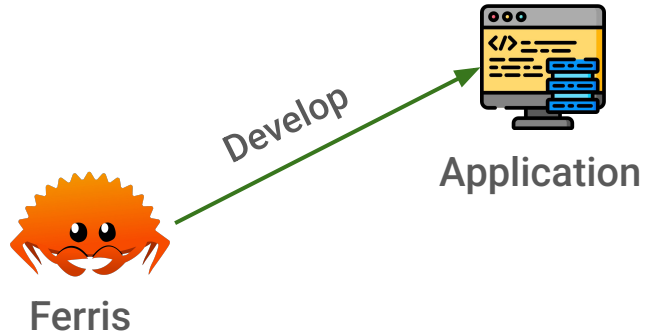
Muhammad Hassnain (mhassnain@ucdavis.edu)

Anirudh Basu (anibasus@ucdavis.edu)

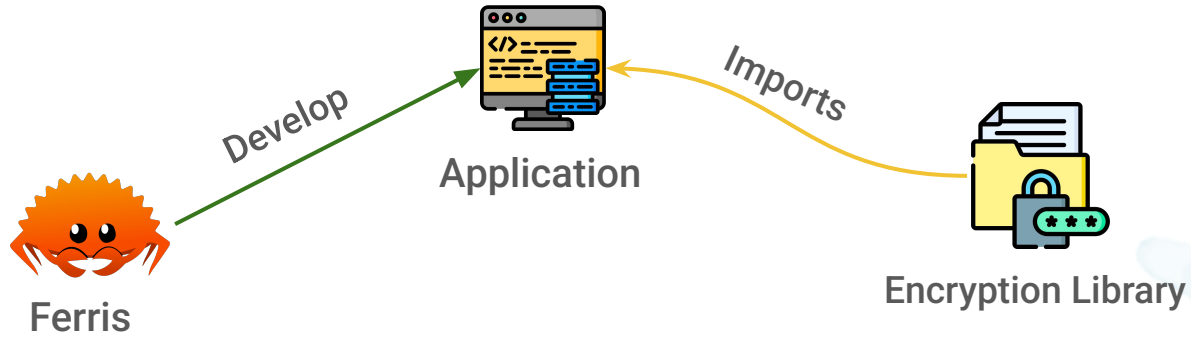
Ethan Ng (eyng@ucdavis.edu)

Caleb Stanford (cdstanford@ucdavis.edu)

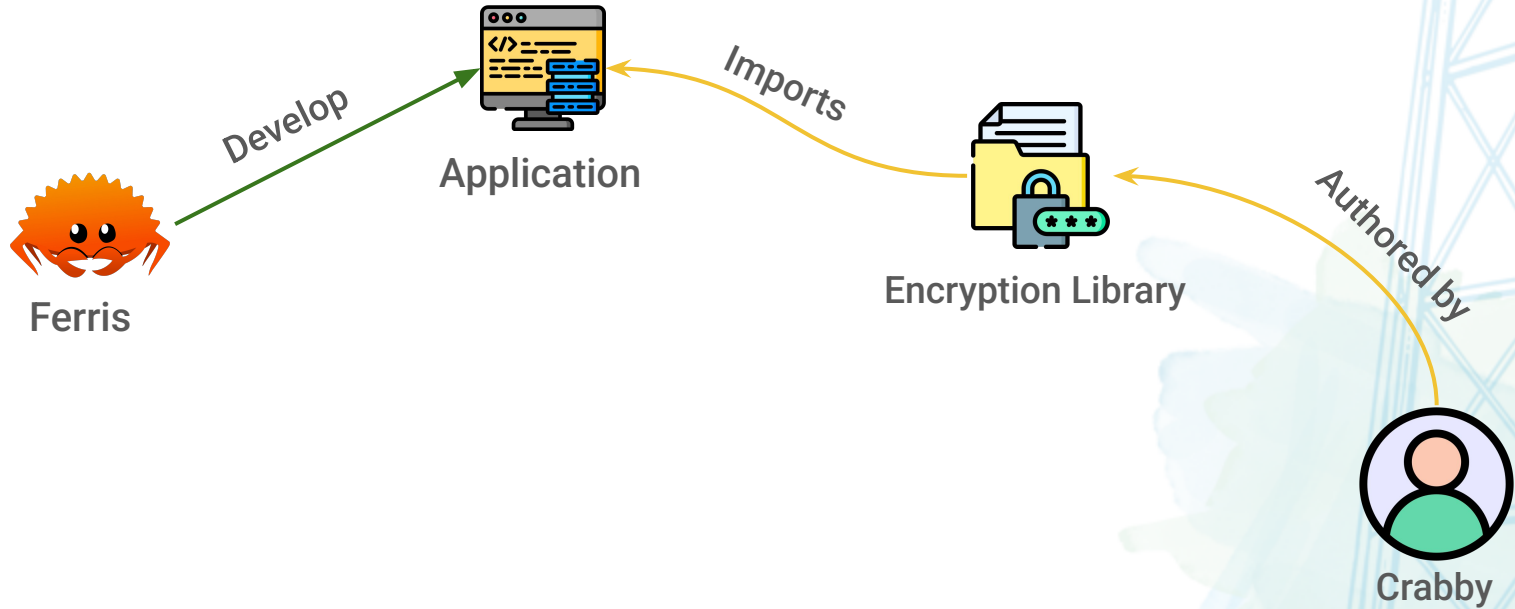
Third Party Dependencies



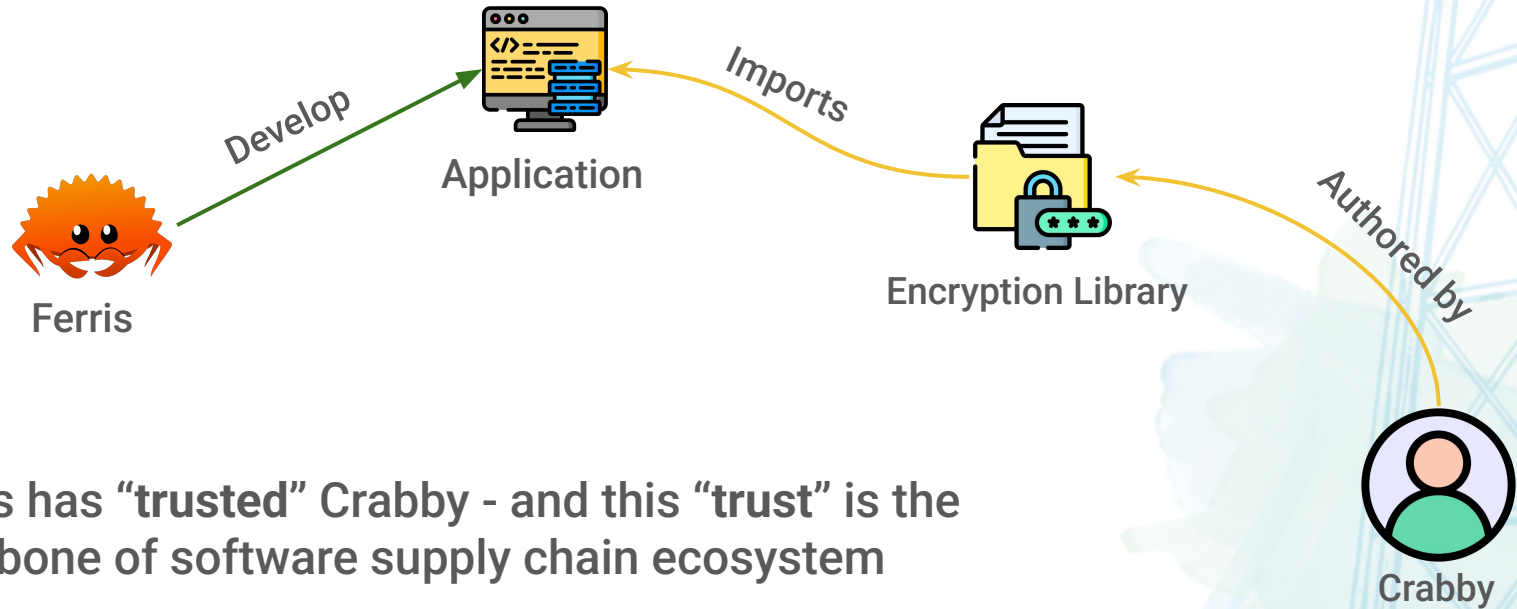
Third Party Dependencies



Third Party Dependencies

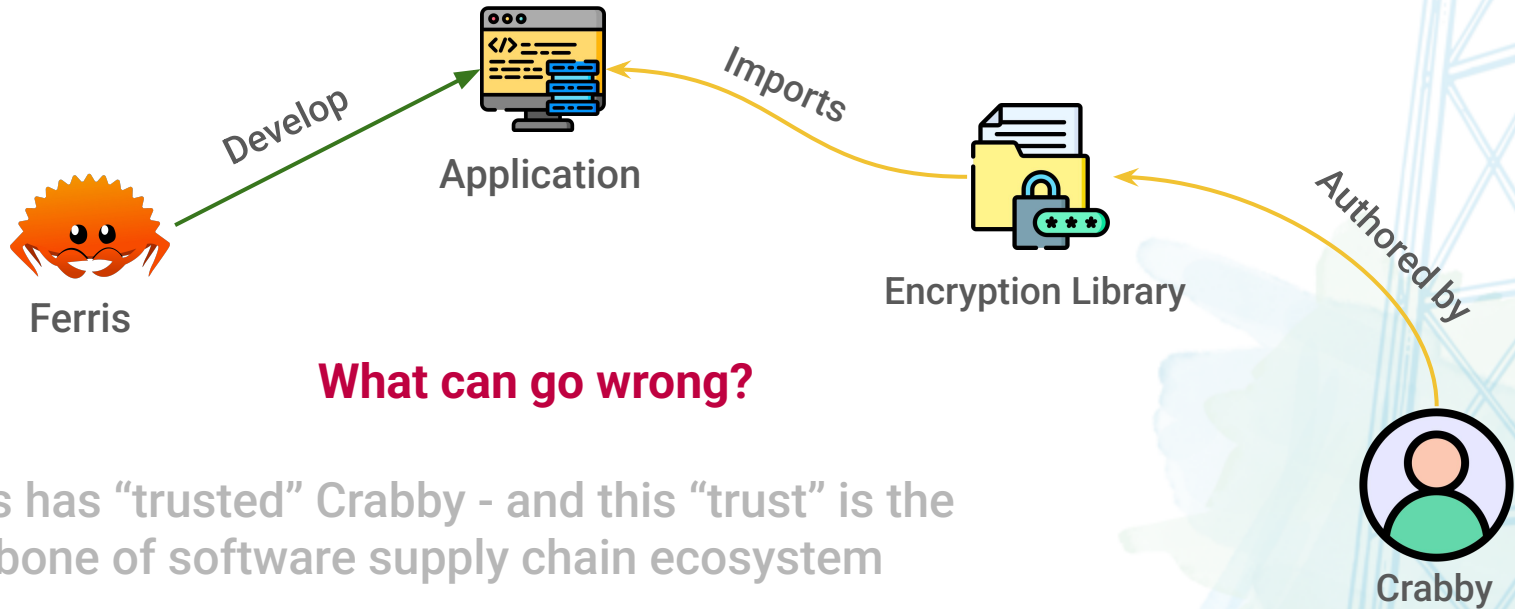


Third Party Dependencies

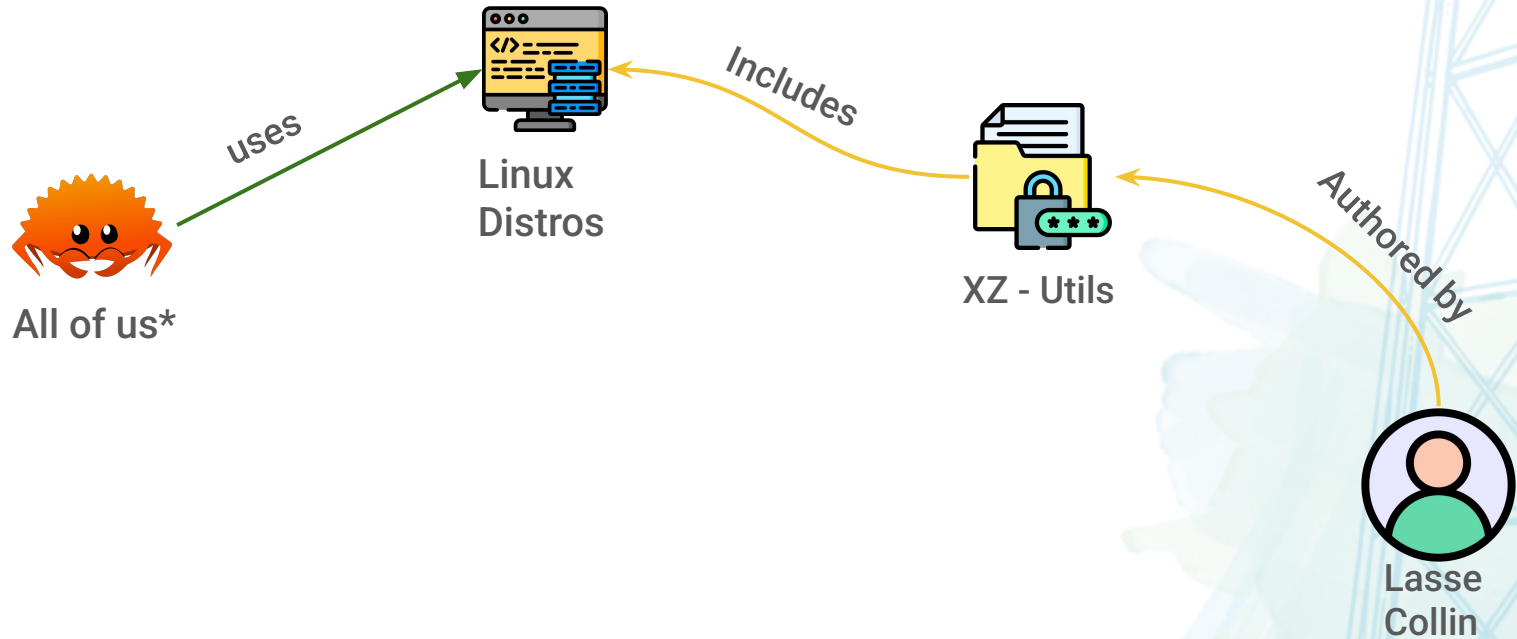


Ferris has “trusted” Crabby - and this “trust” is the backbone of software supply chain ecosystem

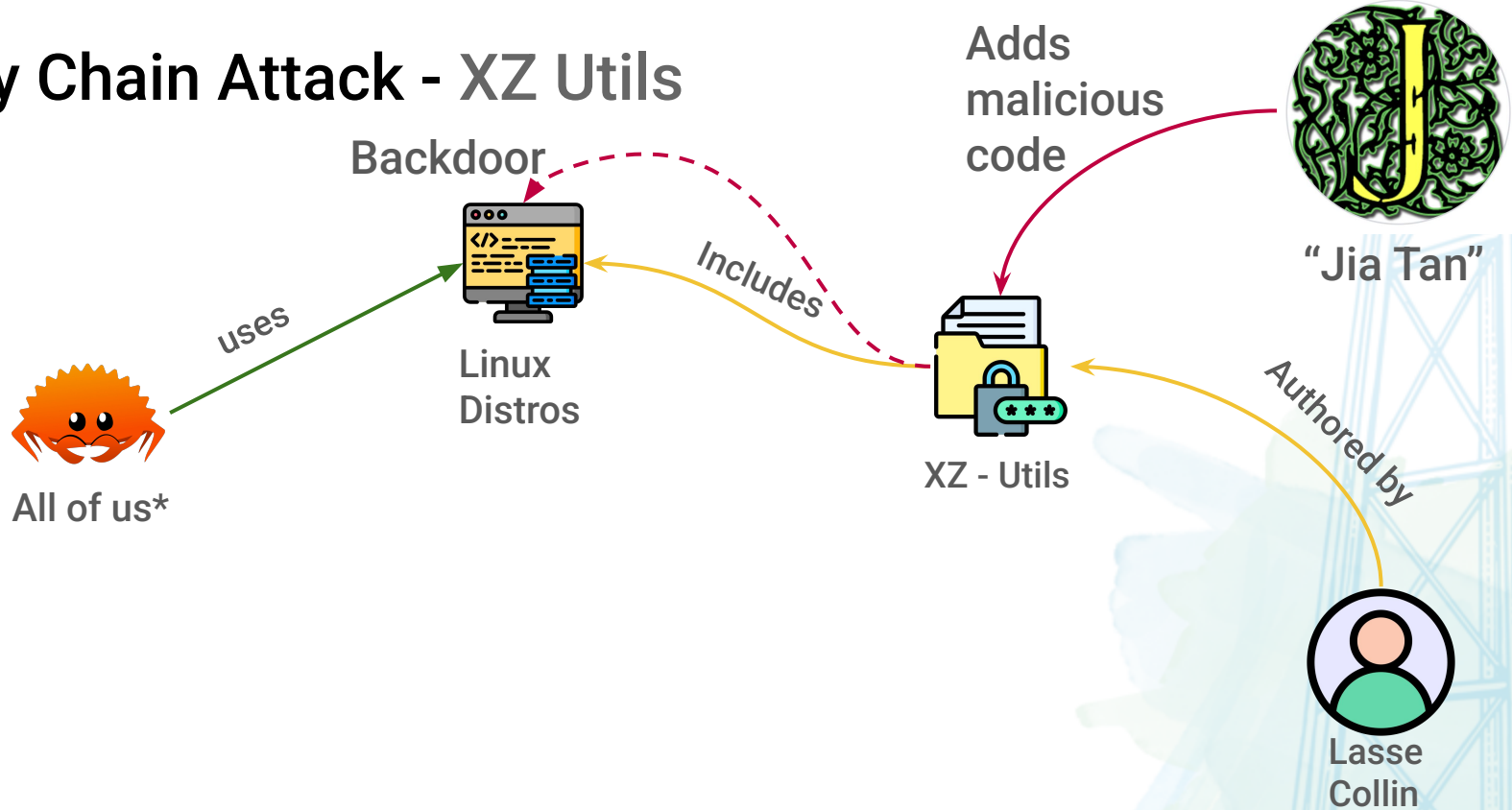
Third Party Dependencies



Supply Chain Attack - XZ Utils



Supply Chain Attack - XZ Utils



Supply Chain Attacks - Rust

Security advisory: malicious crate rustdecimal

May 10, 2022 · The Rust Security Response WG

Supply Chain Attacks - Rust

Security advisory: malicious crates.io: Malicious crates faster_log and async_println

Ma

Sept. 24, 2025 · Walter Pearce on behalf of [the crates.io team](#)

On September 24th, the crates.io team was notified by Kirill Boychenko from the Socket Threat Research Team of two malicious crates which were actively searching file contents for Ethereum private keys, Solana private keys, and arbitrary byte arrays for exfiltration.

Formally Checked Tools

Some developer's use:

- Static Analysis Tools
- Program Verification
- Some other methodology.....



Formally Checked Tools

Some developer's use:

- Static Analysis Tools
- Program Verification
- Some other methodology.....

Does not take into account human factors!

Human Factors

Human Factors include things like:

- Audits
- Download Counts
- Developer Reputation



Human Factors

Human Factors include things like:

- Audits
- Download Counts
- Developer Reputation

51% of developers use
some form of heuristics
during dependency
review



Hamer et al. 2025 [[21](#)]

Human Factors

Human Factors include things like:

- Audits
- Download Counts
- Developer Reputation

No formal guarantees!

51% of developers use
some form of heuristics
during dependency
review



Hamer et al. 2025 [[21](#)]

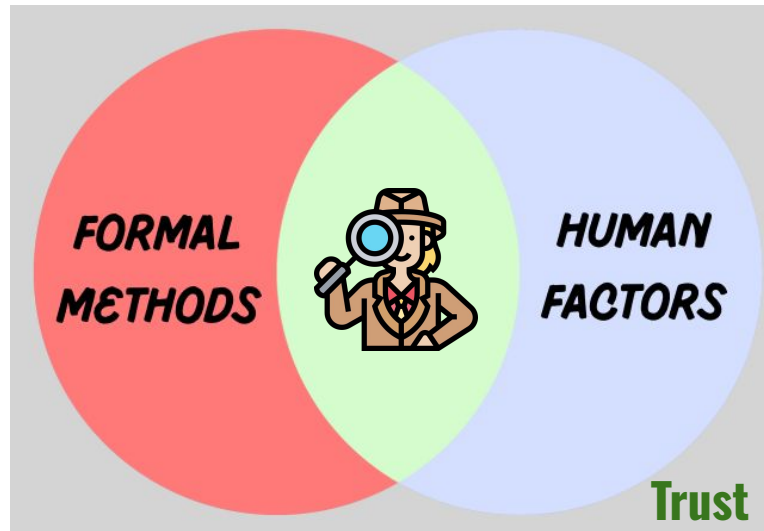
Research Question

Can trust be formally checked while still capturing human factors?



Research Question

Can trust be formally checked while still capturing human factors?



Cargo Sherlock - Contributions



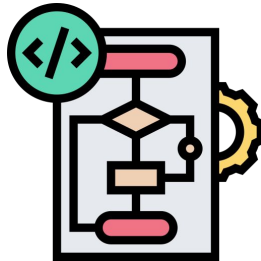
Supply Chain Trust
→ Minimum Trust
Problem

Cargo Sherlock - Contributions



Supply Chain Trust
→ Minimum Trust
Problem

Two Algorithms to
solve the Minimum
Trust Problem

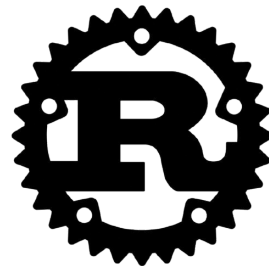
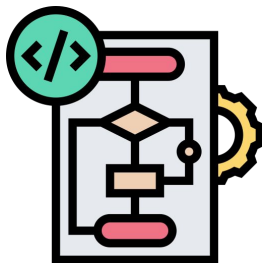


Cargo Sherlock - Contributions



Supply Chain Trust
→ Minimum Trust
Problem

Two Algorithms to
solve the Minimum
Trust Problem



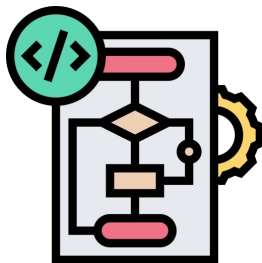
Implementation for
Rust Ecosystem

Cargo Sherlock - Contributions

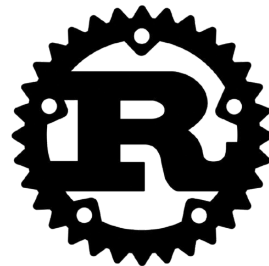


Supply Chain Trust
→ Minimum Trust
Problem

Two Algorithms to
solve the Minimum
Trust Problem



Implementation for
Rust Ecosystem



Evaluation on real
and synthetic supply
chain attacks





Cargo Sherlock - Tool Output

Analysis Report for anyhow-1.0.97

Cost Range: 0 (Min) – 100 (Max)
Trust Cost (lower is better):  5
Severity Label: **SAFE**

Assumptions for Trusting

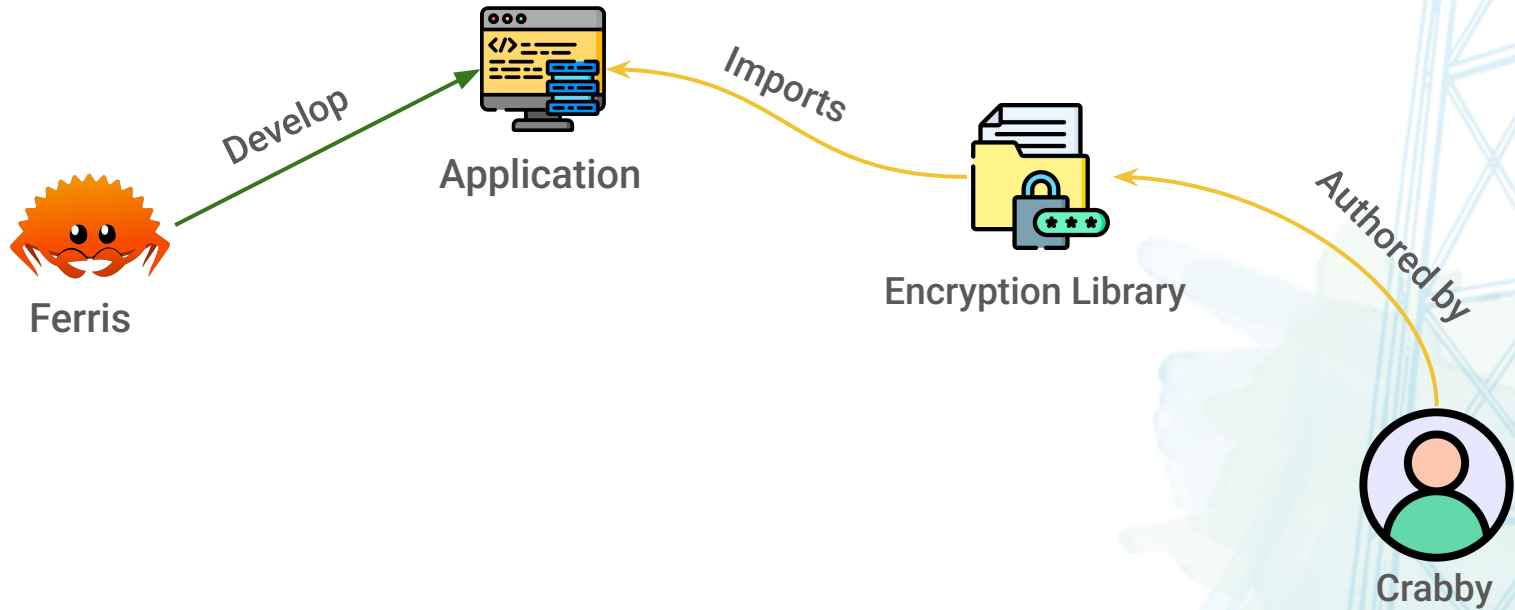
- anyhow-1.0.97 having a current audit implies it is safe: 5 cost

Cargo Sherlock - Outline

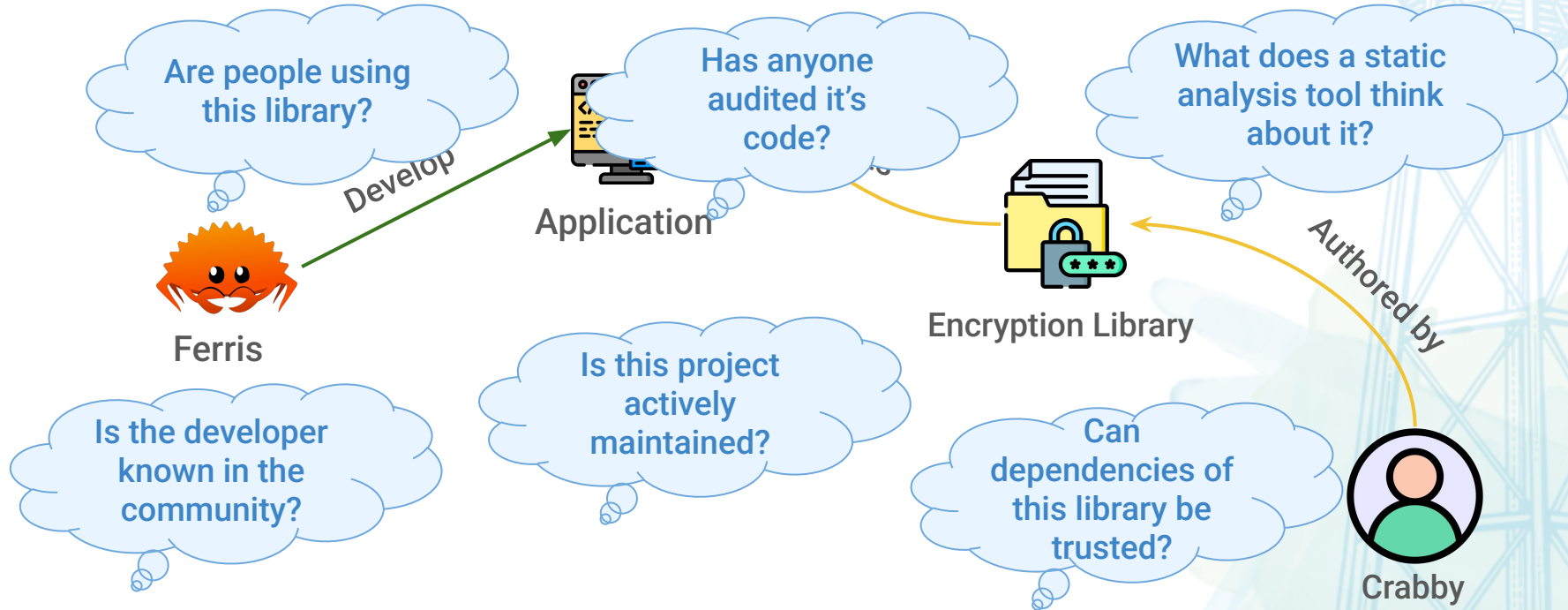
1. Minimum Trust Problem
2. Goals
3. Design and Implementation
4. Evaluation



Cargo Sherlock - Minimum Trust Problem



Cargo Sherlock - Minimum Trust Problem



Cargo Sherlock - Minimum Trust Pr

Ferris has a set of assumptions to pick from!



Cargo Sherlock - Minimum Trust Pr

Ferris has a set of assumptions to pick from!

How is Ferris going to pick a set of assumptions?



Cargo Sherlock - Minimum Trust Problem

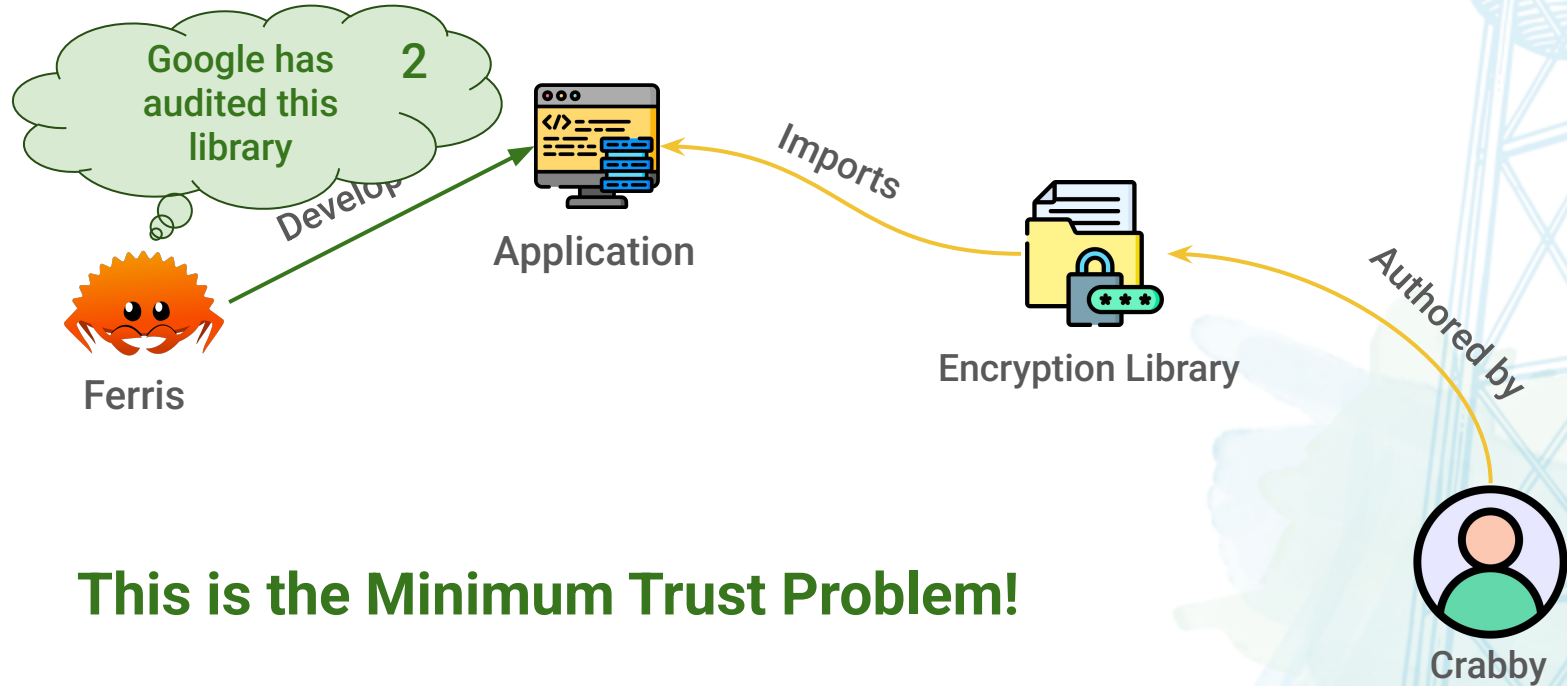
Ferris has a set of assumptions to pick from!

How is Ferris going to pick a set of assumptions?

Each assumption has an associated cost

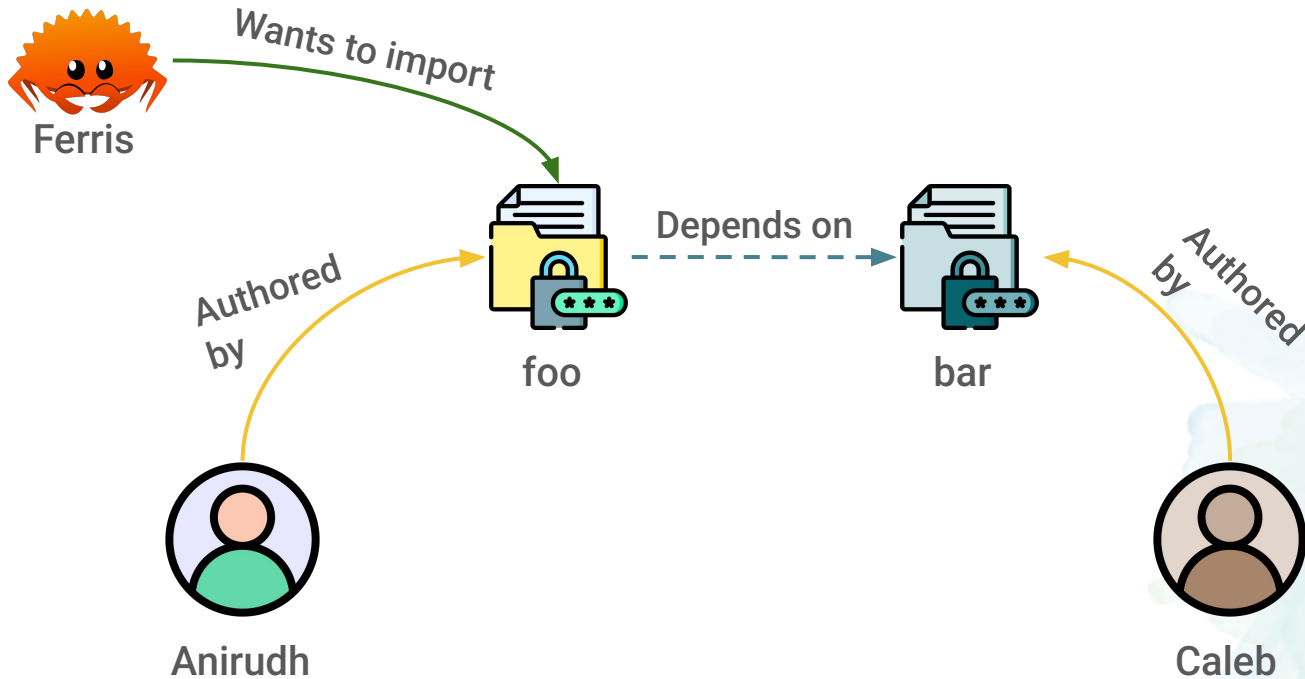


Cargo Sherlock - Minimum Trust Problem



This is the Minimum Trust Problem!

Minimum Trust Problem

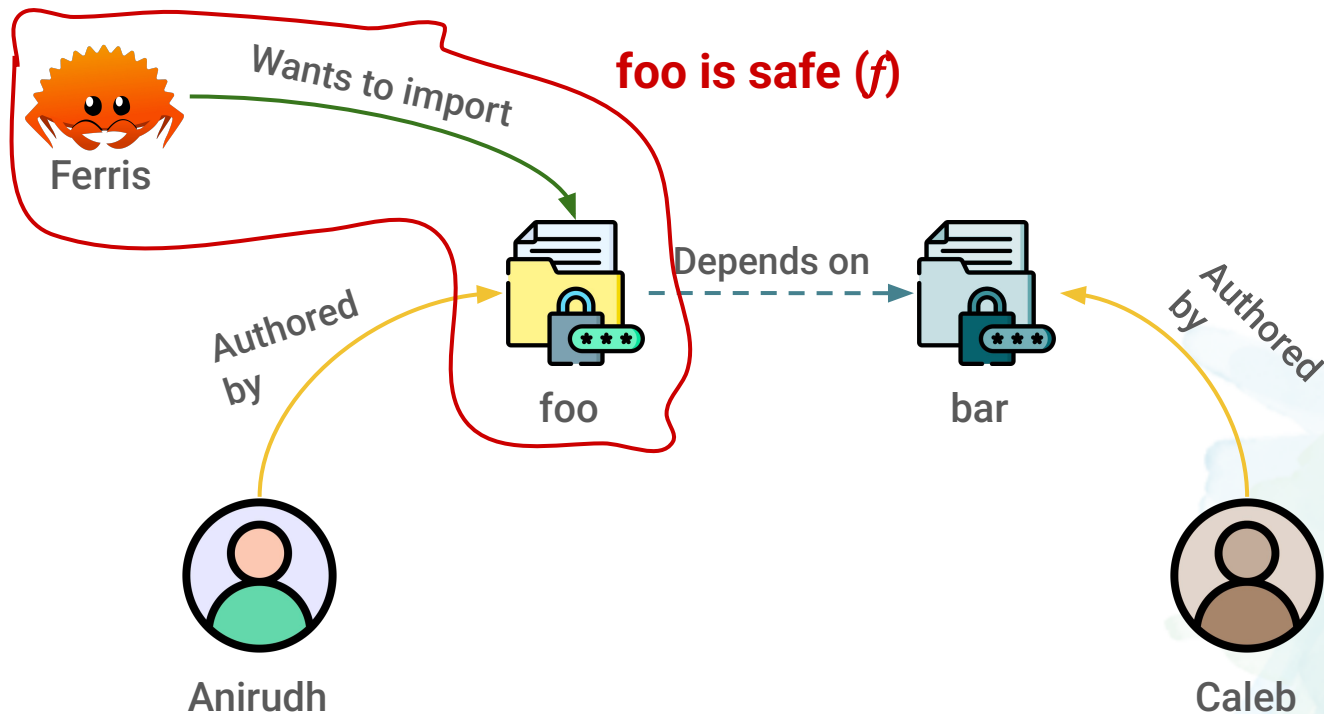


Minimum Trust Problem



No.	Assumption	Encoding	Cost
1	foo is safe.	f	100
2	bar is safe.	b	100
3	foo has many downloads.	d	30
4	If foo has many downloads, then it is safe.	$d \Rightarrow f$	20
5	Anirudh is trustworthy.	a	20
6	Caleb is trustworthy.	c	5
7	If Anirudh is trustworthy and bar is safe, then foo is safe.	$(a \wedge b) \Rightarrow f$	10
8	If Caleb is trustworthy, then bar is safe.	$c \Rightarrow b$	10

Minimum Trust Problem



Minimum Trust Problem

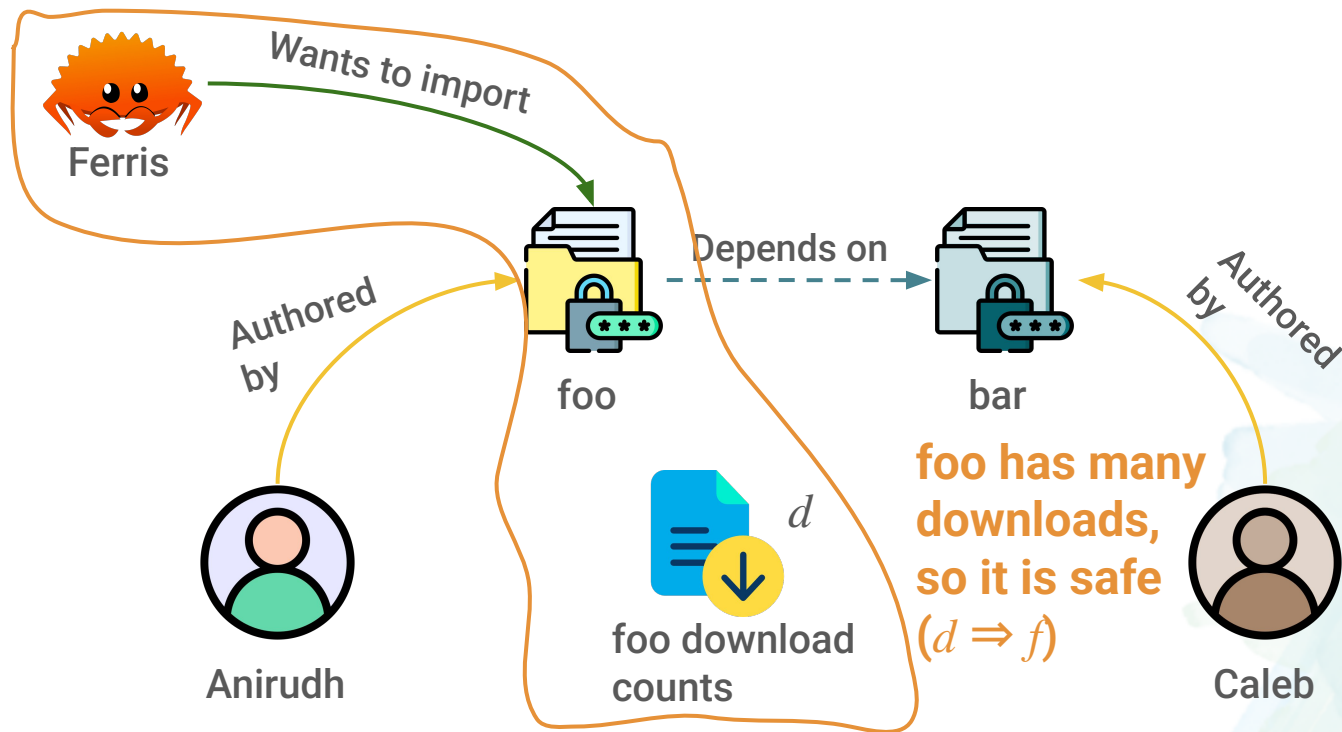


100

No.	Assumption	Encoding	Cost
1	foo is safe.	f	100
2	bar is safe.	b	100
3	foo has many downloads.	d	30
4	If foo has many downloads, then it is safe.	$d \Rightarrow f$	20
5	Anirudh is trustworthy.	a	20
6	Caleb is trustworthy.	c	5
7	If Anirudh is trustworthy and bar is safe, then foo is safe.	$(a \wedge b) \Rightarrow f$	10
8	If Caleb is trustworthy, then bar is safe.	$c \Rightarrow b$	10



Minimum Trust Problem



Minimum Trust Problem

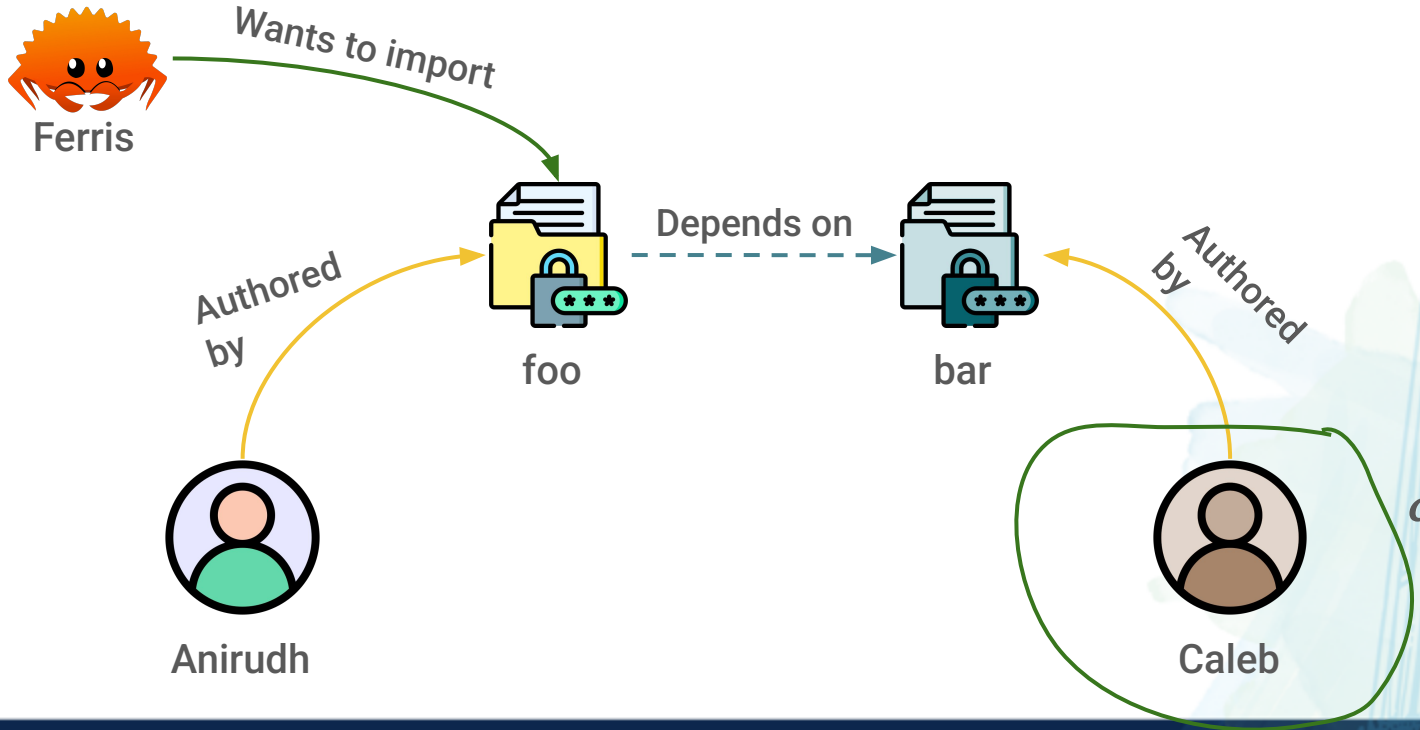


No.	Assumption	Encoding	Cost
1	foo is safe.	f	100
2	bar is safe.	b	100
3	foo has many downloads.	d	30
4	If foo has many downloads, then it is safe.	$d \Rightarrow f$	20
5	Anirudh is trustworthy.	a	20
6	Caleb is trustworthy.	c	5
7	If Anirudh is trustworthy and bar is safe, then foo is safe.	$(a \wedge b) \Rightarrow f$	10
8	If Caleb is trustworthy, then bar is safe.	$c \Rightarrow b$	10

100

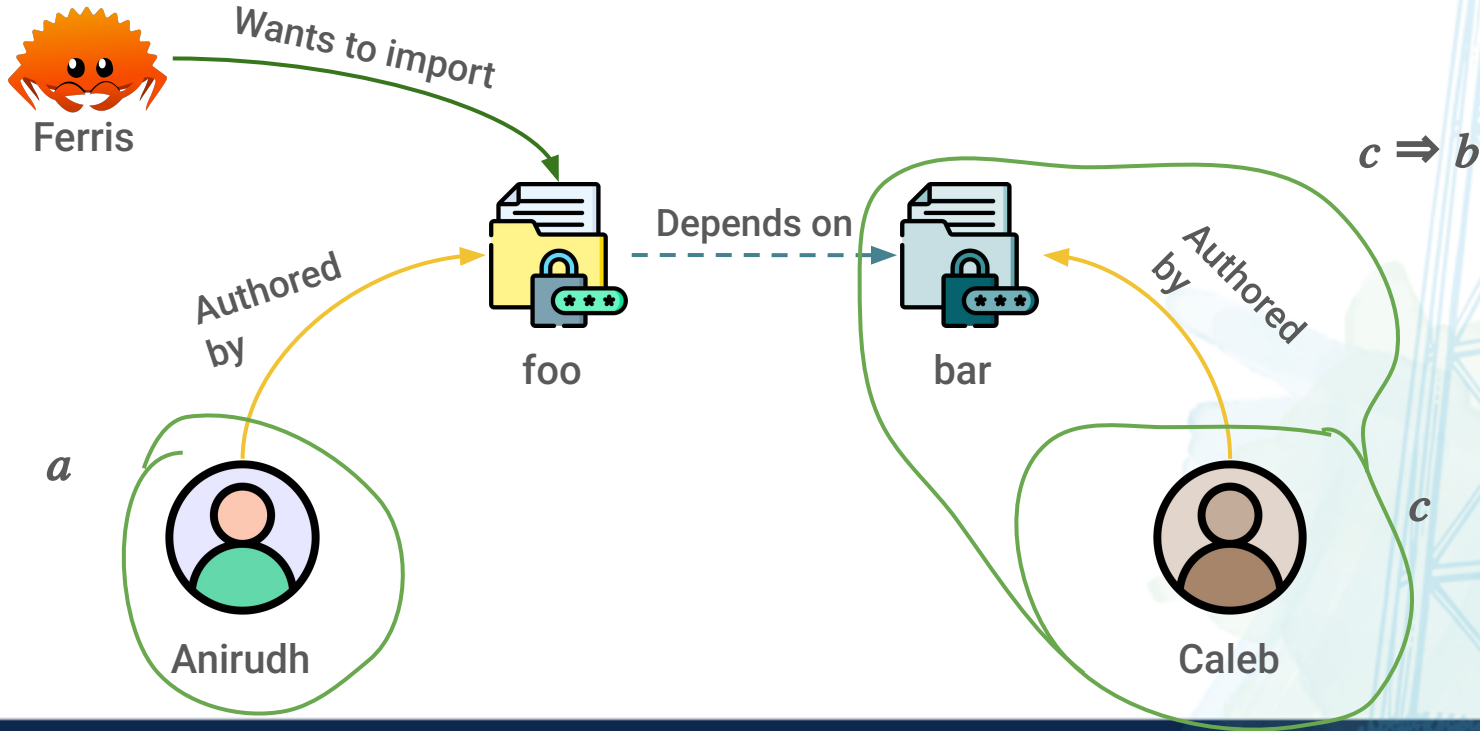
50

Minimum Trust Problem

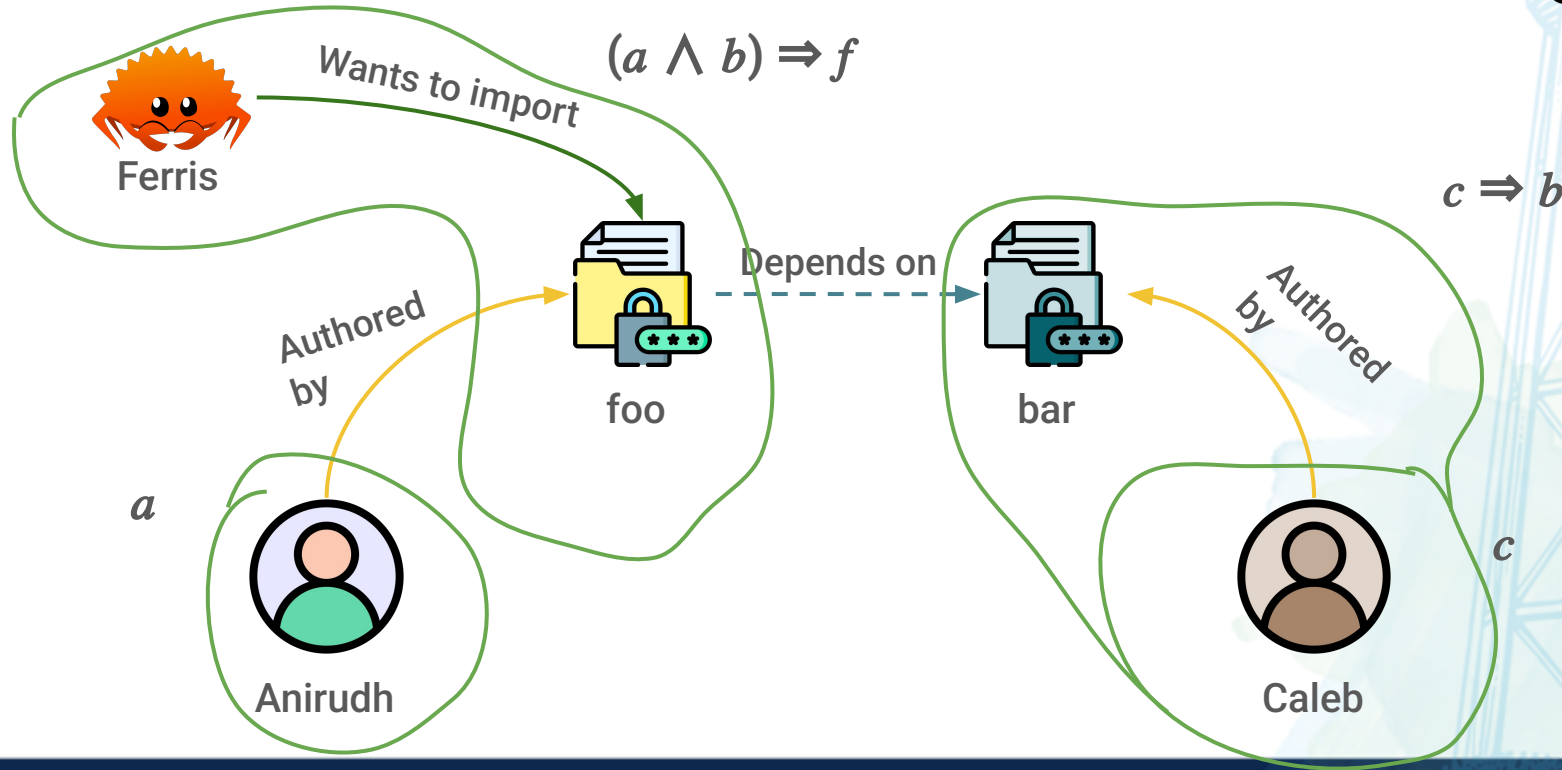




Minimum Trust Problem



Minimum Trust Problem



Minimum Trust Problem



No.	Assumption	Encoding	Cost
1	foo is safe.	f	100
2	bar is safe.	b	100
3	foo has many downloads.	d	30
4	If foo has many downloads, then it is safe.	$d \Rightarrow f$	20
5	Anirudh is trustworthy.	a	20
6	Caleb is trustworthy.	c	5
7	If Anirudh is trustworthy and bar is safe, then foo is safe.	$(a \wedge b) \Rightarrow f$	10
8	If Caleb is trustworthy, then bar is safe.	$c \Rightarrow b$	10

100

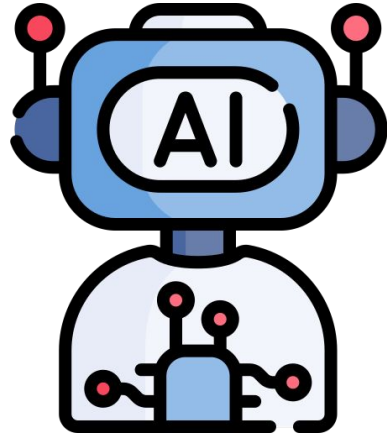
50

45

How can we automate this?



How can we automate this?



Machine Learning

How can we automate this?



Machine Learning

Specific

Auditable

Cargo Sherlock - Goals



1. **Specific** : Trust cost should be specific to an assumption or a set of assumptions.

Cargo Sherlock - Goals



1. **Specific** : Trust cost should be specific to an assumption or a set of assumptions.

If the library is
audited by
Google it is safe

2



Cargo Sherlock - Goals



2. **Auditable** : It should be possible for human to read and understand and trace back the assumptions made by analysis.

Cargo Sherlock - Goals



2. **Auditable** : It should be possible for human to read and understand and trace back the assumptions made by analysis.

If Author X is
trusted than
library is safe

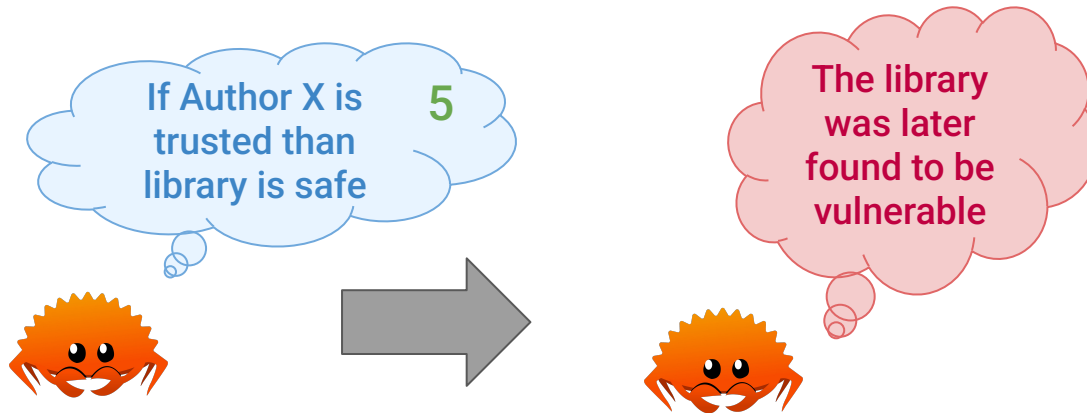
5



Cargo Sherlock - Goals



2. **Auditable** : It should be possible for human to read and understand and trace back the assumptions made by analysis.



Cargo Sherlock - Goals



2. **Auditable** : It should be possible for human to read and understand and trace back the assumptions made by analysis.





Cargo Sherlock - Solution

SMT $\rightarrow$ Satisfiability Modulo Theories

- Checks whether a given formula is **satisfiable**.
- Extends SAT with theories such as:
 - Arithmetic
 - Arrays
 - strings

Example:

- $x > 3 \wedge x < 10 \rightarrow$ **satisfiable**
- $x > 3 \wedge x < 2 \rightarrow$ **unsatisfiable**



Cargo Sherlock - Solution

SMT $\rightarrow$ Satisfiability Modulo Theories

- Checks whether a given formula is **satisfiable**.
- Extends SAT with theories such as:
 - Arithmetic
 - Arrays
 - strings

Example:

- $x > 3 \wedge x < 10 \rightarrow$ **satisfiable**
- $x > 3 \wedge x < 2 \rightarrow$ **unsatisfiable**

With an SMT solver we
can automate the solving
of Minimum Trust
Problem!

Z3

Naive Solution

No.	Encoding	Cost
1	f	100
2	b	100
3	d	30
4	$d \Rightarrow f$	20
5	a	20
6	c	5
7	$(a \wedge b) \Rightarrow f$	10
8	$c \Rightarrow b$	10

Naive Solution

1. Use variables to track assumptions being made
 - a_1 is true iff you make assumption no. 1

No.	Encoding	Cost
1	f	100
2	b	100
3	d	30
4	$d \Rightarrow f$	20
5	a	20
6	c	5
7	$(a \wedge b) \Rightarrow f$	10
8	$c \Rightarrow b$	10

Naive Solution

1. Use variables to track assumptions being made
 - a_1 is true iff you make assumption no. 1
2. Define a first-order constraint on the minimum cost set of assumptions that prove the library is safe
 - Call this minimum cost set $a_1^*, a_2^*, \dots, a_8^*$

No.	Encoding	Cost
1	f	100
2	b	100
3	d	30
4	$d \Rightarrow f$	20
5	a	20
6	c	5
7	$(a \wedge b) \Rightarrow f$	10
8	$c \Rightarrow b$	10

Naive Solution

1. Use variables to track assumptions being made
 - a_1 is true iff you make assumption no. 1
2. Define a first-order constraint on the minimum cost set of assumptions that prove the library is safe
 - Call this minimum cost set $a_1^*, a_2^*, \dots, a_8^*$

In this example, we make assumptions $a_5^*, a_6^*, a_7^*, a_8^*$ so $a_5^*, a_6^*, a_7^*, a_8^* = \text{true}$

No.	Encoding	Cost
1	f	100
2	b	100
3	d	30
4	$d \Rightarrow f$	20
5	a	20
6	c	5
7	$(a \wedge b) \Rightarrow f$	10
8	$c \Rightarrow b$	10

Naive Solution

Given constraint: $\forall a_1, a_2, \dots, a_8, f, b, d, a, c$

$$\neg[a_1 \Rightarrow f \wedge a_2 \Rightarrow b \wedge a_3 \Rightarrow d \wedge a_4 \Rightarrow (d \Rightarrow f) \wedge$$

$$a_5 \Rightarrow a \wedge a_6 \Rightarrow c \wedge a_7 \Rightarrow (a \wedge b \Rightarrow f) \wedge a_8 \Rightarrow (c \Rightarrow s) \wedge \neg f \\ \wedge$$

$$\text{cost}(a_1^*, a_2^*, \dots, a_8^*) \leq \text{cost}(a_1, a_2, \dots, a_8)] \wedge$$

$$\neg[a_1^* \Rightarrow f \wedge a_2^* \Rightarrow b \wedge a_3^* \Rightarrow d \wedge a_4^* \Rightarrow (d \Rightarrow f) \wedge$$

$$a_5^* \Rightarrow a \wedge a_6^* \Rightarrow c \wedge a_7^* \Rightarrow (a \wedge b \Rightarrow f) \wedge a_8^* \Rightarrow (c \Rightarrow s) \\ \wedge \neg f]$$

Find a satisfying assignment for $(a_1^*, a_2^*, \dots, a_8^*)$.

Z3

Naive Solution

Given constraint: $\forall a_1, a_2, \dots, a_8, f, b, d, a, c$

$\neg[a_1 \Rightarrow f \wedge a_2 \Rightarrow b \wedge a_3 \Rightarrow d \wedge a_4 \Rightarrow (d \Rightarrow f) \wedge$

$a_5 \Rightarrow a \wedge a_6 \Rightarrow c \wedge a_7 \Rightarrow (a \wedge b \Rightarrow f) \wedge a_8 \Rightarrow (c \Rightarrow s) \wedge \neg f \wedge$

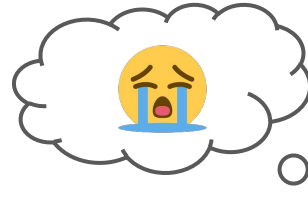
$\text{cost}(a_1^*, a_2^*, \dots, a_8^*) \leq \text{cost}(a_1, a_2, \dots, a_8)] \wedge$

$\neg[a_1^* \Rightarrow f \wedge a_2^* \Rightarrow b \wedge a_3^* \Rightarrow d \wedge a_4^* \Rightarrow (d \Rightarrow f) \wedge$

$a_5^* \Rightarrow a \wedge a_6^* \Rightarrow c \wedge a_7^* \Rightarrow (a \wedge b \Rightarrow f) \wedge a_8^* \Rightarrow (c \Rightarrow s) \wedge \neg f]$

Find a **satisfying assignment** for $(a_1^*, a_2^*, \dots, a_8^*)$.

Too many quantifiers! Computationally expensive - $\forall \exists$ problem



Z3

Naive Solution

Quantifier prefix: $\forall a_1, a_2, \dots, a_8, f, b, d, a, c$

- Number of quantifiers / quantified variables grows with problem size
- Doesn't scale for libraries with many dependencies!

No.	Encoding	Cost
1	f	100
2	b	100
3	d	30
4	$d \Rightarrow f$	20
5	a	20
6	c	5
7	$(a \wedge b) \Rightarrow f$	10
8	$c \Rightarrow b$	10

Revisiting the problem

Goal: prove f with the given assumptions



Prove f

No.	Encoding	Cost
1	f	100
2	b	100
3	d	30
4	$d \Rightarrow f$	20
5	a	20
6	c	5
7	$(a \wedge b) \Rightarrow f$	10
8	$c \Rightarrow b$	10

Revisiting the problem

How can we prove f ? Three ways:

- Make assumption 1 (a_1)



No.	Encoding	Cost
1	f	100
2	b	100
3	d	30
4	$d \Rightarrow f$	20
5	a	20
6	c	5
7	$(a \wedge b) \Rightarrow f$	10
8	$c \Rightarrow b$	10

Revisiting the problem

How can we prove f ? Three ways:

- Make assumption 1 (a_1)
- Make assumption 4 (a_4) and prove d

Prove a_1 OR ($a_4 \wedge d$)



No.	Encoding	Cost
1	f	100
2	b	100
3	d	30
4	$d \Rightarrow f$	20
5	a	20
6	c	5
7	$(a \wedge b) \Rightarrow f$	10
8	$c \Rightarrow b$	10

Revisiting the problem

How can we prove f ? Three ways:

- Make assumption 1 (a_1)
- Make assumption 4 (a_4) and prove d
- Make assumption 7 (a_7) and prove $a \wedge b$

Prove a_1 OR ($a_4 \wedge d$)
OR ($a_7 \wedge a \wedge b$)



No.	Encoding	Cost
1	f	100
2	b	100
3	d	30
4	$d \Rightarrow f$	20
5	a	20
6	c	5
7	$(a \wedge b) \Rightarrow f$	10
8	$c \Rightarrow b$	10

Revisiting the problem

Now, how can we prove d ? Only one way:

- Make assumption 3 (a_3)



Prove a_1 OR $(a_4 \wedge a_3)$
OR $(a_7 \wedge a \wedge b)$



No.	Encoding	Cost
1	f	100
2	b	100
3	d	30
4	$d \Rightarrow f$	20
5	a	20
6	c	5
7	$(a \wedge b) \Rightarrow f$	10
8	$c \Rightarrow b$	10

Revisiting the problem

Repeat this process.

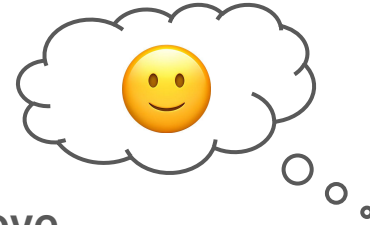
Eventually...

Prove a_1 OR $(a_4 \wedge a_3)$ OR
 $(a_7 \wedge a_5 \wedge (a_2 \vee (a_8 \wedge$
 $a_6)))$



No.	Encoding	Cost
1	f	100
2	b	100
3	d	30
4	$d \Rightarrow f$	20
5	a	20
6	c	5
7	$(a \wedge b) \Rightarrow f$	10
8	$c \Rightarrow b$	10

Horn Clause Solution



This tells us exactly what sets of assumptions prove the conclusion **without quantifiers!**

This process only works with Horn clause assumptions

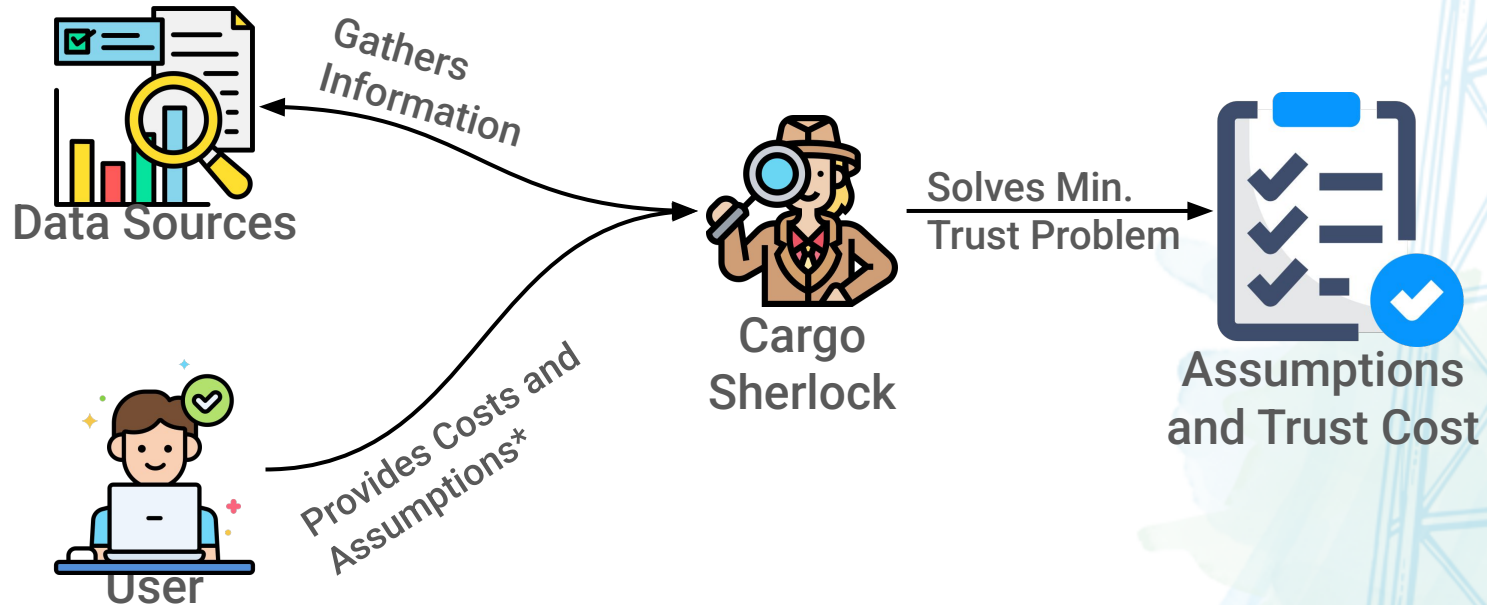
Prove a_1 OR $(a_4 \wedge a_3)$ OR
 $(a_7 \wedge a_5 \wedge (a_2 \vee (a_8 \wedge$
 $a_6)))$

+ minimize costs

Z3



Design and Implementation - Overview



* Cargo Sherlock comes with default list of assumptions

Design and Implementation - Data Collection



Crates.io



- Author
- Downloads

RustSec



Vulnerability Info

Static Analysis Tools



- Miri
- Cargo Scan

GitHub



- Stars
- Forks

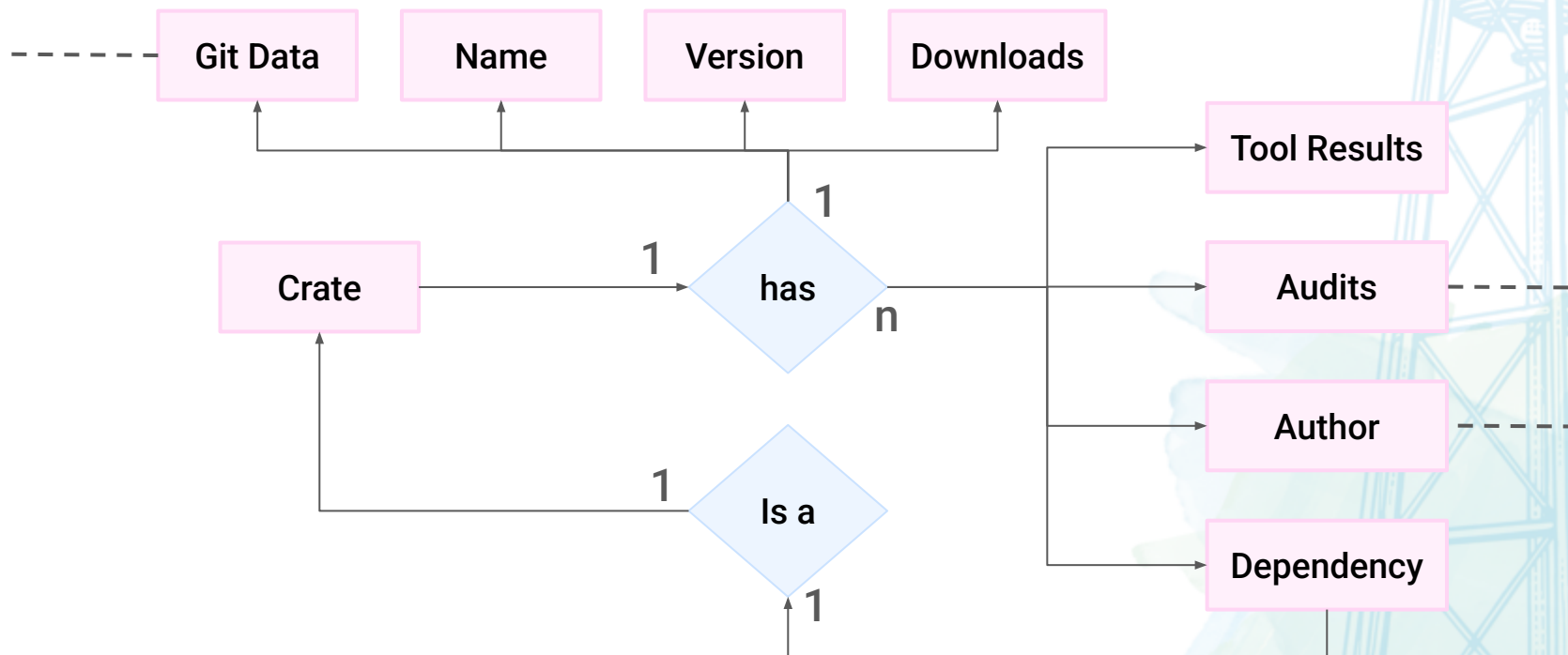
Audit Files



- Google
- Mozilla
- ByteCode Alliance

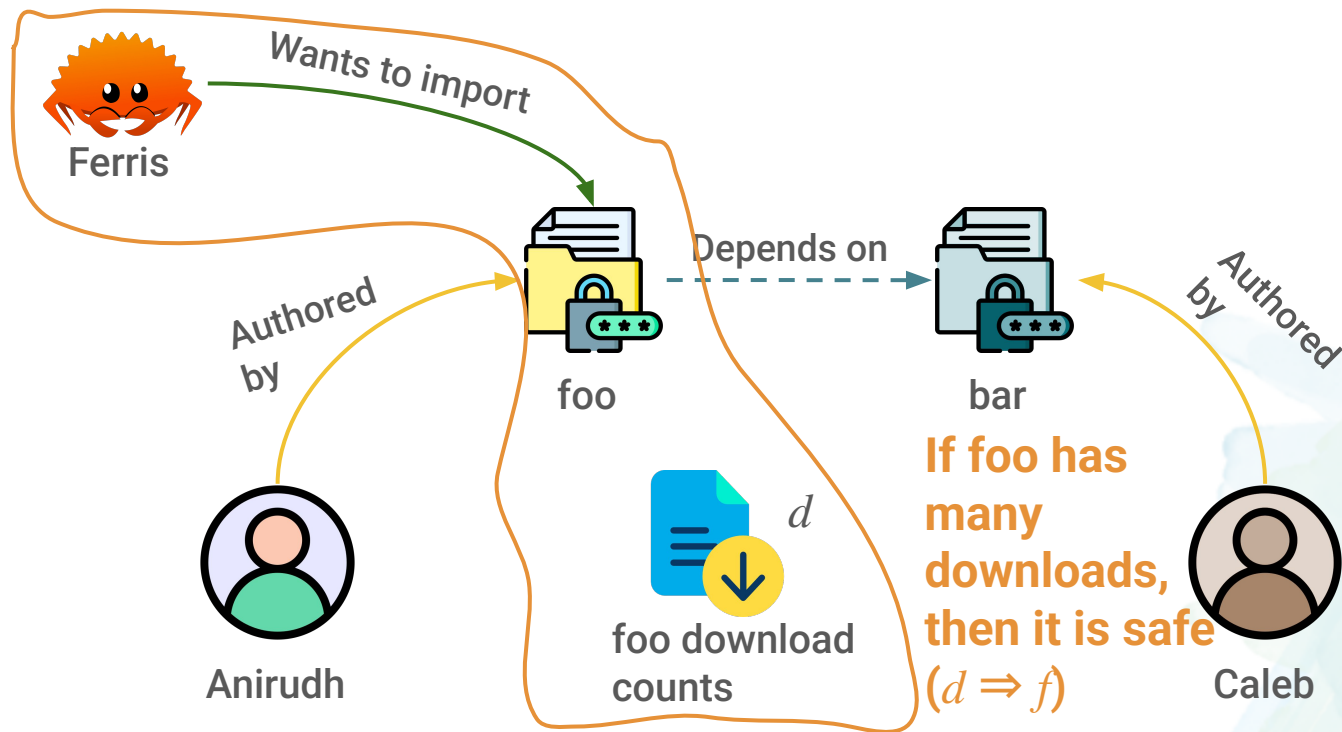


Design and Implementation - Relational Model



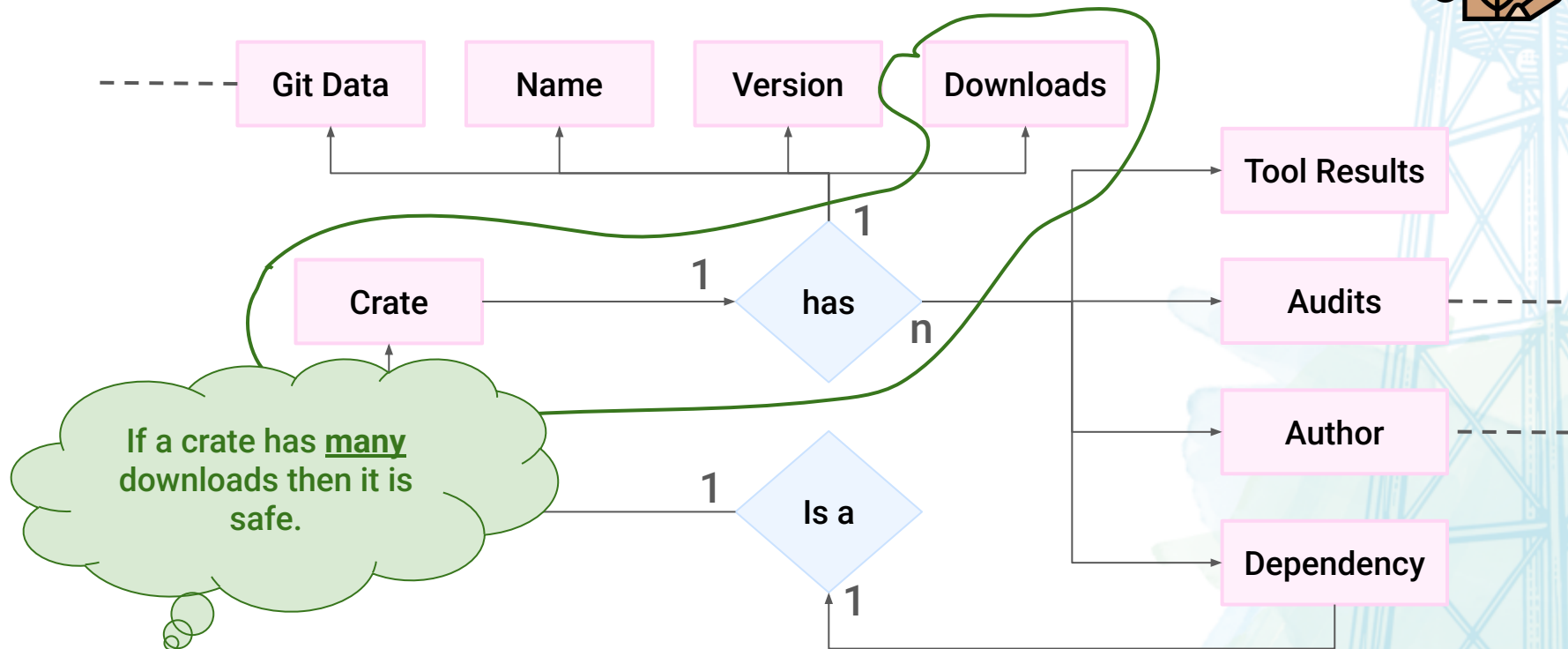


Minimum Trust Problem





Design and Implementation - Relational Model



Design and Implementation - Assumptions



- What does many mean?

If a crate has many downloads then it is safe.

Design and Implementation - Assumptions



1. Parameterized: assumptions with variable trust cost.

If a crate has many downloads then it is safe.

Design and Implementation - Assumptions



1. Parameterized: assumptions with variable trust cost.

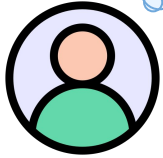
If a crate has many downloads then it is safe.

2. Fixed : assumptions with fixed trust cost.

If a library has a passed audit then it is safe

Design and Implementation - Selection of Assumptions

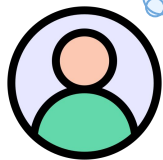
51% of developers
consider developer
reputation before
adding/reviewing
dependency



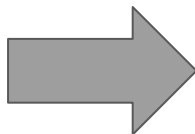
Hamer et al. [21]

Design and Implementation - Selection of Assumptions

51% of developers
consider developer
reputation before
adding/reviewing
dependency



Hamer et al. [21]



If the repo has many
stars and forks, then
it is trusted.

If the author is in
the trusted list,
then the library is
safe



Design and Implementation - Selection of Assumptions

Default assumptions are based on prior work
and our system can accommodate more
custom assumptions as well!



Extending the Model - Negative Assumptions

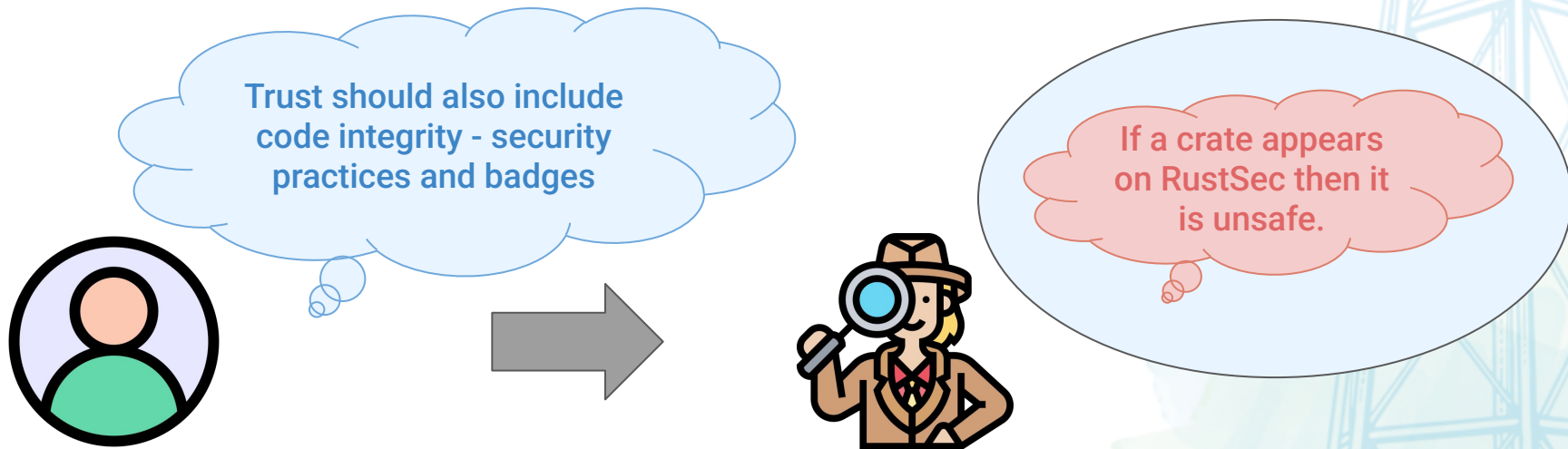


Assumptions that a user may use to perceive that a crate is unsafe.

Extending the Model - Negative Assumptions

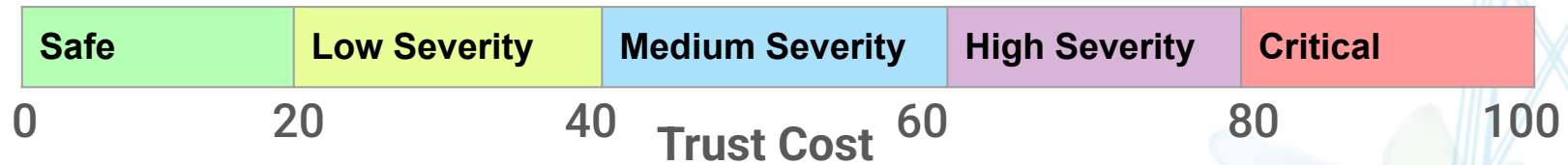


Assumptions that a user may use to perceive that a crate is unsafe.



Boughton et al. [6]

Design and Implementation - Assigning Label



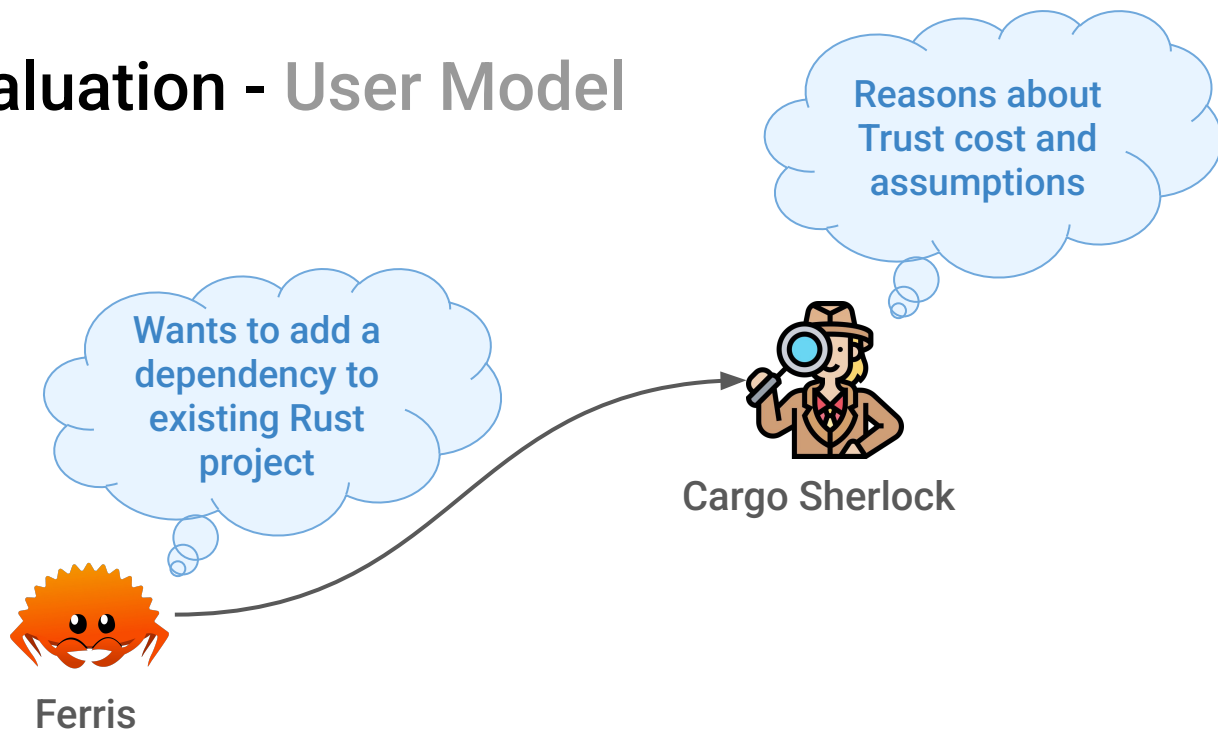
Evaluation - Overview

1. Do we detect synthetically generated typosquatting attacks?
2. Do we detect real world supply chain risks?
3. How do we evaluate on known vulnerable crates?
4. How does performance and scalability of both algorithms compare?

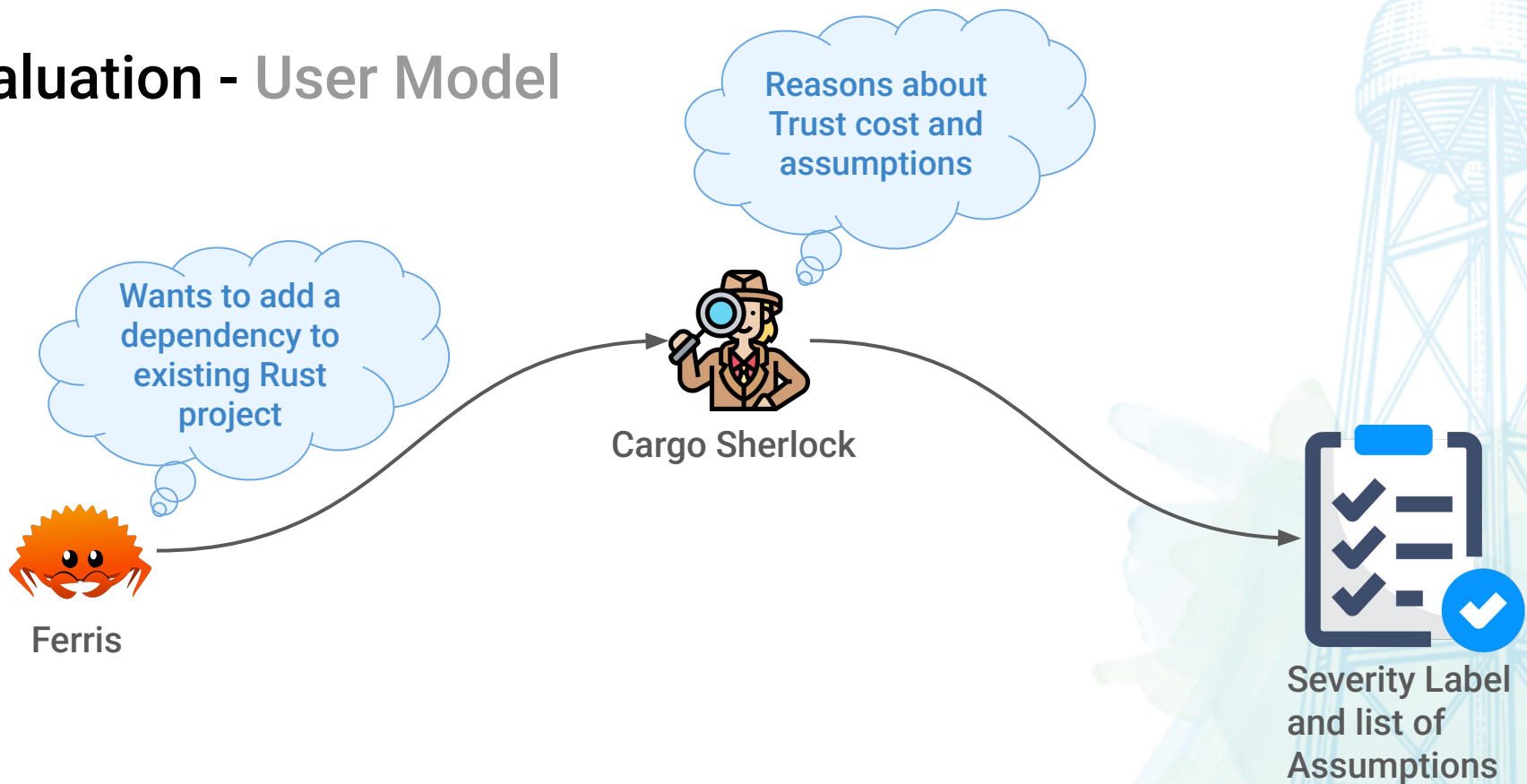
Evaluation - User Model



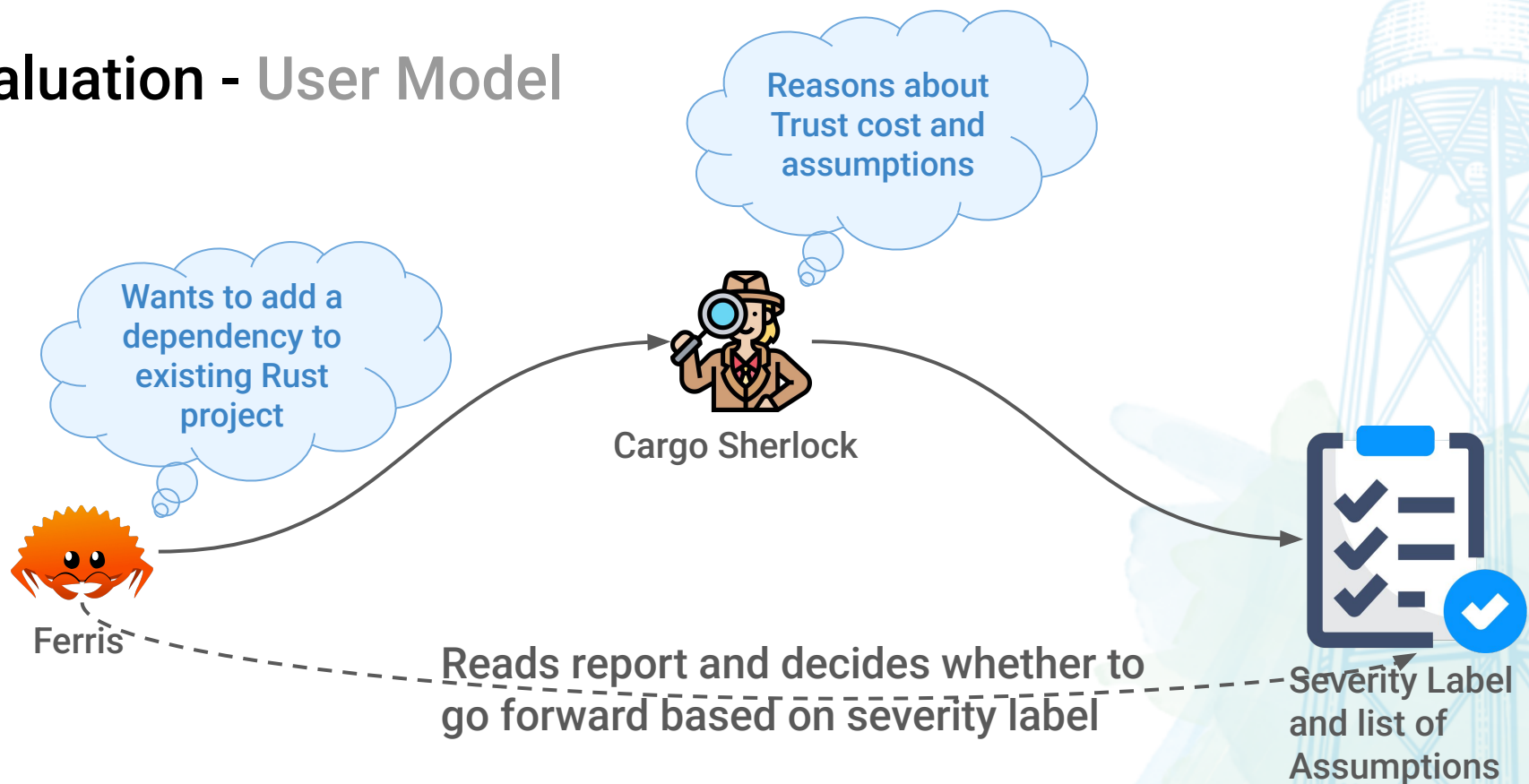
Evaluation - User Model



Evaluation - User Model



Evaluation - User Model

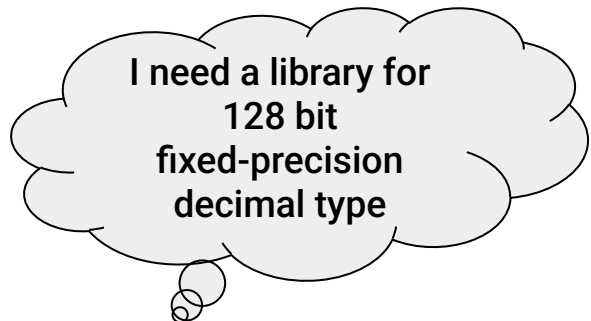


Evaluation - Overview

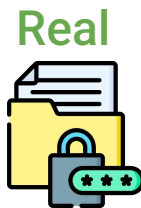
- Typosquatting Attack : exploits typographical errors to deceive a user into downloading a malicious package.

Evaluation - Overview

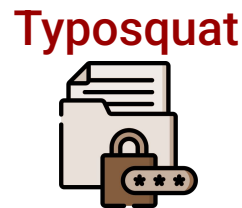
- Typosquatting Attack



Ferris



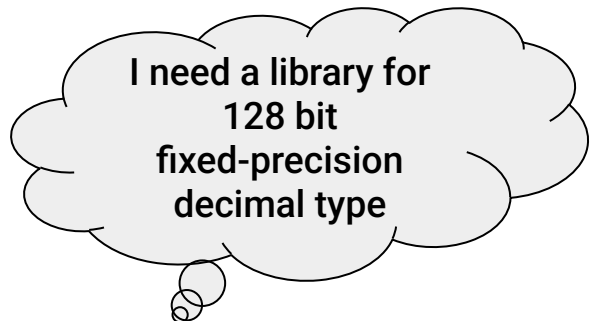
rust_decimal



rustdecimal

Evaluation - Overview

- Typosquatting Attack



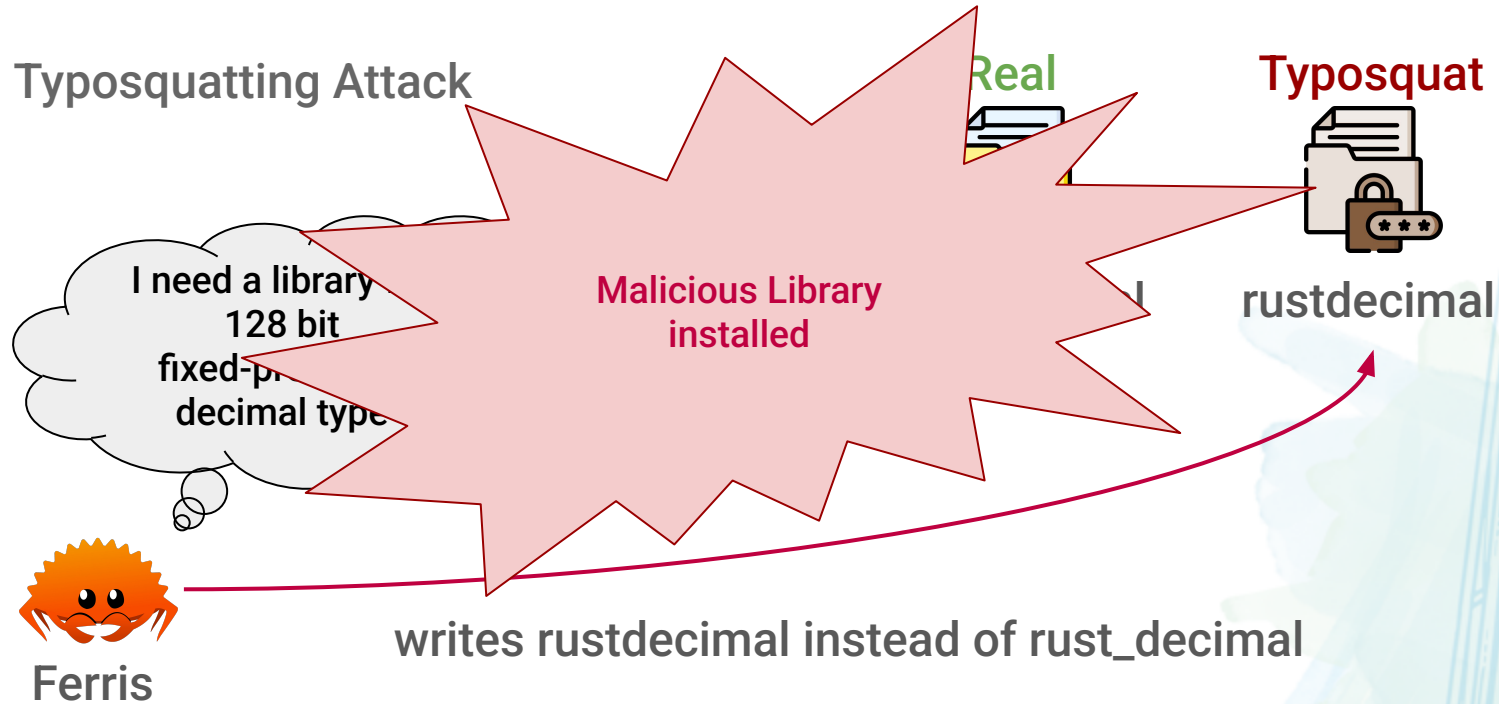
Ferris

writes rustdecimal instead of rust_decimal

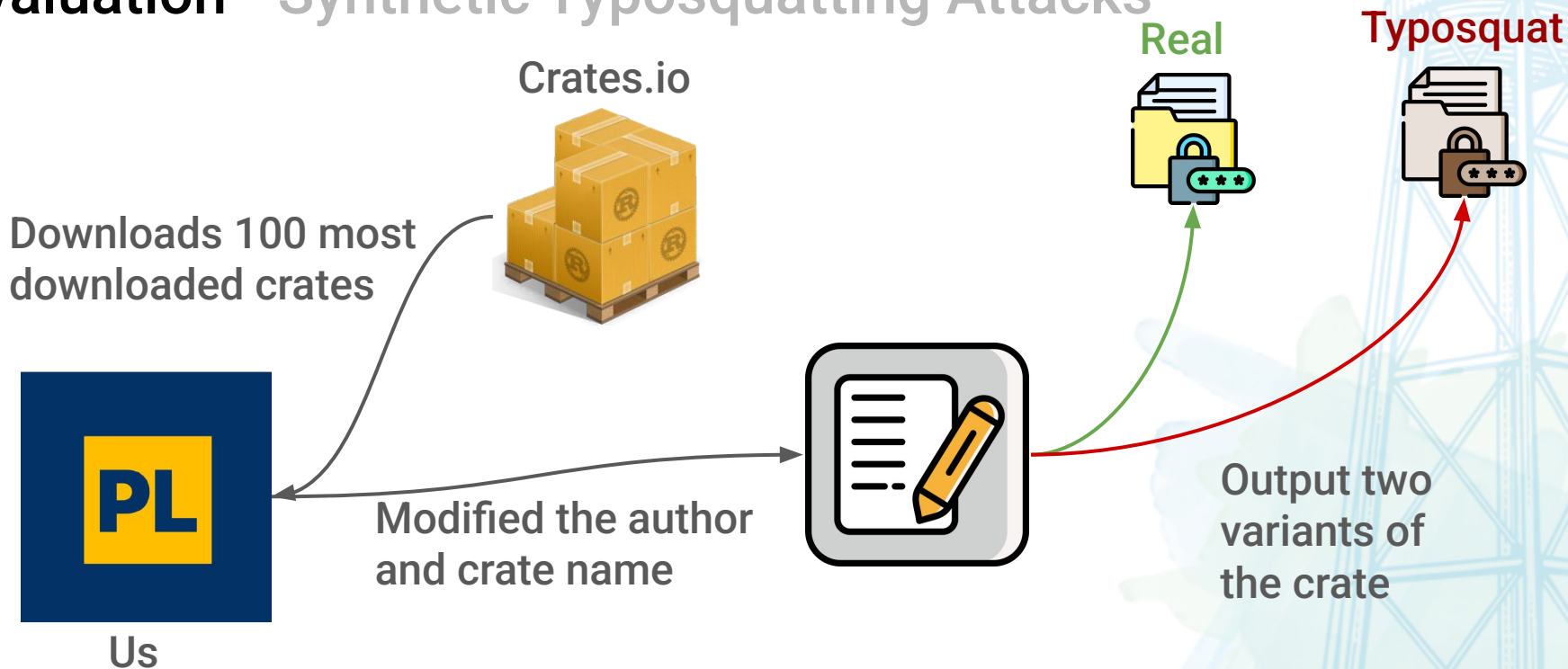


Evaluation - Overview

- Typosquatting Attack



Evaluation - Synthetic Typosquatting Attacks



Evaluation - Synthetic Typosquatting Attacks



Real

Label	Count
Safe	73
Low Severity	27
Medium Severity	0
High Severity	0
Critical	0

Evaluation - Synthetic Typosquatting Attacks



Real

Label	Count
Safe	73
Low Severity	27
Medium Severity	0
High Severity	0
Critical	0



Typosquat

Label	Count
Safe	25
Low Severity	0
Medium Severity	0
High Severity	3
Critical	72

Evaluation - Synthetic Typosquatting Attacks



Real

Label	Count
Safe	
Se	
Medium Severity	
High Severity	0
Critical	0

A user interacting with the tool would identify the typosquatted crate variant when adding a dependency to their code.



Label	Count
Severity	
High Severity	3
Critical	72

Evaluation - Real-World Supply Chain Risks



Evaluation - Real-World Supply Chain Risks

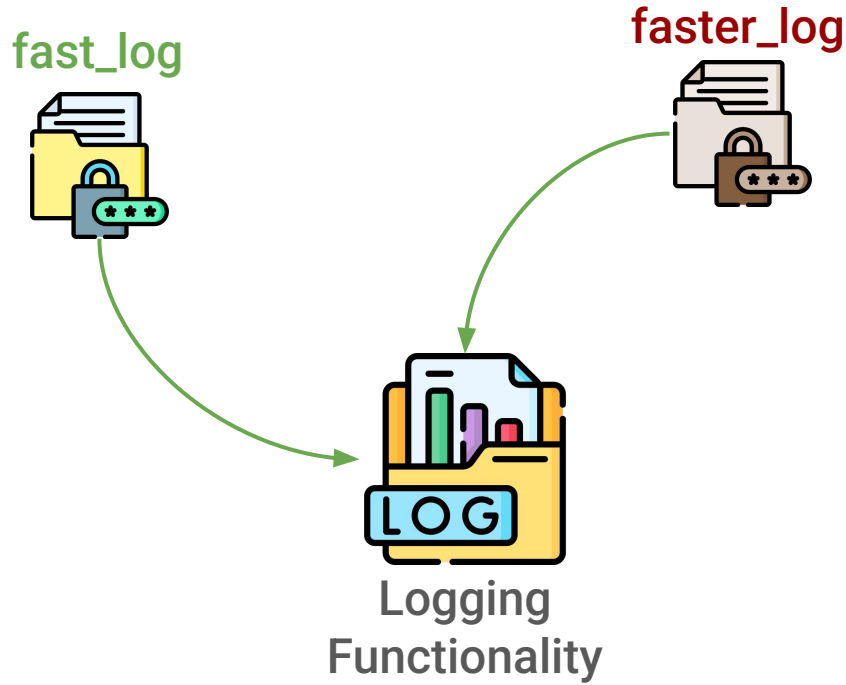
Security advisory: malicious crates.io: Malicious crates faster_log and async_println

Ma

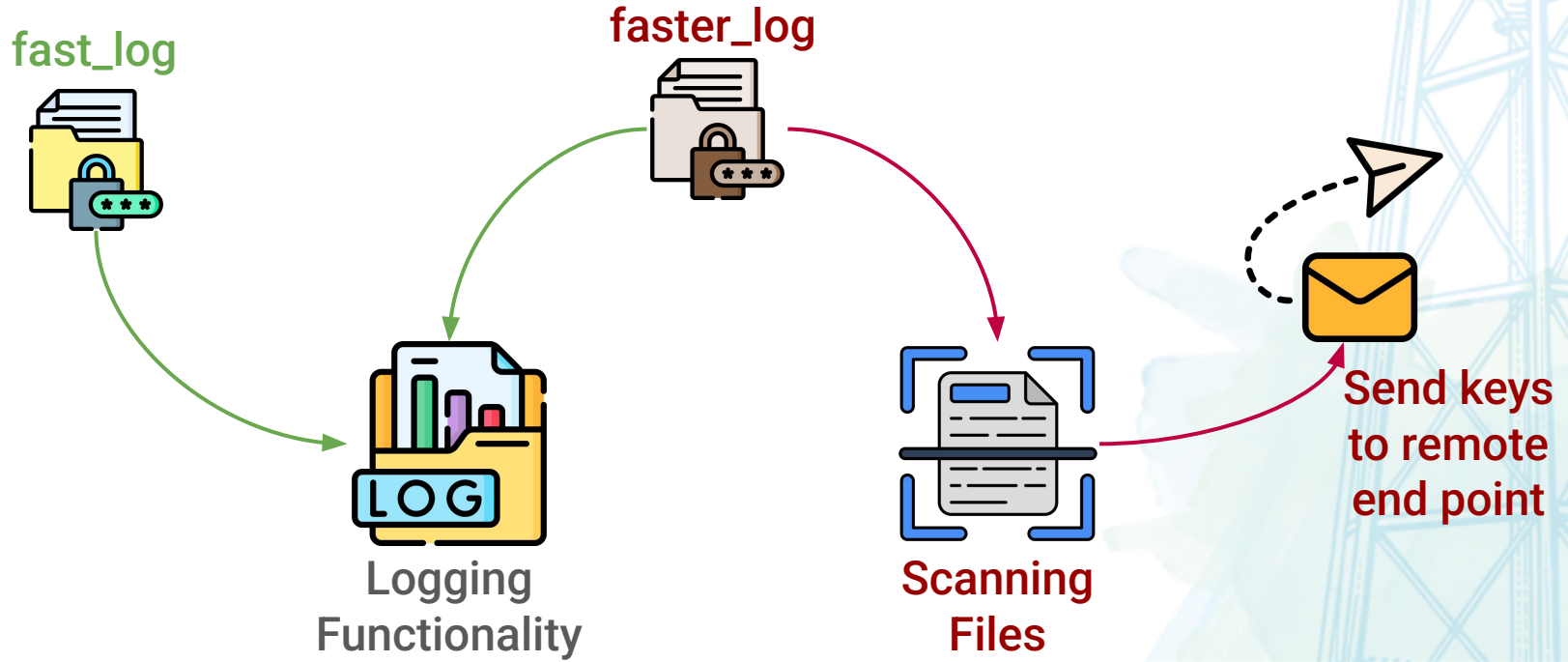
Sept. 24, 2025 · Walter Pearce on behalf of [the crates.io team](#)

On September 24th, the crates.io team was notified by Kirill Boychenko from the Socket Threat Research Team of two malicious crates which were actively searching file contents for Ethereum private keys, Solana private keys, and arbitrary byte arrays for exfiltration.

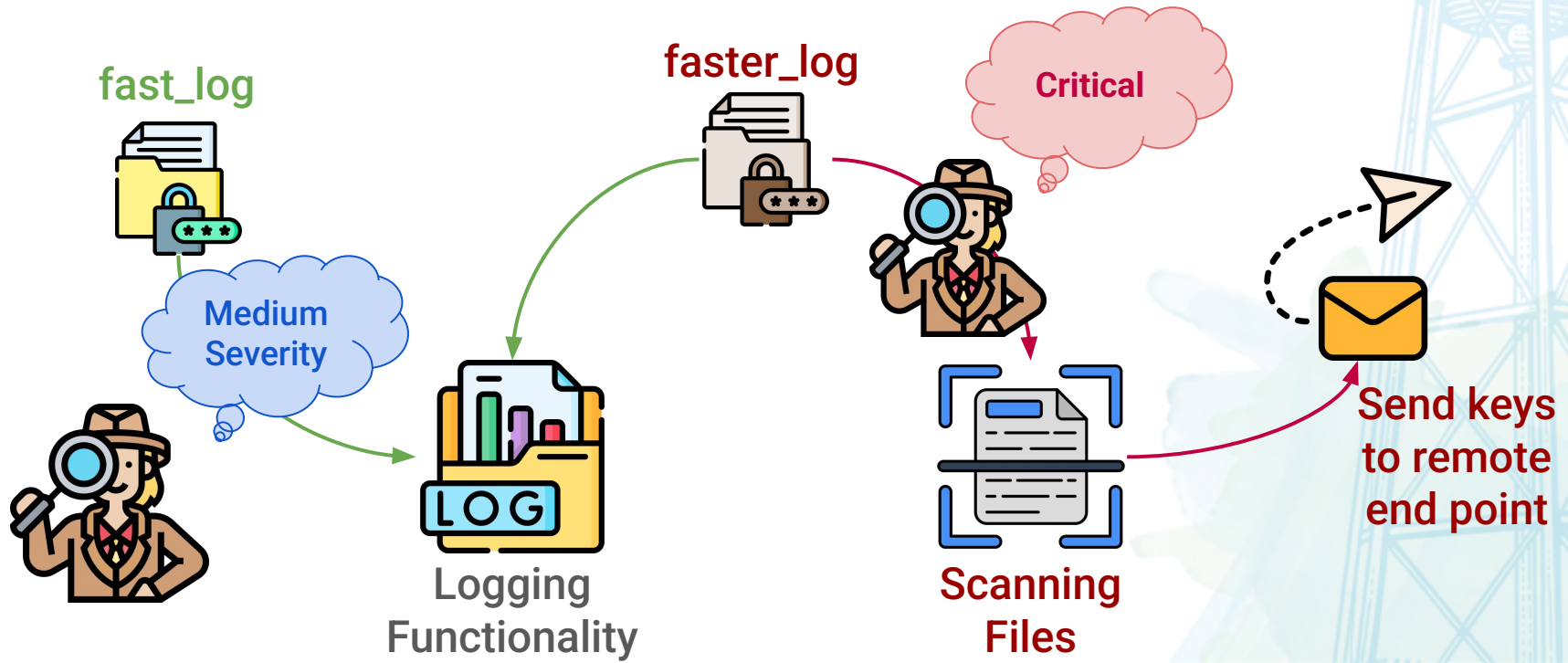
Evaluation - Real-World Supply Chain Risks - faster_log



Evaluation - Real-World Supply Chain Risks - faster_log



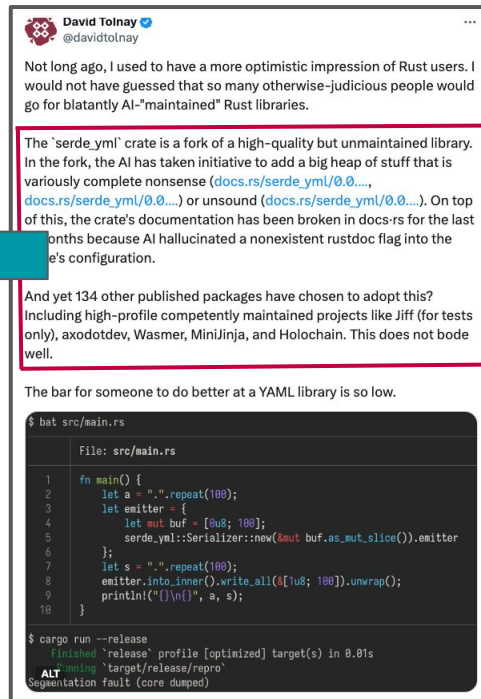
Evaluation - Real-World Supply Chain Risks - faster_log



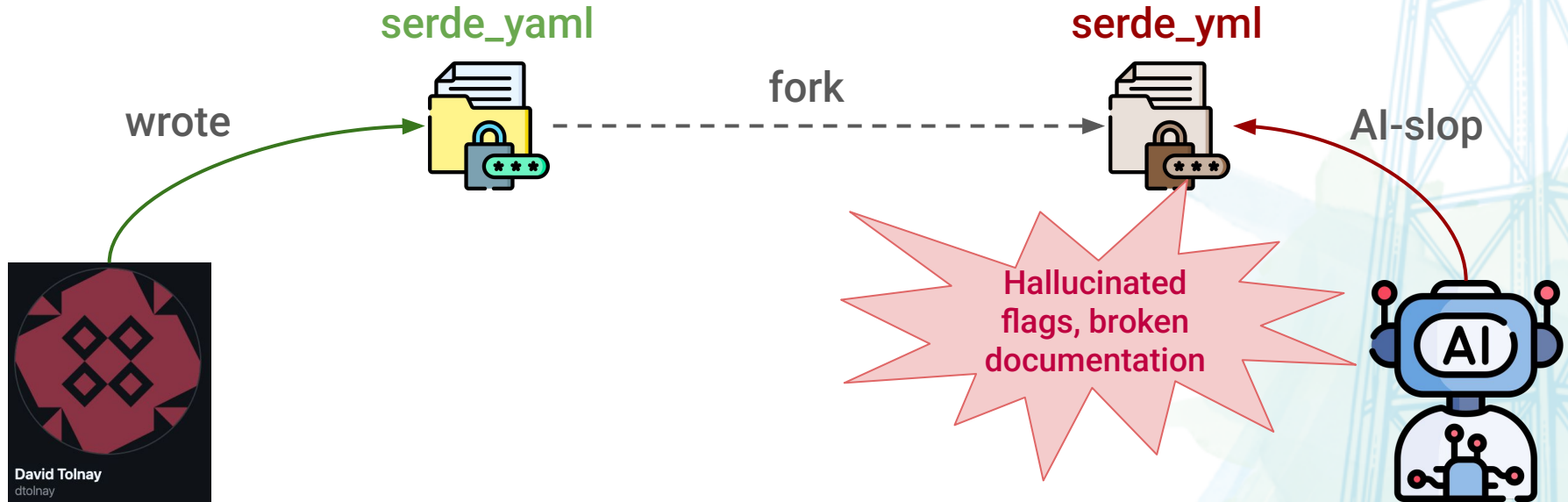
Evaluation - Real-World Supply Chain Risks

- AI fork
- Hallucinated Non Existent Flag
- Documentation Broken
- Adopted by 134 high profile packages!!

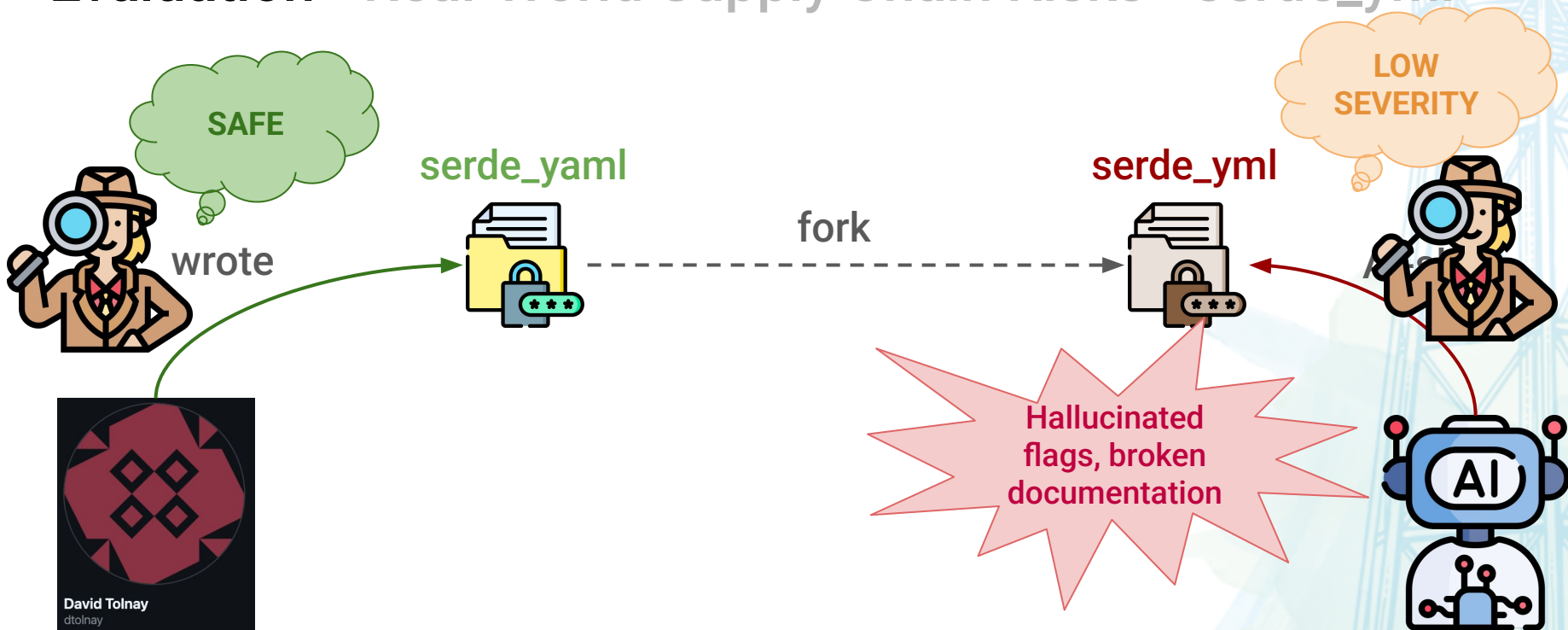
Source : <https://x.com/davidtolnay/status/1883906113428676938>



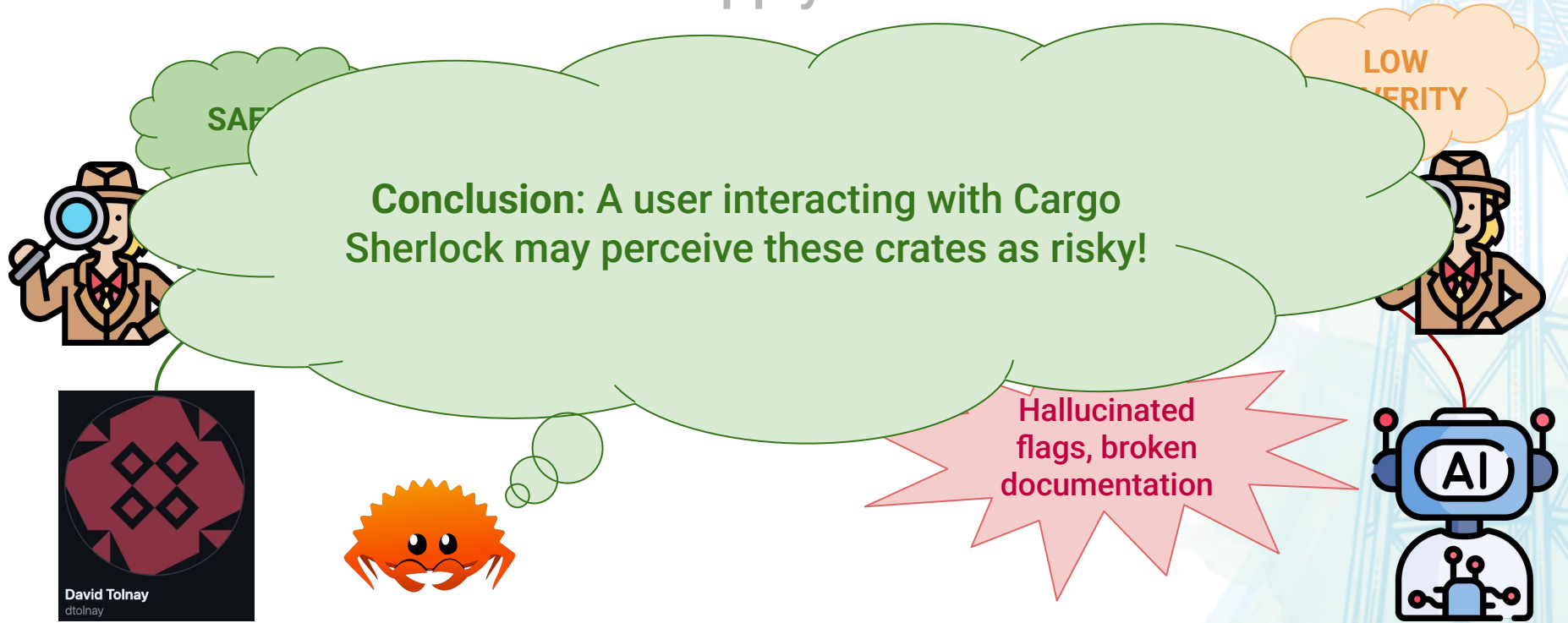
Evaluation - Real-World Supply Chain Risks - serde_yaml



Evaluation - Real-World Supply Chain Risks - serde_yaml



Evaluation - Real-World Supply Chain Risks



Cargo Sherlock - Conclusion



- Third Party Dependencies
- Minimum Trust Problem
- Implementation for Rust Ecosystem.
- Evaluation on synthetic and real world supply chain attacks.



Paper PDF

Cargo Sherlock - Performance of Algorithms



4. How does the performance of Naive and Heuristic algorithm scale with number of dependencies?

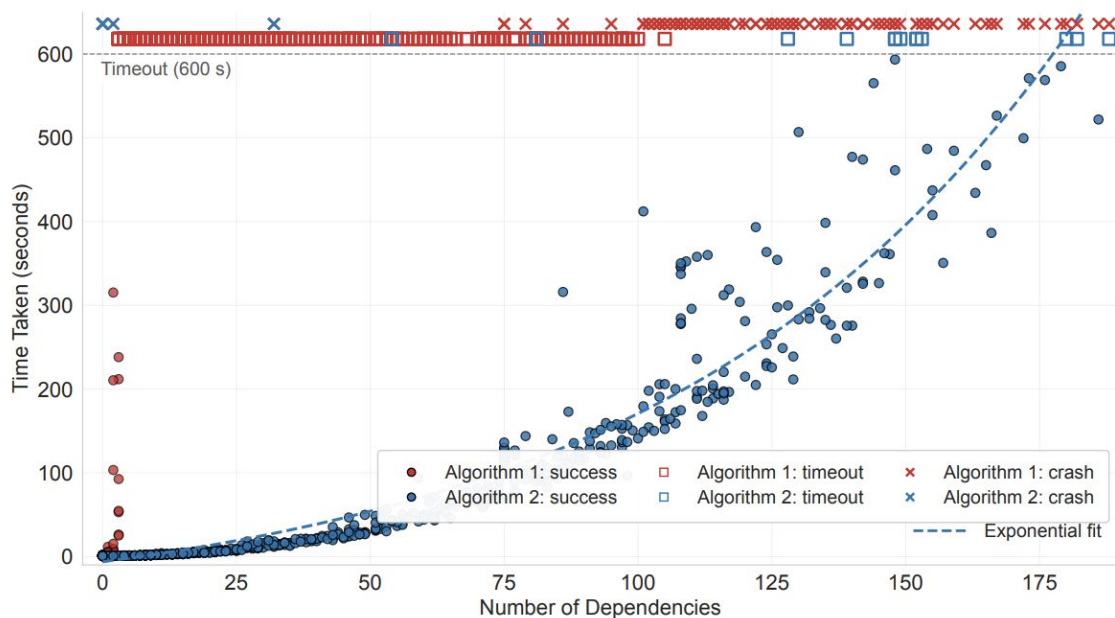
Cargo Sherlock - Performance of Algorithms



Cargo Sherlock - Performance of Algorithms



Algorithm 1 becomes infeasible after 10 dependencies, while Algorithm 2 scales to at least 150 dependencies!



(b) Time taken vs. number of dependencies.

Cargo Sherlock - Conclusion



- Third Party Dependencies
- Minimum Trust Problem
- Implementation for Rust Ecosystem.
- Evaluation on synthetic and real world supply chain attacks.



Paper PDF