inesc id lisboa
Instituto de Engenharia de Sistemas e Computadores Investigação e Desenvolvimento em Lisboa

TÉCNICO LISBOA

FCT
Fundação para a Ciência e a Tecnologia
MINISTÉRIO DA EDUCAÇÃO E CIÊNCIA

U.PORTO
FEUP FACULDADE DE ENGENHARIA UNIVERSIDADE DO PORTO

VIRTUS IMPAVIDA
UFPE

Leveraging Practitioners' Feedback to Improve a Security Linter 👷⚙️

Sofia Reis (@sofiaoreis)
Rui Abreu (@rmaranhao)
Marcelo d'Amorim (@marcelodamorim)
Daniel Fortunato

Automated Software Engineering (ASE'22) | October 2022

Motivation > Security Vulnerabilities in Infrastructure-as-Code Scripts

🚀 Software configuration management and deployment tools like **Puppet** became popular amongst software development warehouses.

Motivation > Security Vulnerabilities in Infrastructure-as-Code Scripts

🚀 Software configuration management and deployment tools like **Puppet** became popular amongst software development warehouses.

👷 These tools help infrastructure teams **increase productivity** by automating various config tasks (e.g., server setup) through scripts that can be *reused* and *versioned*.

Motivation > Security Vulnerabilities in Infrastructure-as-Code Scripts

🚀 Software configuration management and deployment tools like **Puppet** became popular amongst software development warehouses.

👷 These tools help infrastructure teams **increase productivity** by automating various config tasks (e.g., server setup) through scripts that can be *reused* and *versioned*.

💀 As with any piece of code, IaC scripts are also prone to defects such as **security vulnerabilities**.

Motivation > Security Vulnerabilities in Infrastructure-as-Code Scripts

🚀 Software configuration management and deployment tools like **Puppet** became popular amongst software development warehouses.

👷 These tools help infrastructure teams **increase productivity** by automating various config tasks (e.g., server setup) through scripts that can be *reused* and *versioned*.

💀 As with any piece of code, IaC scripts are also prone to defects such as **security vulnerabilities**.

199K vulnerable
IaC templates

paloalto®
NETWORKS

Motivation > Security Vulnerabilities in Infrastructure-as-Code Scripts

🚀 Software configuration management and deployment tools like **Puppet** became popular amongst software development warehouses.

👷 These tools help infrastructure teams **increase productivity** by automating various config tasks (e.g., server setup) through scripts that can be *reused* and *versioned*.

💀 As with any piece of code, IaC scripts are also prone to defects such as **security vulnerabilities**.

199K vulnerable
IaC templates

paloalto NETWORKS

67k potential
Security Smells in IaC

Rahman et al. [ICSE'19; TSE'20]

Oh gosh! 🫣

Assessment > 12 types of weaknesses

Weakness	Name	Example
CWE-798	Use of Hard Coded Credentials	$username = “mariadb”
CWE-269	Use of Hard Coded Password	$password = “!TQ23Rg”
CWE-321	Use of Hard Coded Cryptographic Key	$key = “A67ANBD7”
CWE-319	Use of HTTP without TLS	$req = “http://www.domain.org/secret”
CWE-546	Suspicious Comment	#https://bugs.debian.org/cgi-bin/bugreport.cgi?bug=538392
CWE-326	Use of Weak Crypto Algorithms	password => md5($debian_password)
CWE-284	Invalid IP address Binding	$bind_host = “0.0.0.0”
CWE-258	Empty Password in Configuration File	$rabbitmq_pwd = “”
CWE-250	Admin by default	$user = “admin”
CWE-521	Weak Password	pwd => “12345”
CWE-1007	Homoglyphs Detection (typo-squatting attacks)	$source = "http://deb.debiаn.org/debiаn"
CWE-829	Malicious Dependencies	$postgresql_version = 8.4

Motivation > Automated Security Weakness Detection in Puppet

🎯 Focus on **Puppet**

⚙️ Lightweight Solution Available (called **SLIC**) [Rahman et al., ICSE’19]

99% of precision and accuracy in an oracle dataset

💀 **SLIC** detects 7 types of weaknesses.

1st question: How does **SLIC** perform on a new dataset?

Study 1 > Validation with Students

Research Team

1419 GitHub repositories (~34k Puppet Scripts).

Found **31990 security warnings** involving 9144 of Puppet scripts.

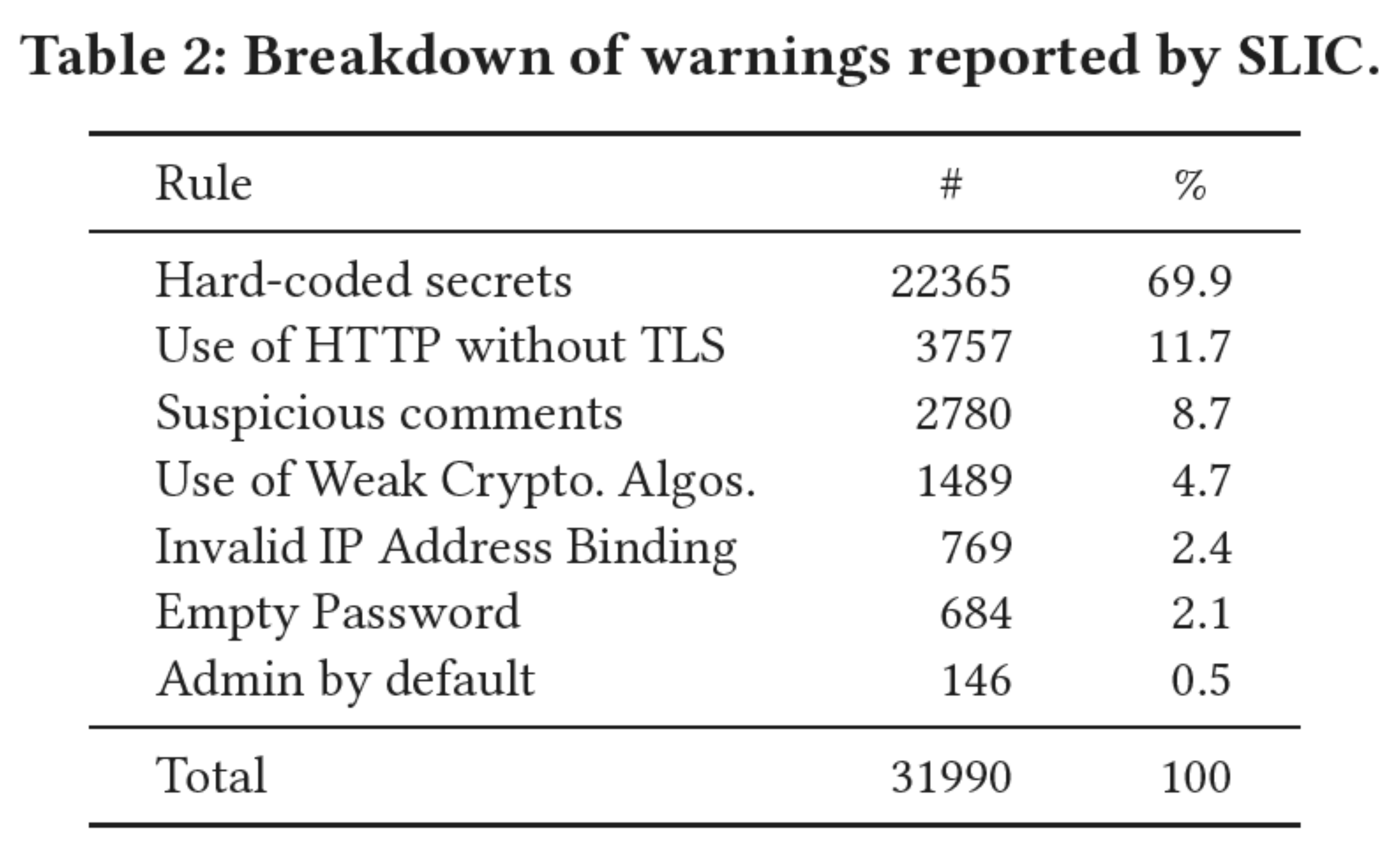


Table 2: Breakdown of warnings reported by SLIC.

Rule	#	%
Hard-coded secrets	22365	69.9
Use of HTTP without TLS	3757	11.7
Suspicious comments	2780	8.7
Use of Weak Crypto. Algos.	1489	4.7
Invalid IP Address Binding	769	2.4
Empty Password	684	2.1
Admin by default	146	0.5
Total	31990	100

Study 1 > Validation with Students

2 authors validated a total of 502 warnings.

Two samples: **proportional** (*stratified*) and **uniform** (*stratified*).

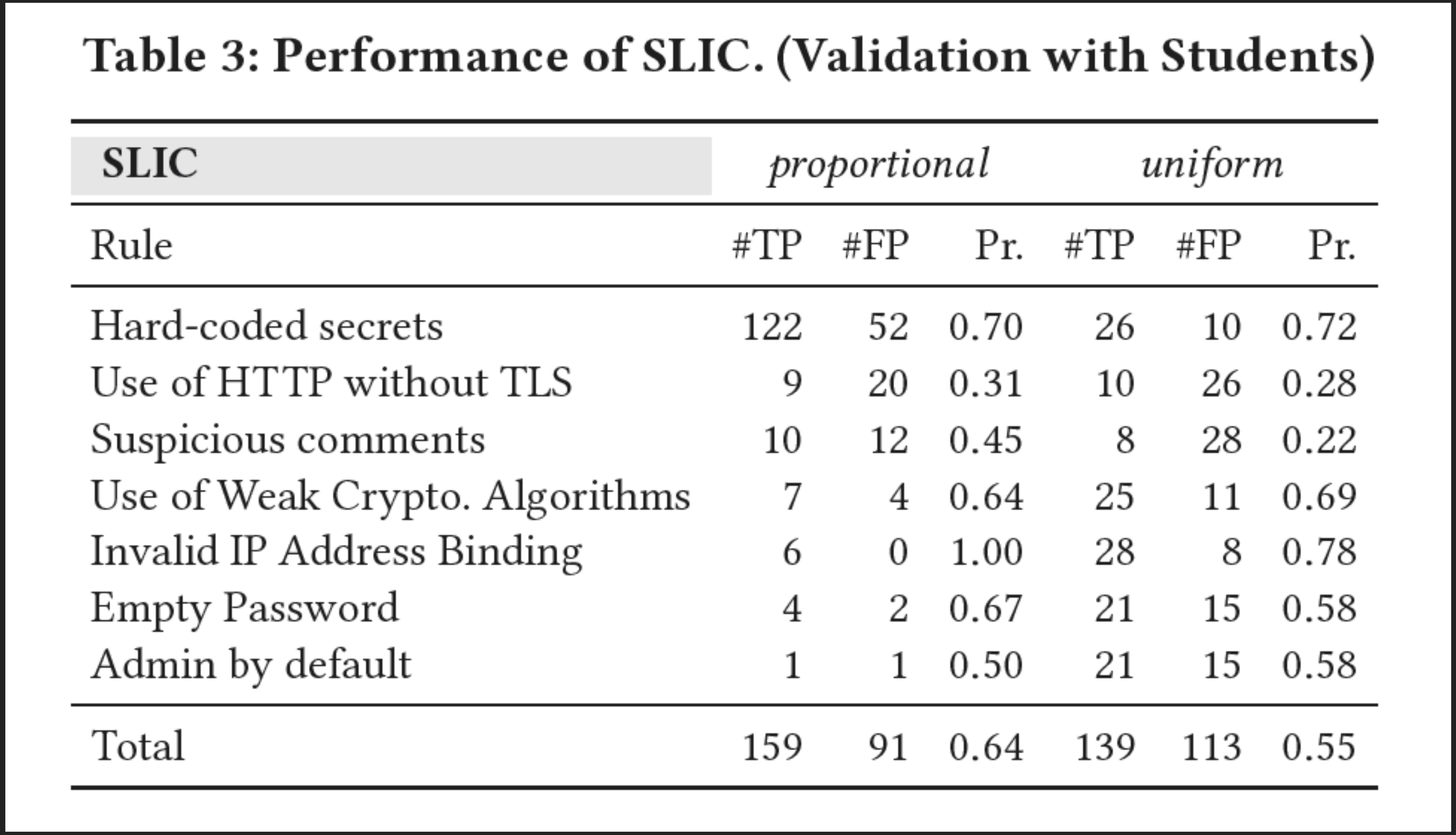


Table 3: Performance of SLIC. (Validation with Students)

SLIC	*proportional*			*uniform*		
Rule	#TP	#FP	Pr.	#TP	#FP	Pr.
Hard-coded secrets	122	52	0.70	26	10	0.72
Use of HTTP without TLS	9	20	0.31	10	26	0.28
Suspicious comments	10	12	0.45	8	28	0.22
Use of Weak Crypto. Algorithms	7	4	0.64	25	11	0.69
Invalid IP Address Binding	6	0	1.00	28	8	0.78
Empty Password	4	2	0.67	21	15	0.58
Admin by default	1	1	0.50	21	15	0.58
Total	159	91	0.64	139	113	0.55

Research Team

Precision decreased from 99% to 64%. 🙁

@sofiaoreis

Study 1 > Validation with Students

2 authors validated a total of 502 warnings.

Two samples: **proportional** (*stratified*) and **uniform** (*stratified*).

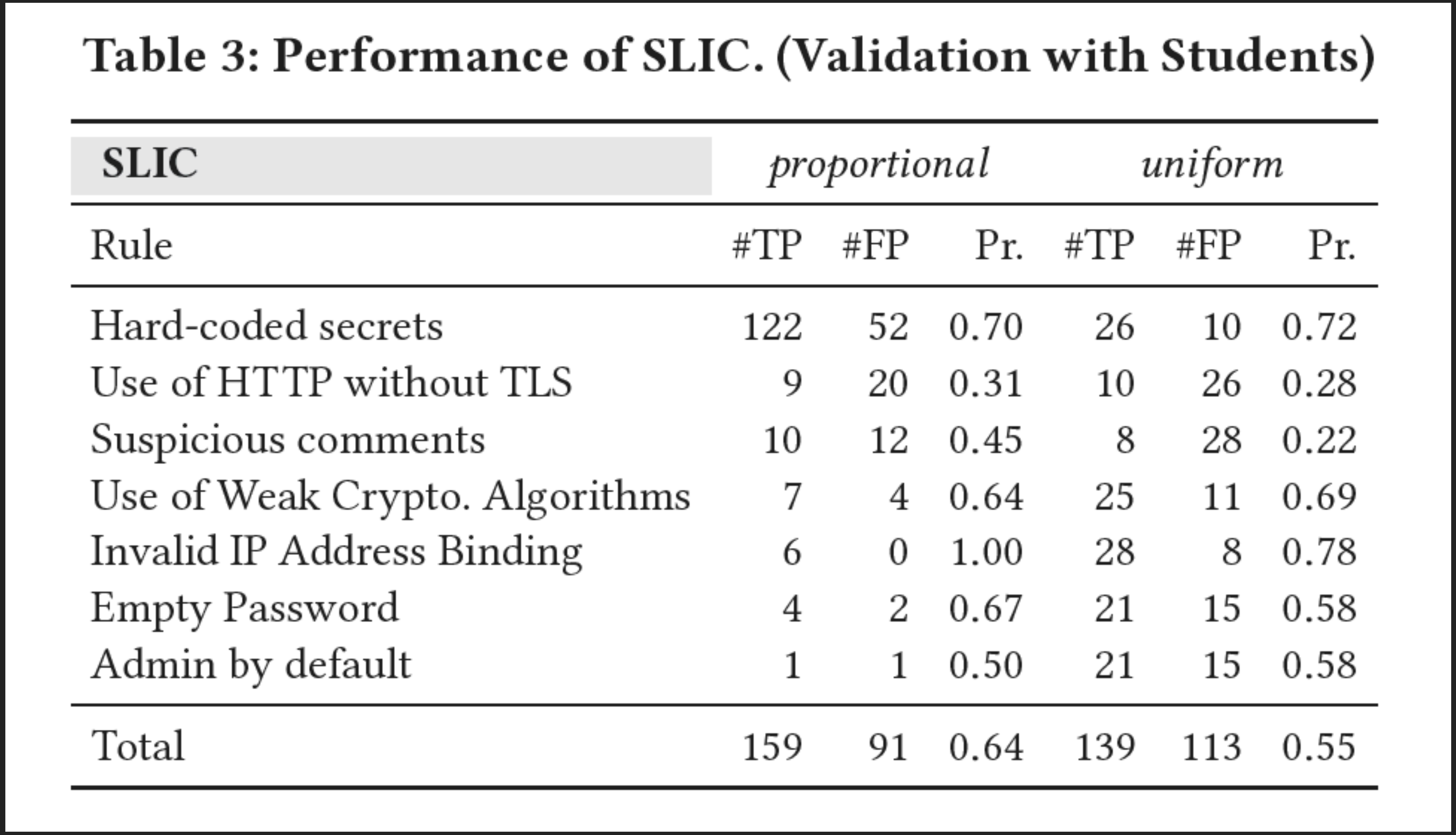


Table 3: Performance of SLIC. (Validation with Students)

SLIC	*proportional*			*uniform*		
Rule	#TP	#FP	Pr.	#TP	#FP	Pr.
Hard-coded secrets	122	52	0.70	26	10	0.72
Use of HTTP without TLS	9	20	0.31	10	26	0.28
Suspicious comments	10	12	0.45	8	28	0.22
Use of Weak Crypto. Algorithms	7	4	0.64	25	11	0.69
Invalid IP Address Binding	6	0	1.00	28	8	0.78
Empty Password	4	2	0.67	21	15	0.58
Admin by default	1	1	0.50	21	15	0.58
Total	159	91	0.64	139	113	0.55

Research Team

Precision decreased from 99% to 64%. 🙁

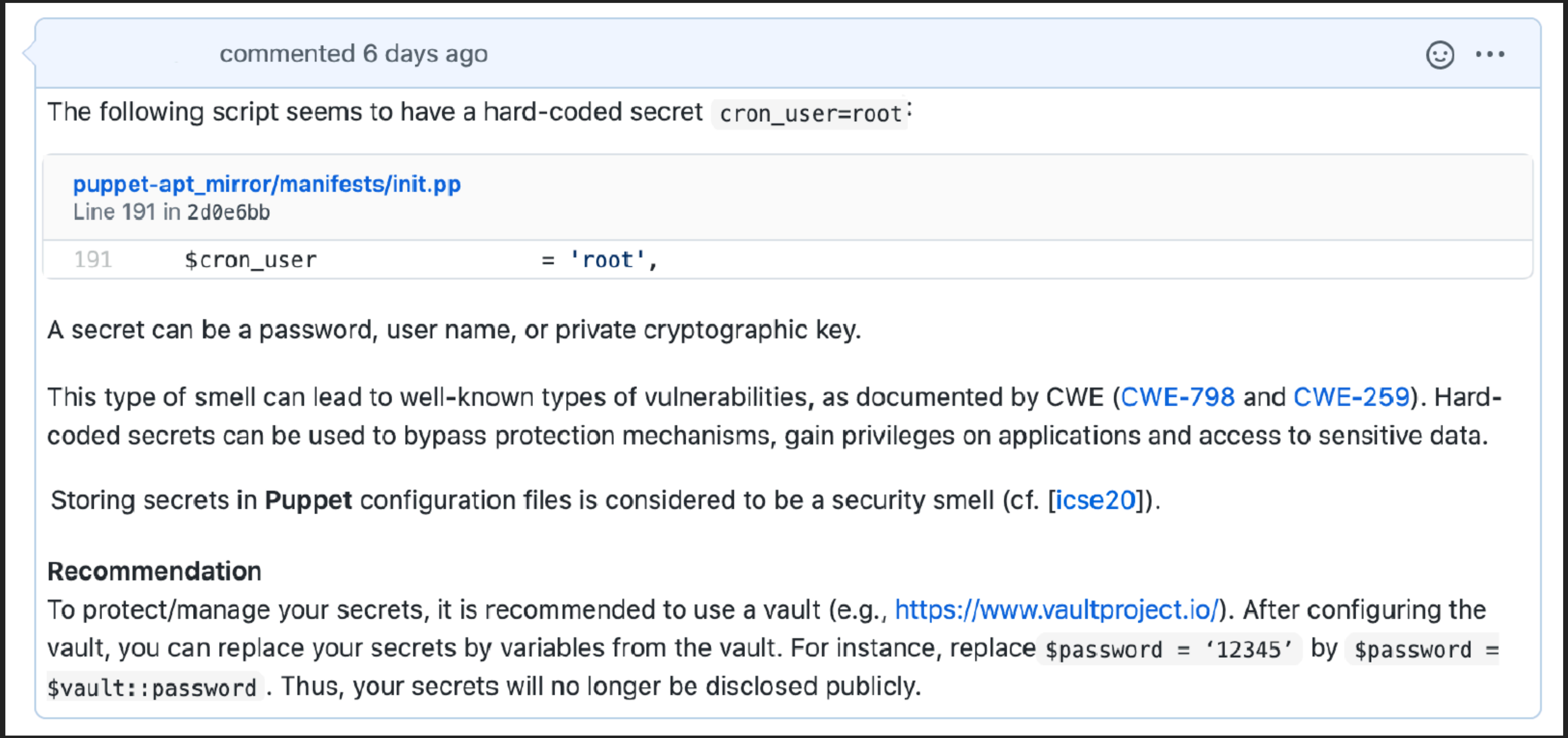
Maybe we don't have enough context?! 🤔

@sofiaoreis

Study 2 > Validation with OSS Maintainers

Issued alerts to projects maintainers involved in the slack puppet community.

Issues included the code sample, issues description and links to more information.



commented 6 days ago

The following script seems to have a hard-coded secret `cron_user=root`:

puppet-apt_mirror/manifests/init.pp
Line 191 in 2d0e6bb

```
191     $cron_user                        = 'root',
```

A secret can be a password, user name, or private cryptographic key.

This type of smell can lead to well-known types of vulnerabilities, as documented by CWE (CWE-798 and CWE-259). Hard-coded secrets can be used to bypass protection mechanisms, gain privileges on applications and access to sensitive data.

Storing secrets in **Puppet** configuration files is considered to be a security smell (cf. [icse20]).

Recommendation
To protect/manage your secrets, it is recommended to use a vault (e.g., https://www.vaultproject.io/). After configuring the vault, you can replace your secrets by variables from the vault. For instance, replace `$password = ‘12345’` by `$password = $vault::password`. Thus, your secrets will no longer be disclosed publicly.

Location

Description

Assessment

Actionable

@sofiaoreis

Study 2 > Validation with OSS Maintainers

Got 51 answers to the 228 issues submitted; but only 33 were **clearly validated**.

❌ “N/A”;“:thumbs_down”

✅ “These todos’s shouldn’t be there, I agree…”

Study 2 > Validation with OSS Maintainers

Got 51 answers to the 228 issues submitted; but only 33 were **clearly validated**.

❌ “N/A”;“:thumbs_down”

✅ “These todos’s shouldn’t be there, I agree…”

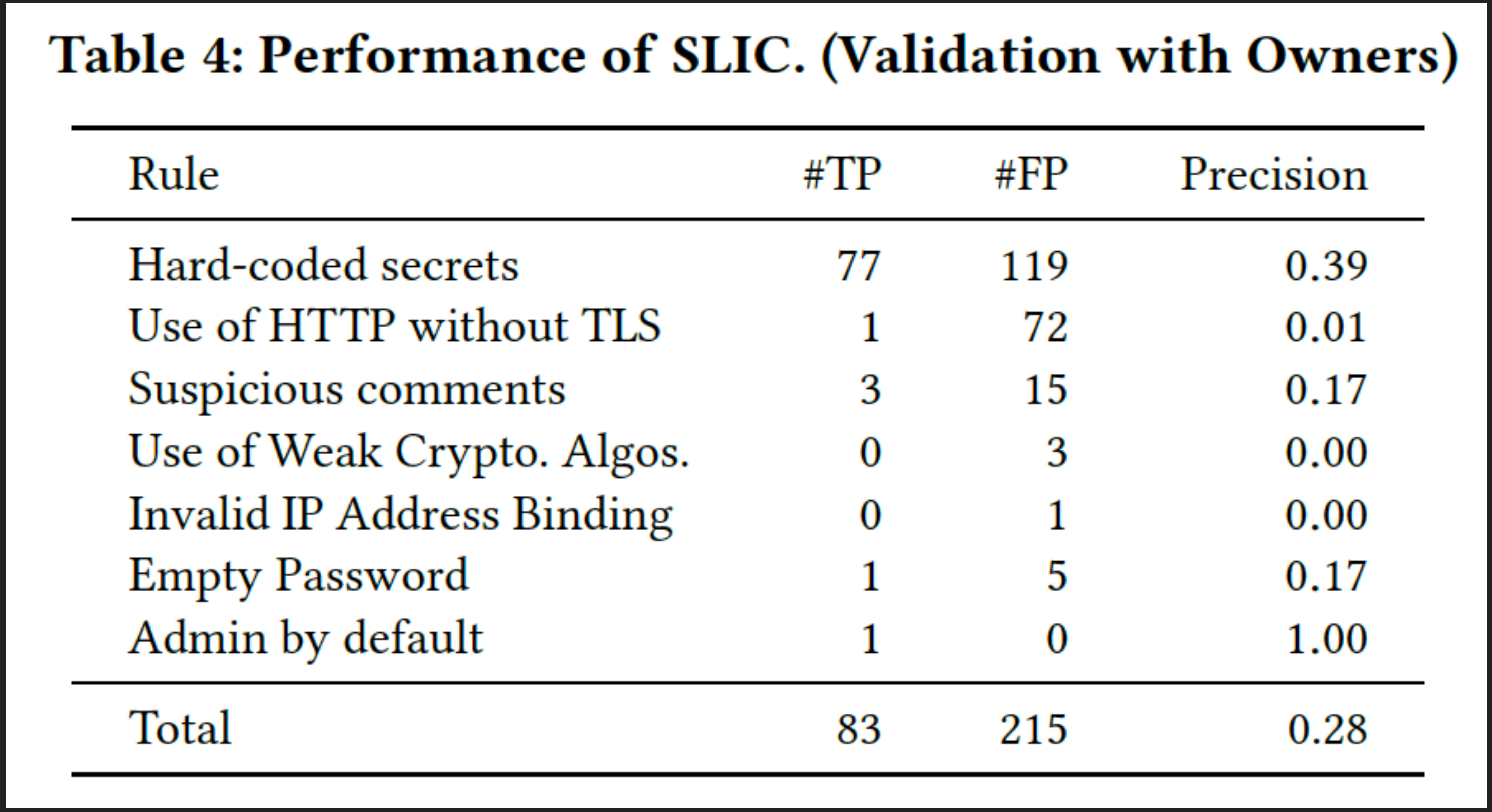


Table 4: Performance of SLIC. (Validation with Owners)

Rule	#TP	#FP	Precision
Hard-coded secrets	77	119	0.39
Use of HTTP without TLS	1	72	0.01
Suspicious comments	3	15	0.17
Use of Weak Crypto. Algos.	0	3	0.00
Invalid IP Address Binding	0	1	0.00
Empty Password	1	5	0.17
Admin by default	1	0	1.00
Total	83	215	0.28

Ups! Precision is even worse.

Precision decreased to 28%, 😣

1st question: How does **SLIC** perform on a new dataset?

🙁 ***Not great!***

1st question: How does **SLIC** perform on a new dataset?

🙁 *Not great!*

Problem > Puppet IaC Security Linters are not reliable yet!

🧐 Precision is even lower when evaluated by maintainers—developers with more knowledge and context of the applications.

1st question: How does **SLIC** perform on a new dataset? 🙁 *Not great!*

Problem > Puppet IaC Security Linters are not reliable yet!

🧐 Precision is even lower when evaluated by maintainers—developers with more knowledge and context of the applications.

🎯 During study 1 and study 2, we were able to list several problems in the tool weakness- and analysis-related.

```
if has_key($userdata, ‘env’)
```

SLIC found a hard coded secret in this logical condition 🙅‍♂️

1st question: How does **SLIC** perform on a new dataset?

Not great!

Problem > Puppet IaC Security Linters are not reliable yet!

Precision is even lower when evaluated by maintainers—developers with more knowledge and context of the applications.

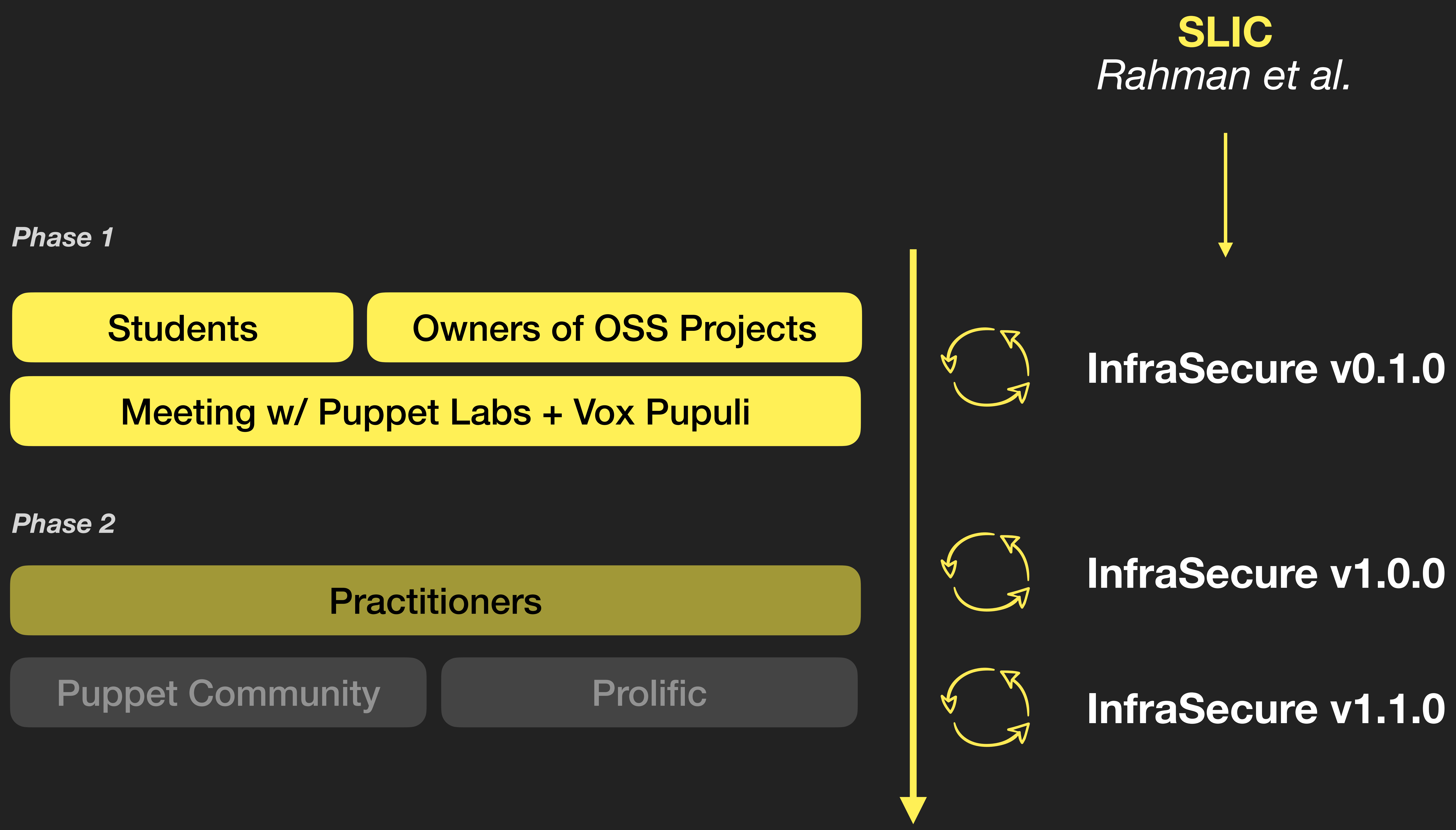
During study 1 and study 2, we were able to list several problems in the tool weakness- and analysis-related.

`if has_key($userdata, ‘env’)` SLIC found a hard coded secret in this logical condition

Static analysis tools can be iteratively improved and extended by incorporating feedback from the developer community [Sadowski, ACM Commun.’18]

@sofiaoreis

Methodology > Improve the linter with Practitioners' Feedback



SLIC

Rahman et al.

Phase 1

Students

Owners of OSS Projects

Meeting w/ Puppet Labs + Vox Pupuli

Phase 2

Practitioners

Puppet Community

Prolific

InfraSecure v0.1.0

InfraSecure v1.0.0

InfraSecure v1.1.0

InfraSecure v0.1.0 > Design Choices

Variable/Attribute Assignments (VASS)

Reduce the number of incorrect predictions

isVarAssign(token) ∧ isAtrAssign(token)

✗

```
if has_key($userdata, ‘env’)
```

SLIC found a hard coded secret in this logical condition

Reasoning about the token value (TOKVAL)

Some of the rules did not reason about *token.value*

✗

```
aws_admin_username = downcase($::operatingsystem)
```

No secret is stored

Credentials that are not consider secrets by the community

isUserDefault(token.value)

[Maintainer] *“The names of these UNIX accounts are not considered to be secret. They are published openly as part of the PE documentation: https://puppet.com/docs/pe/ 2019.8/what_gets_installed_and_where.html#user_and_group_accounts_installed”*

InfraSecure v0.1.0 > Design Choices

Variable/Attribute Assignments (VASS)

Reduce the number of incorrect predictions

isVarAssign(token) ∧ isAtrAssign(token)

❌

```
if has_key($userdata, ‘env’)
```

SLIC found a hard coded secret in this logical condition

Reasoning about the token value (TOKVAL)

Some of the rules did not reason about *token.value*

❌

```
aws_admin_username = downcase($::operatingsystem)
```

No secret is stored

Non-valid values for secrets

InvalidSecret(token.value)

[Maintainer] *“This are default users and default as found in every installed fpm package. there is most of the time a wwwrun or a www-data user depending on the system.”*

InfraSecure v0.1.0 > Rule Improvements

Usage of Weak Crypto Algorithms

Search for in calls to functions

isFunctionCall()

❌

```
md5checksum = ’07bd73571b7028b73fc8ed19bc85226d’
```

Not a call to the md5() function

Invalid IP address binding

IPs follow dot-decimal notation

isInvalidIPBind(token.value)

❌

```
description => ’Open up postgresql for access to sensu from 0.0.0.0/0’
```

STRING != IP

Check our paper for more! **Section 4.3**

InfraSecure v0.1.0 > Design Choices

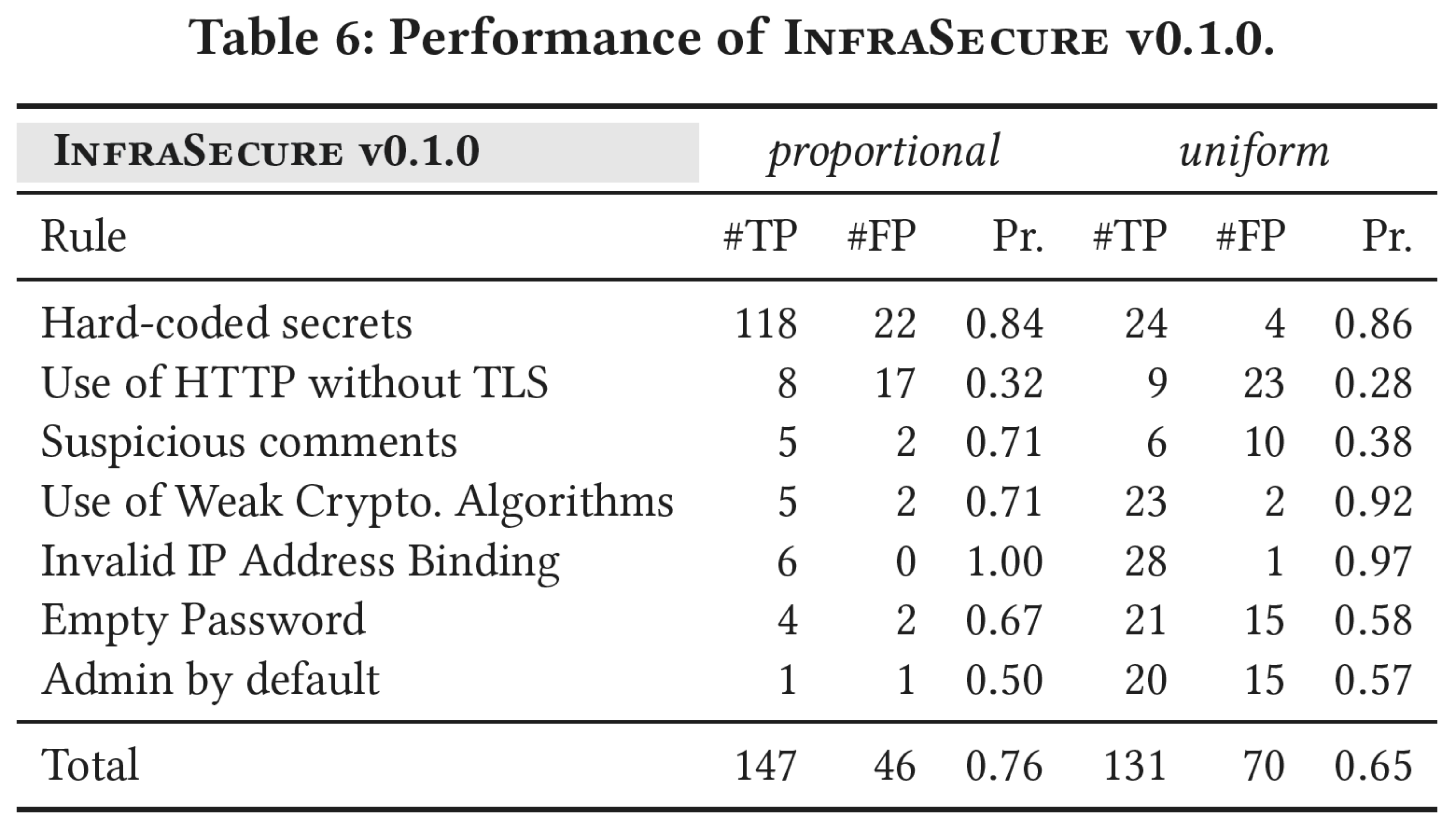


Table 6: Performance of InfraSecure v0.1.0.

InfraSecure v0.1.0	proportional			uniform		
Rule	#TP	#FP	Pr.	#TP	#FP	Pr.
Hard-coded secrets	118	22	0.84	24	4	0.86
Use of HTTP without TLS	8	17	0.32	9	23	0.28
Suspicious comments	5	2	0.71	6	10	0.38
Use of Weak Crypto. Algorithms	5	2	0.71	23	2	0.92
Invalid IP Address Binding	6	0	1.00	28	1	0.97
Empty Password	4	2	0.67	21	15	0.58
Admin by default	1	1	0.50	20	15	0.57
Total	147	46	0.76	131	70	0.65

🥳

Precision increased!

Can we improve even more?

Let's ask ***practitioners****!*

InfraSecure v0.1.0 > Design Choices

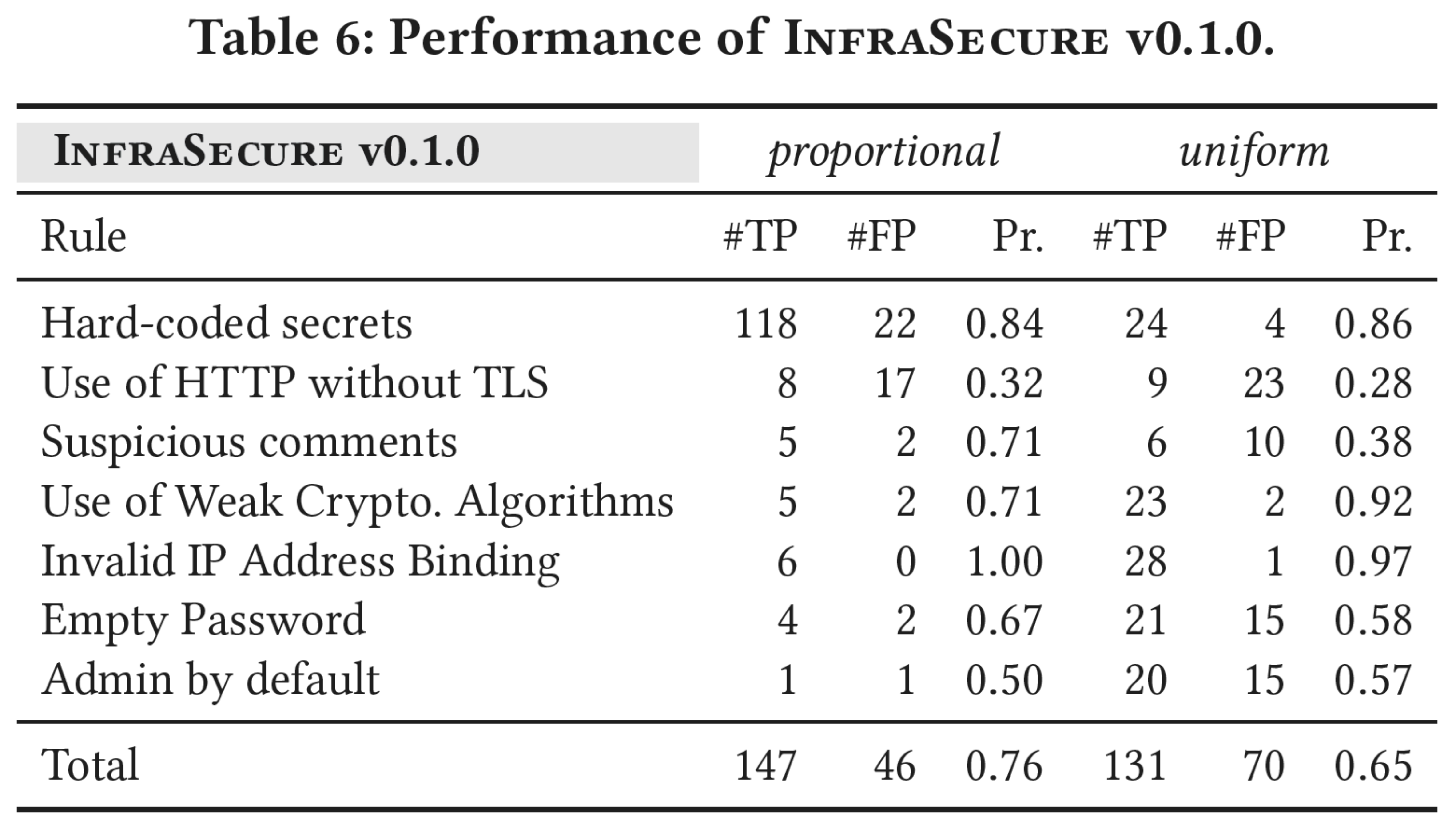


Table 6: Performance of InfraSecure v0.1.0.

InfraSecure v0.1.0	*proportional*			*uniform*		
Rule	#TP	#FP	Pr.	#TP	#FP	Pr.
Hard-coded secrets	118	22	0.84	24	4	0.86
Use of HTTP without TLS	8	17	0.32	9	23	0.28
Suspicious comments	5	2	0.71	6	10	0.38
Use of Weak Crypto. Algorithms	5	2	0.71	23	2	0.92
Invalid IP Address Binding	6	0	1.00	28	1	0.97
Empty Password	4	2	0.67	21	15	0.58
Admin by default	1	1	0.50	20	15	0.57
Total	147	46	0.76	131	70	0.65

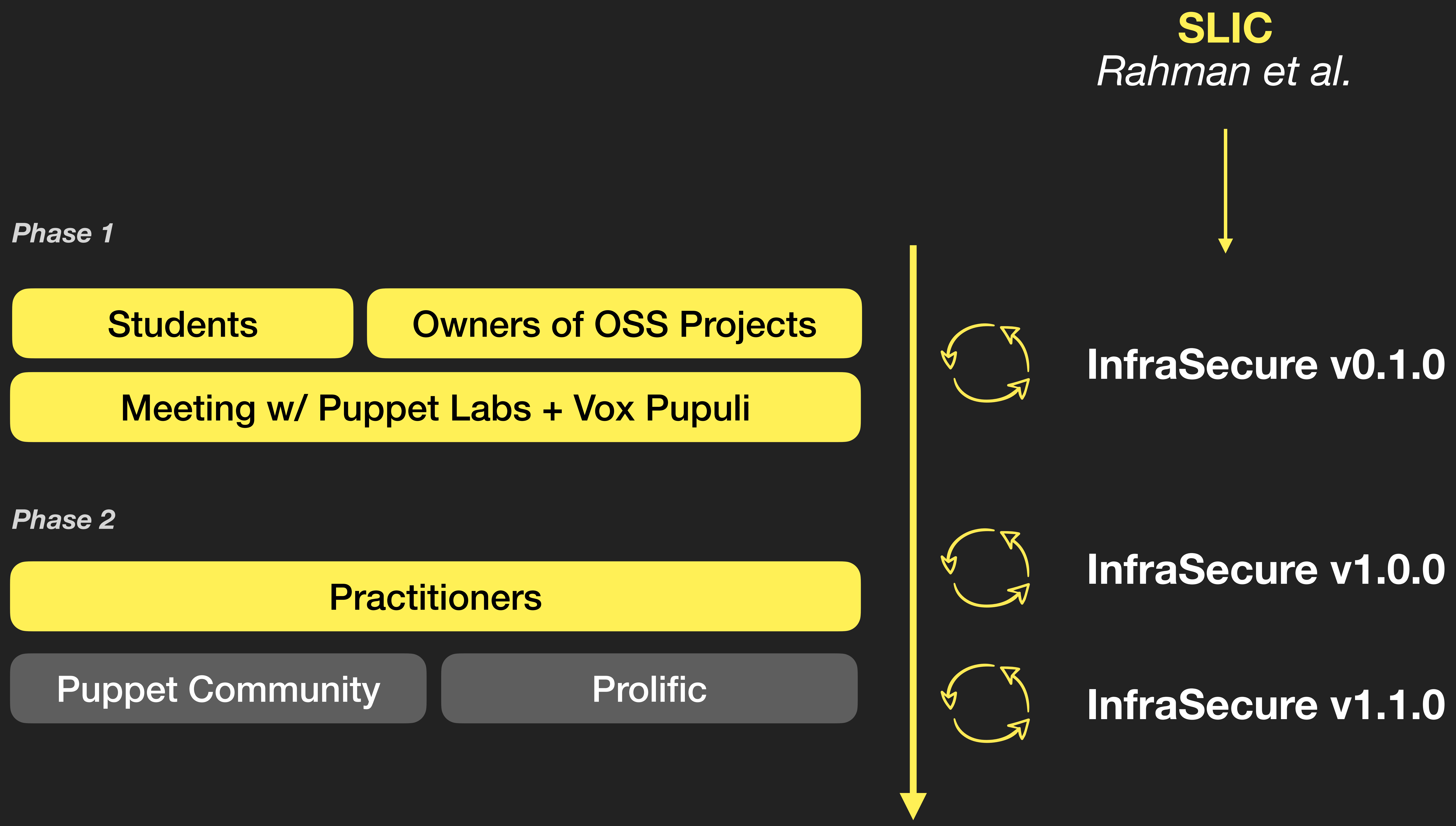
🥳

Precision increased!

Can we improve even more?

Let's ask ***practitioners****!*

Methodology > Improve the linter with Practitioners' Feedback



SLIC

Rahman et al.

Phase 1

Students

Owners of OSS Projects

Meeting w/ Puppet Labs + Vox Pupuli

Phase 2

Practitioners

Puppet Community

Prolific

InfraSecure v0.1.0

InfraSecure v1.0.0

InfraSecure v1.1.0

Study 3 > Validation with Practitioners

Validate InfraSecure v0.1.0 alerts

Experiment shared with the Puppet communities on Slack (puppet.community.slack.com) and Reddit (r/puppet).

14 participants

Prolific

117 participants

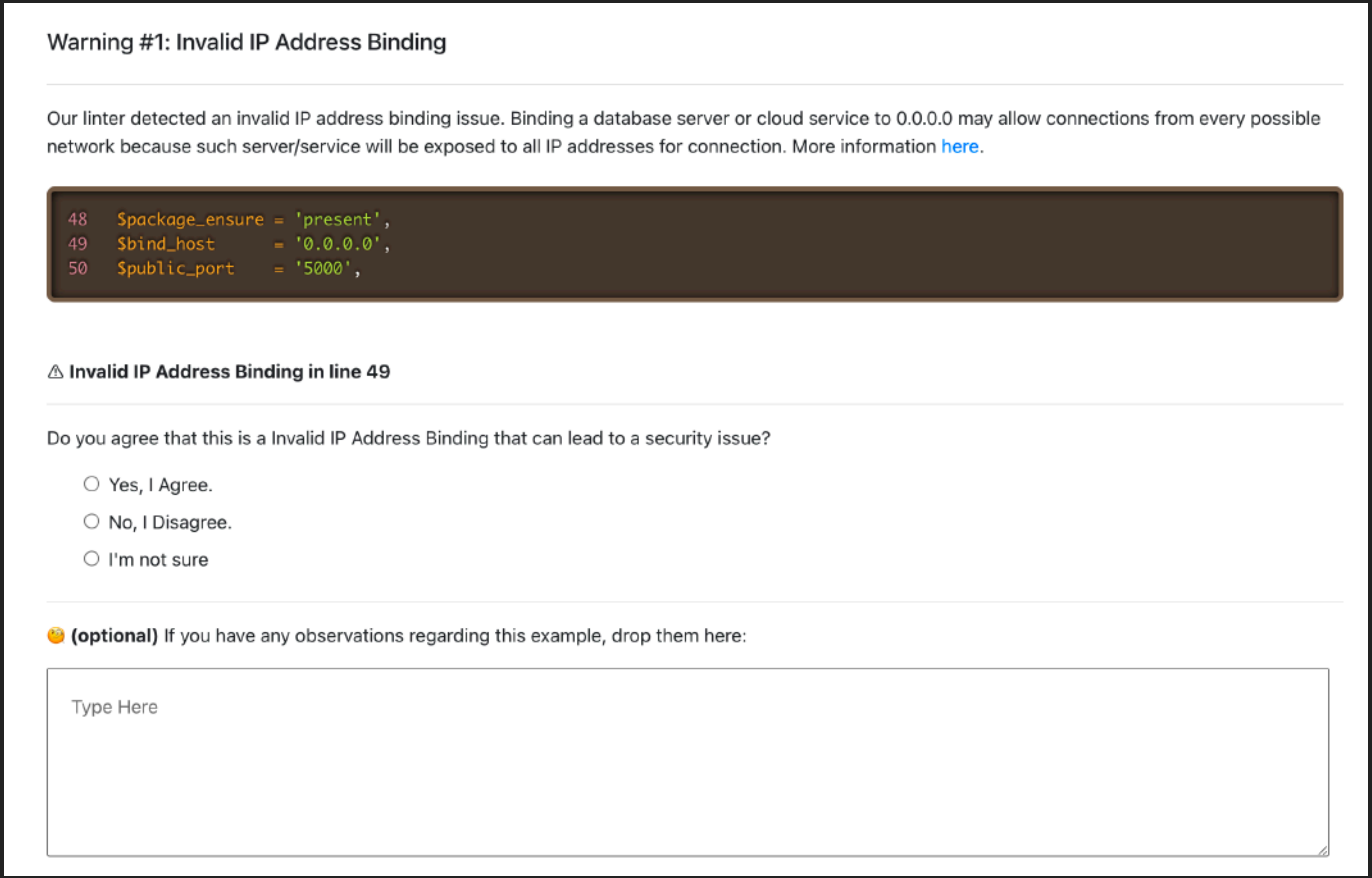
Validation of ⚠

339 warnings

Pre-screening: Specific Industries (e.g., Computer and Electronics), experience with configuration management tools, security and infrastructure as a service; and, a quizz of three programming questions about different puppet configurations. (**check the replication package**)

Study 3 > Validation with Practitioners

Example of the form for alert validation



Warning #1: Invalid IP Address Binding

Our linter detected an invalid IP address binding issue. Binding a database server or cloud service to 0.0.0.0 may allow connections from every possible network because such server/service will be exposed to all IP addresses for connection. More information here.

```
48   $package_ensure = 'present',
49   $bind_host      = '0.0.0.0',
50   $public_port    = '5000',
```

⚠ **Invalid IP Address Binding in line 49**

Do you agree that this is a Invalid IP Address Binding that can lead to a security issue?

○ Yes, I Agree.

○ No, I Disagree.

○ I'm not sure

😳 **(optional)** If you have any observations regarding this example, drop them here:

Type Here

@sofiaoreis

InfraSecure v1.0.0 > More feedback and improvements

Use of HTTP without TLS is fine sometimes

Customizable rule (whitelist with credible sources)

inWhitelist(token.value)

❌ `Apturl => “http://deb.debian.org/debian`

SLIC reports every single occurence of http:// as unsafe.

[Practitioner] *“I think it is fine if localhost is used. Otherwise TLS should be mandatory. All the big financial organizations will not use this check because they cannot create internal certs or use letsencrypt.”*

[Practitioner] *“By default, it’s unsafe to not use HTTPS. But for internal testing/development it is acceptable to me to not use HTTPS all the time.”*

InfraSecure v1.1.0 > New Patterns (Extension)

Weak Password	**isStrongPwd()**	Uses PHP algorithm developed by Thomas Hruska.
Homograph Attacks *supply chain attack*	**hasCyrillic()**	Social engineering attack that purposely uses misspelt domains for malicious purposes.

Malicious Dependencies
supply chain attack
	isResource()
	isMalicious()
		Our database integrates malicious versions of software for 33 different packages used by the Puppet community (e.g., rabbitmq, apt, cassandra, postgresql, etc).

CWE-521	Weak Password	pwd => “12345”
CWE-1007	Homoglyphs Detection (typo-squatting attacks)	$source = "http://deb.debi*a*n.org/debi*a*n"
CWE-829	Malicious Dependencies	$postgresql_version = 8.4

Study 3 > Validation with Practitioners

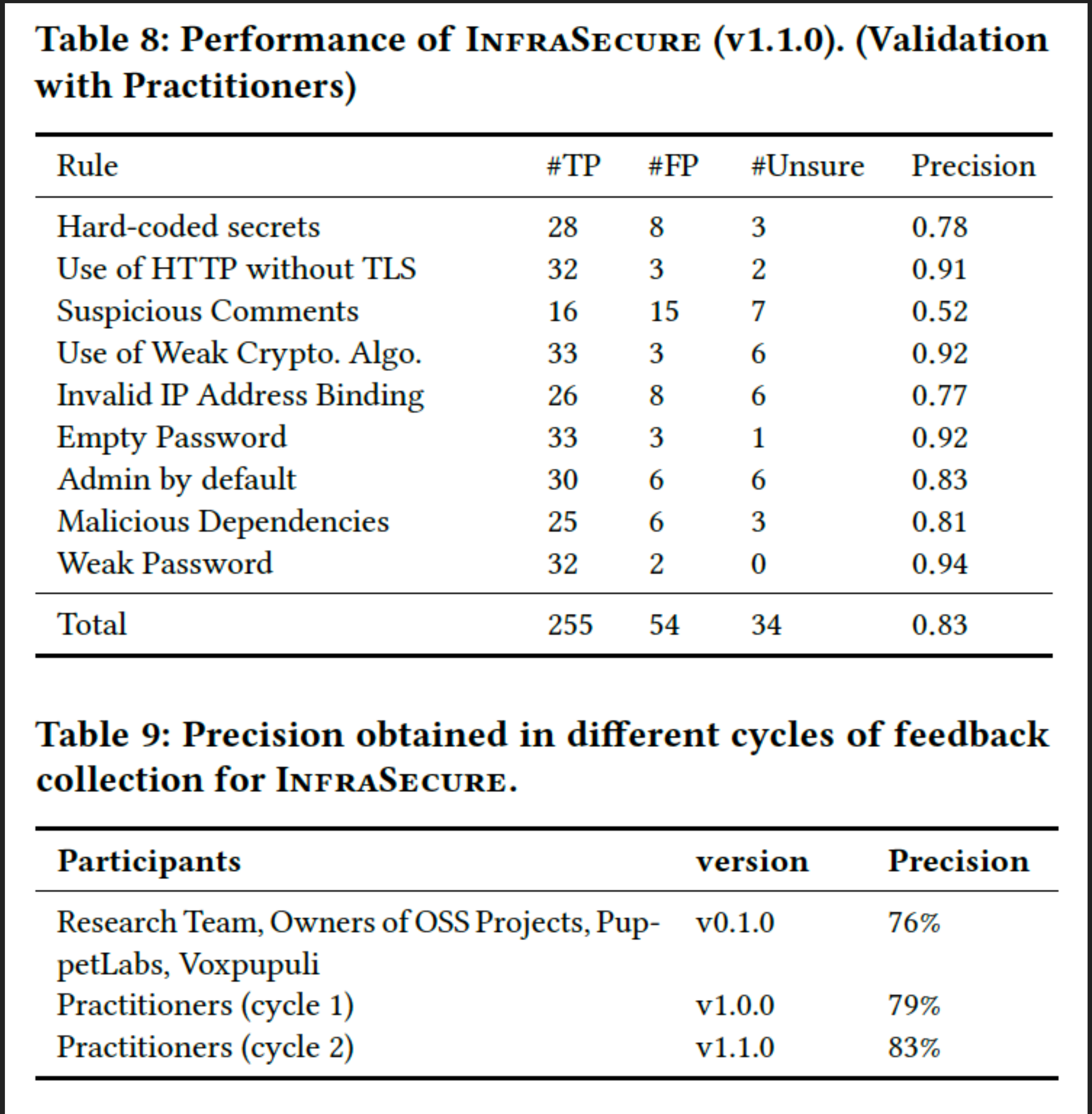


Table 8: Performance of InfraSecure (v1.1.0). (Validation with Practitioners)

Rule	#TP	#FP	#Unsure	Precision
Hard-coded secrets	28	8	3	0.78
Use of HTTP without TLS	32	3	2	0.91
Suspicious Comments	16	15	7	0.52
Use of Weak Crypto. Algo.	33	3	6	0.92
Invalid IP Address Binding	26	8	6	0.77
Empty Password	33	3	1	0.92
Admin by default	30	6	6	0.83
Malicious Dependencies	25	6	3	0.81
Weak Password	32	2	0	0.94
Total	255	54	34	0.83

Table 9: Precision obtained in different cycles of feedback collection for InfraSecure.

Participants	**version**	**Precision**
Research Team, Owners of OSS Projects, PuppetLabs, Voxpupuli	v0.1.0	76%
Practitioners (cycle 1)	v1.0.0	79%
Practitioners (cycle 2)	v1.1.0	83%

Precision increased
between iterations
(28% -> 76% -> 79%
-> 83%)

More Anti-Patterns
Malicious dependencies, Homograph Attacks and Weak Passwords

More Customisation
Whitelist

@sofiaoreis

🤩 Rules

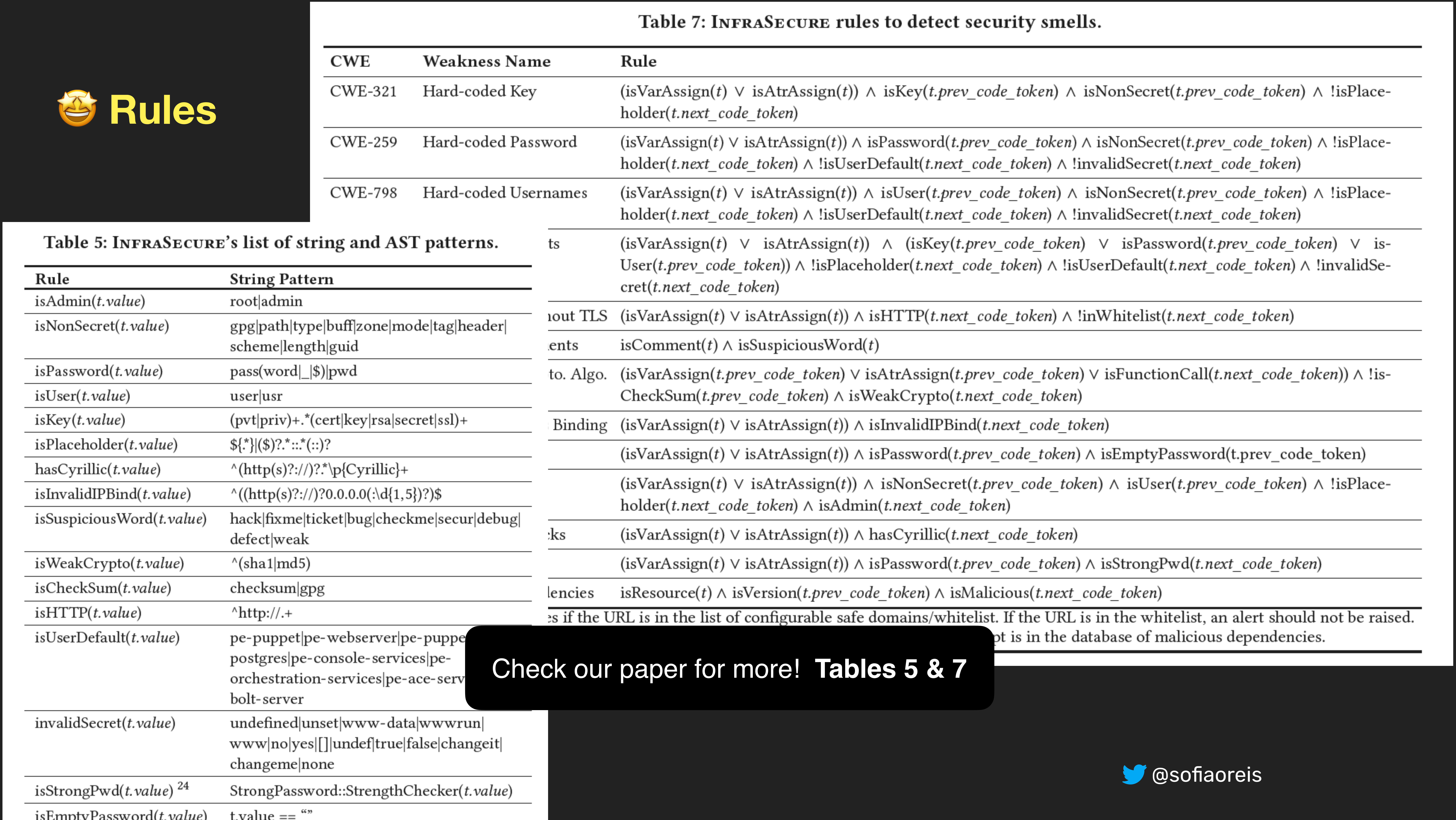


Table 7: InfraSecure rules to detect security smells.

CWE	Weakness Name	Rule
CWE-321	Hard-coded Key	(isVarAssign(*t*) ∨ isAtrAssign(*t*)) ∧ isKey(*t.prev_code_token*) ∧ isNonSecret(*t.prev_code_token*) ∧ !isPlaceholder(*t.next_code_token*)
CWE-259	Hard-coded Password	(isVarAssign(*t*) ∨ isAtrAssign(*t*)) ∧ isPassword(*t.prev_code_token*) ∧ isNonSecret(*t.prev_code_token*) ∧ !isPlaceholder(*t.next_code_token*) ∧ !isUserDefault(*t.next_code_token*) ∧ !invalidSecret(*t.next_code_token*)
CWE-798	Hard-coded Usernames	(isVarAssign(*t*) ∨ isAtrAssign(*t*)) ∧ isUser(*t.prev_code_token*) ∧ isNonSecret(*t.prev_code_token*) ∧ !isPlaceholder(*t.next_code_token*) ∧ !isUserDefault(*t.next_code_token*) ∧ !invalidSecret(*t.next_code_token*)
	ts	(isVarAssign(*t*) ∨ isAtrAssign(*t*)) ∧ (isKey(*t.prev_code_token*) ∨ isPassword(*t.prev_code_token*) ∨ isUser(*t.prev_code_token*)) ∧ !isPlaceholder(*t.next_code_token*) ∧ !isUserDefault(*t.next_code_token*) ∧ !invalidSecret(*t.next_code_token*)
	out TLS	(isVarAssign(*t*) ∨ isAtrAssign(*t*)) ∧ isHTTP(*t.next_code_token*) ∧ !inWhitelist(*t.next_code_token*)
	ents	isComment(*t*) ∧ isSuspiciousWord(*t*)
	to. Algo.	(isVarAssign(*t.prev_code_token*) ∨ isAtrAssign(*t.prev_code_token*) ∨ isFunctionCall(*t.next_code_token*)) ∧ !isCheckSum(*t.prev_code_token*) ∧ isWeakCrypto(*t.next_code_token*)
	Binding	(isVarAssign(*t*) ∨ isAtrAssign(*t*)) ∧ isInvalidIPBind(*t.next_code_token*)
		(isVarAssign(*t*) ∨ isAtrAssign(*t*)) ∧ isPassword(*t.prev_code_token*) ∧ isEmptyPassword(t.prev_code_token)
		(isVarAssign(*t*) ∨ isAtrAssign(*t*)) ∧ isNonSecret(*t.prev_code_token*) ∧ isUser(*t.prev_code_token*) ∧ !isPlaceholder(*t.next_code_token*) ∧ isAdmin(*t.next_code_token*)
	ks	(isVarAssign(*t*) ∨ isAtrAssign(*t*)) ∧ hasCyrillic(*t.next_code_token*)
		(isVarAssign(*t*) ∨ isAtrAssign(*t*)) ∧ isPassword(*t.prev_code_token*) ∧ isStrongPwd(*t.next_code_token*)
	lencies	isResource(*t*) ∧ isVersion(*t.prev_code_token*) ∧ isMalicious(*t.next_code_token*)

s if the URL is in the list of configurable safe domains/whitelist. If the URL is in the whitelist, an alert should not be raised.

pt is in the database of malicious dependencies.

Table 5: InfraSecure's list of string and AST patterns.

Rule	String Pattern
isAdmin(*t.value*)	root|admin
isNonSecret(*t.value*)	gpg|path|type|buff|zone|mode|tag|header|scheme|length|guid
isPassword(*t.value*)	pass(word|_|$)|pwd
isUser(*t.value*)	user|usr
isKey(*t.value*)	(pvt|priv)+.*(cert|key|rsa|secret|ssl)+
isPlaceholder(*t.value*)	${.*}\|($)?.*::.*(::)?
hasCyrillic(*t.value*)	^(http(s)?://)?.*\p{Cyrillic}+
isInvalidIPBind(*t.value*)	^((http(s)?://)?0.0.0.0(:\d{1,5})?)$
isSuspiciousWord(*t.value*)	hack|fixme|ticket|bug|checkme|secur|debug|defect|weak
isWeakCrypto(*t.value*)	^(sha1|md5)
isCheckSum(*t.value*)	checksum|gpg
isHTTP(*t.value*)	^http://.+
isUserDefault(*t.value*)	pe-puppet|pe-webserver|pe-puppe postgres|pe-console-services|pe-orchestration-services|pe-ace-serv bolt-server
invalidSecret(*t.value*)	undefined|unset|www-data|wwwrun|www|no|yes|[]|undef|true|false|changeit|changeme|none
isStrongPwd(*t.value*) [24]	StrongPassword::StrengthChecker(*t.value*)
isEmptyPassword(*t.value*)	t.value == “”

Check our paper for more! **Tables 5 & 7**

@sofiaoreis

Main Conclusions

👍 ***(1) It is feasible to tune security linters to produce acceptable precision.***

🦸 ***(2) Involving practitioners in discussions is an effective way to guide the improvement of those linters.***

💯 *In the process of feedback collection, tool owners can learn more on how to extend the **anti-patterns coverage** and how to better **customise** the tool!*