INNCYBER Innovation HUB

Digital transformation, Cyber & IoT

Java Typestate Checker

Members

- Lorenzo Bacchiani
- Mario Bravetti
- Marco Giunti
- João Mota
- António Ravara

ALMA MATER STUDIORUM
UNIVERSITÀ DI BOLOGNA

NOVA SCHOOL OF
SCIENCE & TECHNOLOGY

Context: Bugs are cybersecurity risks

CWE-841: Improper Enforcement of Behavioral Workflow

Weakness ID: 841
Abstraction: Base
Structure: Simple

View customized information: Conceptual | Operational | Mapping Friendly | **Complete** | Custom

Description

The product supports a session in which more than one behavior must be performed by an actor, but it does not properly ensure that the actor performs the behaviors in the required sequence.

Extended Description

By performing actions in an unexpected order, or by omitting steps, an attacker could manipulate the business logic of the product or cause it to enter an invalid state. In some cases, this can also expose resultant weaknesses.

For example, a file-sharing protocol might require that an actor perform separate steps to provide a username, then a password, before being able to transfer files. If the file-sharing server accepts a password command followed by a transfer command, without any username being provided, the product might still perform the transfer.

Note that this is different than CWE-696, which focuses on when the product performs actions in the wrong sequence; this entry is closely related, but it is focused on ensuring that the actor performs actions in the correct sequence.

Workflow-related behaviors include:

- Steps are performed in the expected order

Context: Bugs are cybersecurity risks

INNCYBER Innovation HUB
Digital transformation, Cyber & IoT

Common Weakness Enumeration
A Community-Developed List of Software & Hardware Weakness Types

Top 25 | Top HW CWE | New to CWE? Start here!

CWE-755: Improper Handling of Exceptional Conditions

Weakness ID: 755
Abstraction: Class
Structure: Simple

View customized information: Conceptual | Operational | Mapping Friendly | **Complete** | Custom

Description

The product does not handle or incorrectly handles an exceptional condition.

Relationships

Relevant to the view "Research Concepts" (CWE-1000)

Nature	Type	ID	Name
ChildOf	P	703	Improper Check or Handling of Exceptional Conditions
ParentOf	B	209	Generation of Error Message Containing Sensitive Information
ParentOf	B	248	Uncaught Exception
ParentOf	B	274	Improper Handling of Insufficient Privileges
ParentOf	B	280	Improper Handling of Insufficient Permissions or Privileges
ParentOf	B	390	Detection of Error Condition Without Action
ParentOf	B	392	Missing Report of Error Condition
ParentOf	B	395	Use of NullPointerException Catch to Detect NULL Pointer Dereference
ParentOf	B	396	Declaration of Catch for Generic Exception

Context: Bugs are cybersecurity risks

Common Weakness Enumeration
A Community-Developed List of Software & Hardware Weakness Types

Top 25 | Top HW CWE | New to CWE? Start here!

CWE CATEGORY: Pointer Issues

Category ID: 465

Summary

Weaknesses in this category are related to improper handling of pointers.

Membership

Nature	Type	ID	Name
MemberOf	V	699	Software Development
HasMember	B	466	Return of Pointer Value Outside of Expected Range
HasMember	B	468	Incorrect Pointer Scaling
HasMember	B	469	Use of Pointer Subtraction to Determine Size
HasMember	B	476	NULL Pointer Dereference
HasMember	V	587	Assignment of a Fixed Address to a Pointer
HasMember	B	763	Release of Invalid Pointer or Reference
HasMember	B	822	Untrusted Pointer Dereference
HasMember	B	823	Use of Out-of-range Pointer Offset
HasMember	B	824	Access of Uninitialized Pointer
HasMember	B	825	Expired Pointer Dereference

Vulnerability Mapping Notes

Context: Bugs cause system crashes

Two vulnerabilities were related to the trap service in NTPD. While trap is not enabled by default, if the service is explicitly enabled, attackers can send specially crafted packets to cause a null pointer dereference (CVE-2016-9311) that will crash NTPD. The configuration modification vulnerability in the control mode (mode 6) functionality of NTPD (CVE-2016-9310) can be exploited by a remote, unauthenticated attacker.

www.infoworld.com/article/3144471/ntp-fixes-denial-of-service-flaws.html

Common Weakness Enumeration

Top 25 | Top HW CWE | New to CWE? Start here!

Home | ID Lookup: Go | Lists | Community | News | Search

Weaknesses in this category are related to improper handling of pointers.

Membership

Nature	Type	ID	N
MemberOf	V	699	S
HasMember	B	466	R
HasMember	B	468	Ir
HasMember	B	469	U
HasMember	B	476	N
HasMember	V	587	A
HasMember	B	763	R
HasMember	B	822	U
HasMember	B	823	U
HasMember	B	824	A
HasMember	B	825	E

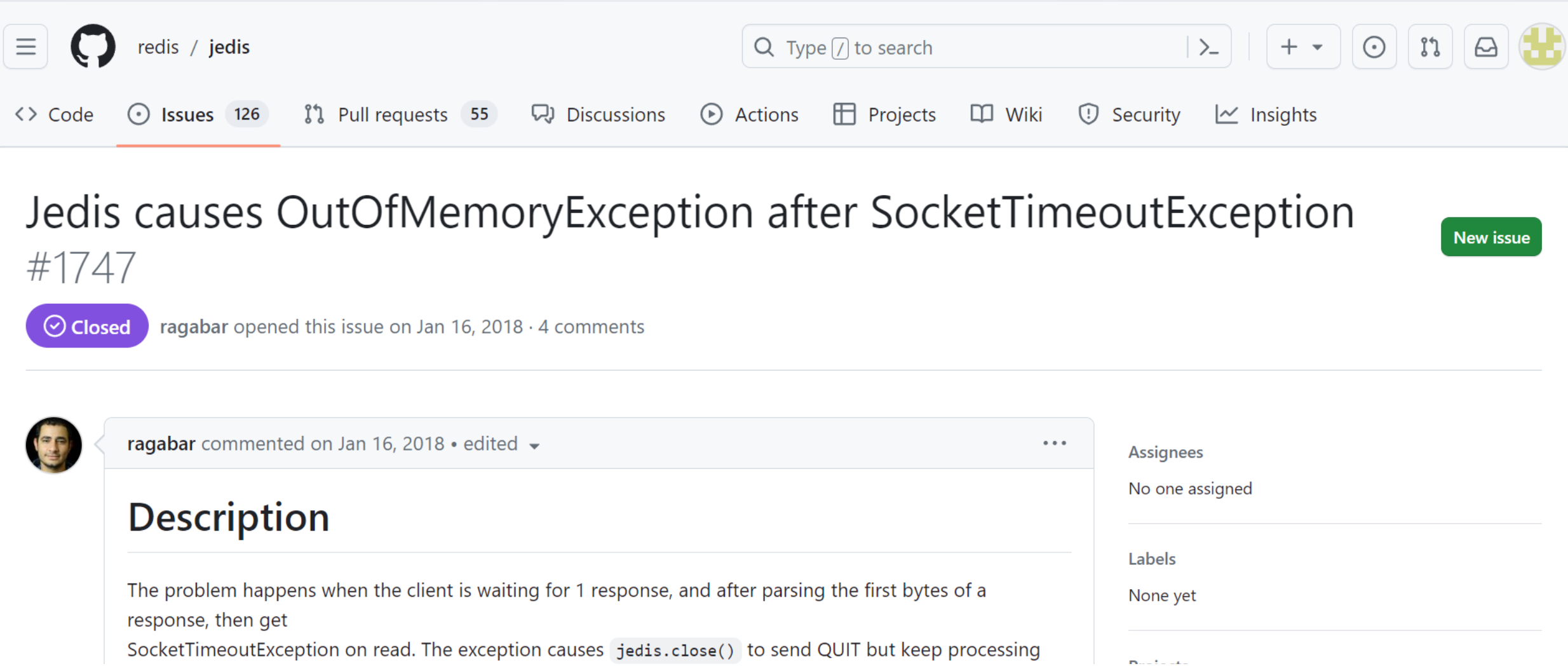
Vulnerability Mapping Notes

industrialcyber.co/news/null-pointer-dereference-vulnerability-found-in-linphone-sip-protocol-stack/

NULL pointer dereference vulnerability found in Linphone SIP Protocol Stack

SEPTEMBER 01, 2021

Context: Bugs cause system crashes



redis / jedis

Type / to search

Code | Issues 126 | Pull requests 55 | Discussions | Actions | Projects | Wiki | Security | Insights

Jedis causes OutOfMemoryException after SocketTimeoutException #1747

New issue

Closed ragabar opened this issue on Jan 16, 2018 · 4 comments

ragabar commented on Jan 16, 2018 • edited

Description

The problem happens when the client is waiting for 1 response, and after parsing the first bytes of a response, then get
SocketTimeoutException on read. The exception causes `jedis.close()` to send QUIT but keep processing

Assignees

No one assigned

Labels

None yet

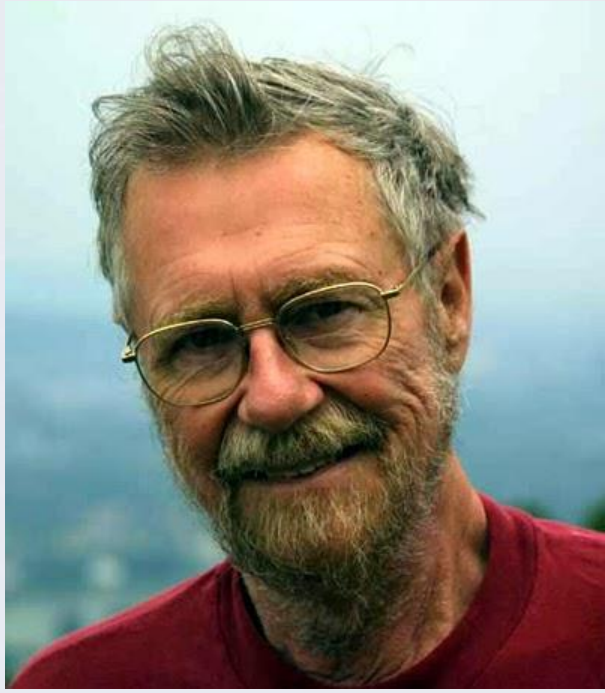
Reading a socket in a “broken” state would raise an exception…

Problem: How to ensure safety?

“Program testing can be used to show the presence of bugs, but never to show their absence!”

Edsger W. Dijkstra

Turing award in 1972: “The Humble Programmer”



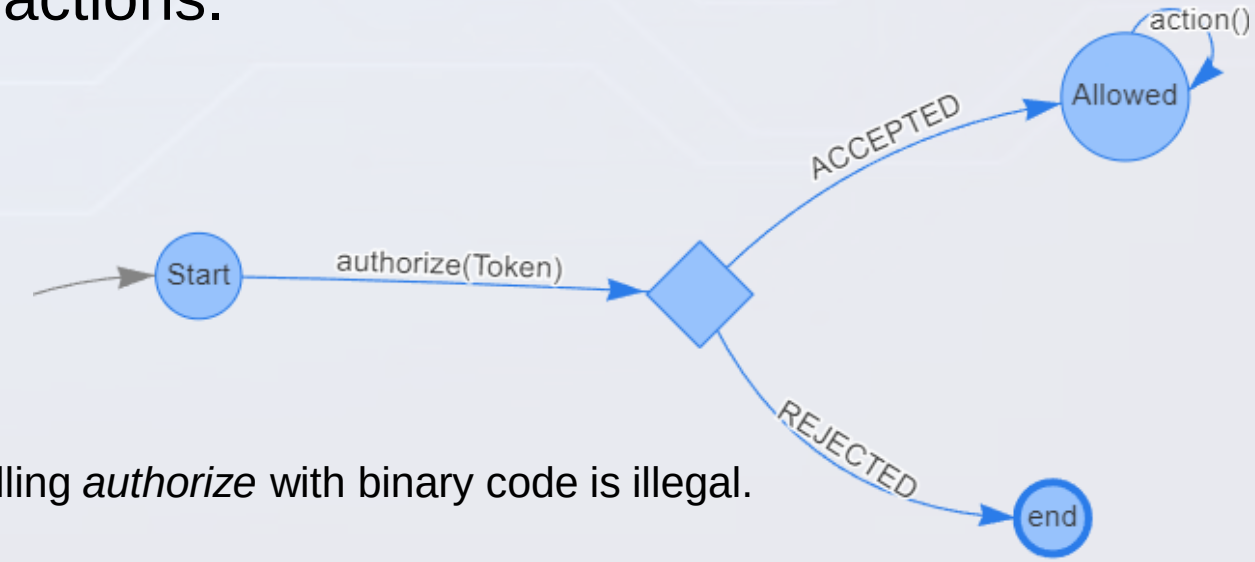
Provably safe code helps avoiding cybersecurity risks

Java Typestate Checker

Ensures that each object of Java code has its protocol respected.

Protocols define:

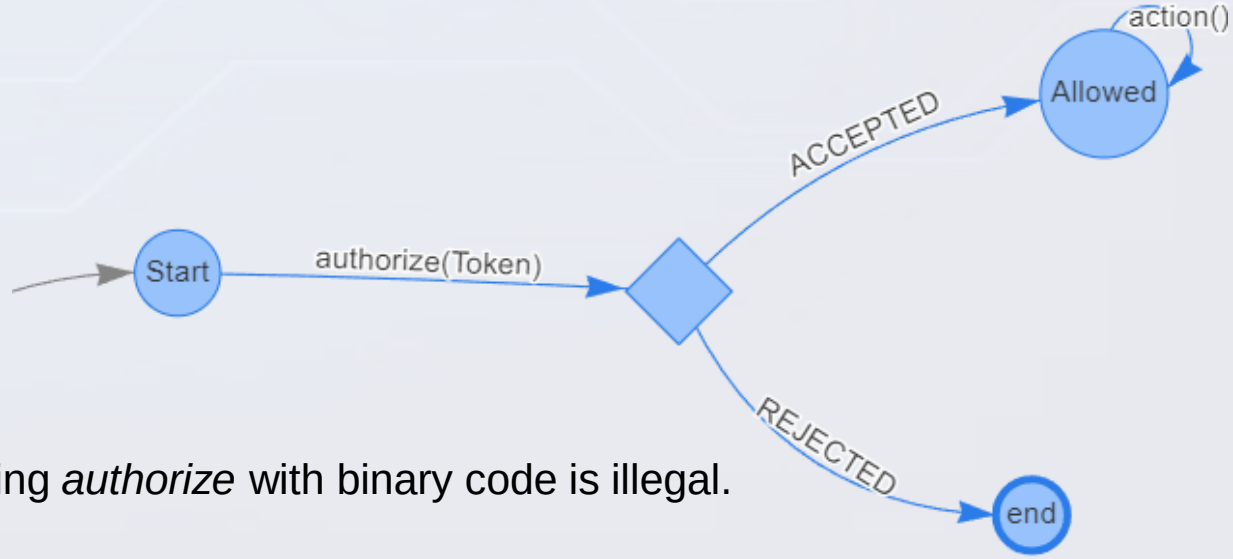
- the states of objects;
- the actions that can be safely performed in each state;
- the states resulting from those actions.



E.g., calling *authorize* with binary code is illegal.

Java Typestate Checker

1. The programmer defines the protocol;
2. The tool checks that:
   - Actions are performed in the right order (protocol compliance);
   - Protocols are completed;
   - Memory accesses are safe.



E.g., calling *authorize* with binary code is illegal.

Why the Java Typestate Checker?

INNCYBER Innovation HUB
Digital transformation, Cyber & IoT

Although there are many verification tools…

- Several check memory safety but…
- Few check protocol compliance…
- And none check protocol completion…

JaTyC is the only checking all these together.

Conclusions

JaTyC verified code is less prone to cybersecurity attacks

Avoids deploying bugs like:

- Critical action (e.g., authorization) steps being forgotten,

  like CWE-841: Improper Enforcement of Behavioral Workflow;
- Action steps performed out of order,

  like CWE-841: Improper Enforcement of Behavioral Workflow;
- Resource leakage,

  like CWE-459: Incomplete Cleanup;
- System crashes due to memory misusage,

  like CWE-465: Pointer Issues.

github.com/jdmota/java-typestate-checker

INNCYBER Innovation HUB

Digital transformation, Cyber & IoT

PREMIVALOR Consulting
Your Value Partner

altice

altice labs

edp

paloalto NETWORKS

pwc

minsait
An Indra company

cybersafe