RUHR-UNIVERSITÄT BOCHUM

RUB

RUBIN

WISSENSCHAFTSMAGAZIN

SONDERAUSGABE

IT-SICHERHEIT

Wie sich künstlich erzeugte Bilder verraten

Drei harte Nüsse für Quantencomputer

Start-up: Fit für die neue Mobilfunkgeneration

#33 Jahrgang

Sonderausgabe 2023

Fuzzing

INTELLIGENTE **AFFEN**

Bochumer Forscher finden Sicherheitslücken in IT-Systemen besonders schnell. Ihr Trick: Sie konzentrieren sich auf das Wesentliche. Das Vorgehen erklären sie mithilfe des Theorems der endlos tippenden Affen.

Ein Programmcode ist ein bisschen wie ein Dschungel: komplex aufgebaut, schwer von außen einzusehen, mit unzähligen möglichen Wegen, die man nehmen kann. Schwachstellen in einem solchen Code zu finden ist wie Tiere zwischen den Bäumen im Urwald zu suchen: Man weiß, dass sie da sind, aber man sieht sie nicht direkt. Doktorand Tobias Scharnowski entwickelt daher neue Methoden, um im Dschungel der Einsen und Nullen effizient Programmierfehler aufspüren zu können. Er forscht am Lehrstuhl für Systemsicherheit des Horst-Görtz-Instituts der Ruhr-Universität Bochum, betreut von Prof. Dr. Thorsten Holz.

Die Forscher interessieren sich vor allem für eingebettete Systeme: „Wir versuchen, die Sicherheit von Computern zu erhöhen, von denen die meisten Menschen gar nicht wissen, dass sie überhaupt Computer sind“, beschreibt Scharnowski. Smarte Glühlampen, ans Internet angeschlossene Kühlschränke oder intelligente Thermostate sind ein paar Beispiele für die eingebetteten Systeme, die auf der Agenda der Systemsicherheitsforscher stehen. Diese Gegenstände enthalten elektronische Steuerungstechnik mit vielen Zeilen Programmcode, in die sich Fehler eingeschlichen haben können. Es geht den IT-Experten aber nicht nur um Gegenstände aus dem Haushalt. Vor allem interessieren sie sich für Steuerungssysteme in der Industrie, zum Beispiel aus dem Bereich der kritischen Infrastrukturen wie der Energieversorgung. Sicherheitslücken könnten hier besonders dramatische Auswirkungen haben.

Tobias Scharnowski und Thorsten Holz nutzen das sogenannte Fuzzing, um Fehler im Programmcode aufzuspüren. Als Fuzzer bezeichnet man Algorithmen, die die zu testende Software mit zufälligen Inputs füttern und prüfen, ob sie die Anwendung damit zum Absturz bringen können. Solche Crashs weisen auf Programmierfehler hin. Immer wieder variiert der Fuzzer den Input, um Schritt für Schritt möglichst viele Programmbestandteile zu erkunden.

EINGEBETTETE SYSTEME

Ein eingebettetes System ist eine Kombination aus einer Hardware und einer Software, die einen speziellen Zweck innerhalb eines größeren Systems erfüllt – zum Beispiel im Auto die elektronische Steuerung der Sitze. Eigentlich ist ein eingebettetes System ein Computer, der einem eng umgrenzten Zweck dient.

SOFTWARE, HARDWARE, FIRMWARE

Als Hardware bezeichnet man alle Geräte im Computerbereich – anders als Soft- und Firmware existiert sie also in der realen Welt, die mit den Händen angefasst werden kann. Software und Firmware hingegen sind Programme, die nur virtuell existieren. Die Firmware ist dabei eine spezielle Art von Software, die verwendet wird, um Hardware zu steuern; sie erfüllt also einen genau definierten Zweck für diese Hardware.

FUZZING

Fuzzing ist eine Methode zum Finden von Schwachstellen in Software. Dabei wird die Software mit vielen verschiedenen Inputs gefüttert und solange ausgeführt, bis eine Eingabe sie zum Absturz bringt. Ein Programmabsturz deutet auf einen Fehler hin.

Die beiden Forscher suchen nach Schwachstellen im Programmcode von Firmware, also von einer speziellen Software, die zur Steuerung von Hardware gebraucht wird.

Für bestimmte Anwendungsbereiche ist das Fuzzing bereits etabliert, zum Beispiel, um Betriebssysteme wie Windows oder Linux zu testen. Eingebettete Systeme hingegen wurden noch nicht ausgiebig damit untersucht; denn sie bringen einige Herausforderungen mit sich: Bei ihnen ist die Software – die sogenannte Firmware – in eine Hardware eingebettet, mit der sie interagiert. Über die Hardware und ihre Funktionsweise haben die Forscher aber in der Regel wenig Informationen. „Das ist wie eine Blackbox für uns“, beschreibt Thorsten Holz. Hinzu kommt, dass diese Blackbox in der Regel nicht besonders leistungsstark ist – oft haben die Systeme verhältnismäßig wenig Speicher und langsame Prozessoren. Ein Problem, wenn die Forscher das Fuzzing direkt im System durchführen wollen. Es würde viel zu lange dauern, alle möglichen Inputs durchzuprobieren und auf die Antwort des Systems zu warten.

Deswegen analysiert das Team die Firmware nicht direkt in der industriellen Steuereinheit oder in der Glühbirne. Stattdessen bauen sie die Hardware virtuell nach – emulieren nennt sich dieser Prozess. Der Emulator gaukelt der Firmware vor, sich in dem realen Gegenstand zu befinden. Dazu muss er genauso mit dem Programm interagieren, wie es die echte Hardware tun würde. „Wir müssen also alle Schnittstellen, die es zwischen Hardware und Firmware gibt, nachahmen“, erklärt Thorsten Holz. Gelingt das, können die Wissenschaftler die Firmware in einem leistungsfähigen System testen.

Trotzdem würde es lange dauern, wenn sie ihren Fuzzer alle theoretisch denkbaren Inputs ausprobieren lassen würden. Deswegen schalten die Forscher dem eigentlichen Fuzzing-Prozess einen weiteren Schritt vor, in dem sie die möglichen Inputs eingrenzen. Sie modellieren zunächst, in welchem Rahmen sich die Eingaben befinden müssen, um für die Firmware logisch zu sein. Ein Beispiel: Gehen wir davon es, dass es sich bei der Hardware um einen Kühlschrank mit einem Temperaturfühler handelt. Die gemessenen Temperaturen kann die Kühlschrank-Hardware an die Software ▸

> „WENN EIN SYSTEM NOCH NIE MIT FUZZING GETESTET WURDE, DANN GIBT ES DARIN AUCH UNENTDECKTE SCHWACH-STELLEN.“

Thorsten Holz

des Kühlschranks, also seine Firmware, melden. Realistischerweise können nicht alle möglichen Temperaturen auftreten, sondern nur ein gewisser Bereich. Daher ist auch die Firmware nur für einen bestimmten Temperaturbereich programmiert. Andere Werte könnte sie gar nicht verarbeiten, also muss man sie auch nicht im Fuzzing testen.

„Im Fuzzing-Prozess nutzen wir also nur die Inputs, die die Firmware auch erwartet und mit denen sie umgehen kann“, beschreibt Thorsten Holz und vergleicht den Prozess mit dem Infinite Monkey Theorem: „Dieses Theorem besagt, dass, wenn man Affen lange genug auf eine Tastatur drücken lassen würde, irgendwann auch Shakespeares Werke dabei herauskommen würden.“ So wäre es mit dem Fuzzer auch: Wenn man ihn lange genug probieren lassen würde, würde er durch Zufall irgendwann sinnvolle Inputs nutzen. Aber es würde lange dauern. „Wir wollen unsere Affen aber etwas intelligenter machen“, sagt Tobias Scharnowski. „Wir nehmen ihnen alle Tasten weg, die sie nicht brauchen, und versuchen sie dazu zu bringen, nur sinnvoll auf die Tasten zu drücken. Mit den Inputs, die übrigbleiben, können wir den Code trotzdem bis in die hintersten Ecken testen.“ Auf diese Weise wird das Fuzzing mit dem Bochumer System, Fuzzware genannt, besonders effizient.

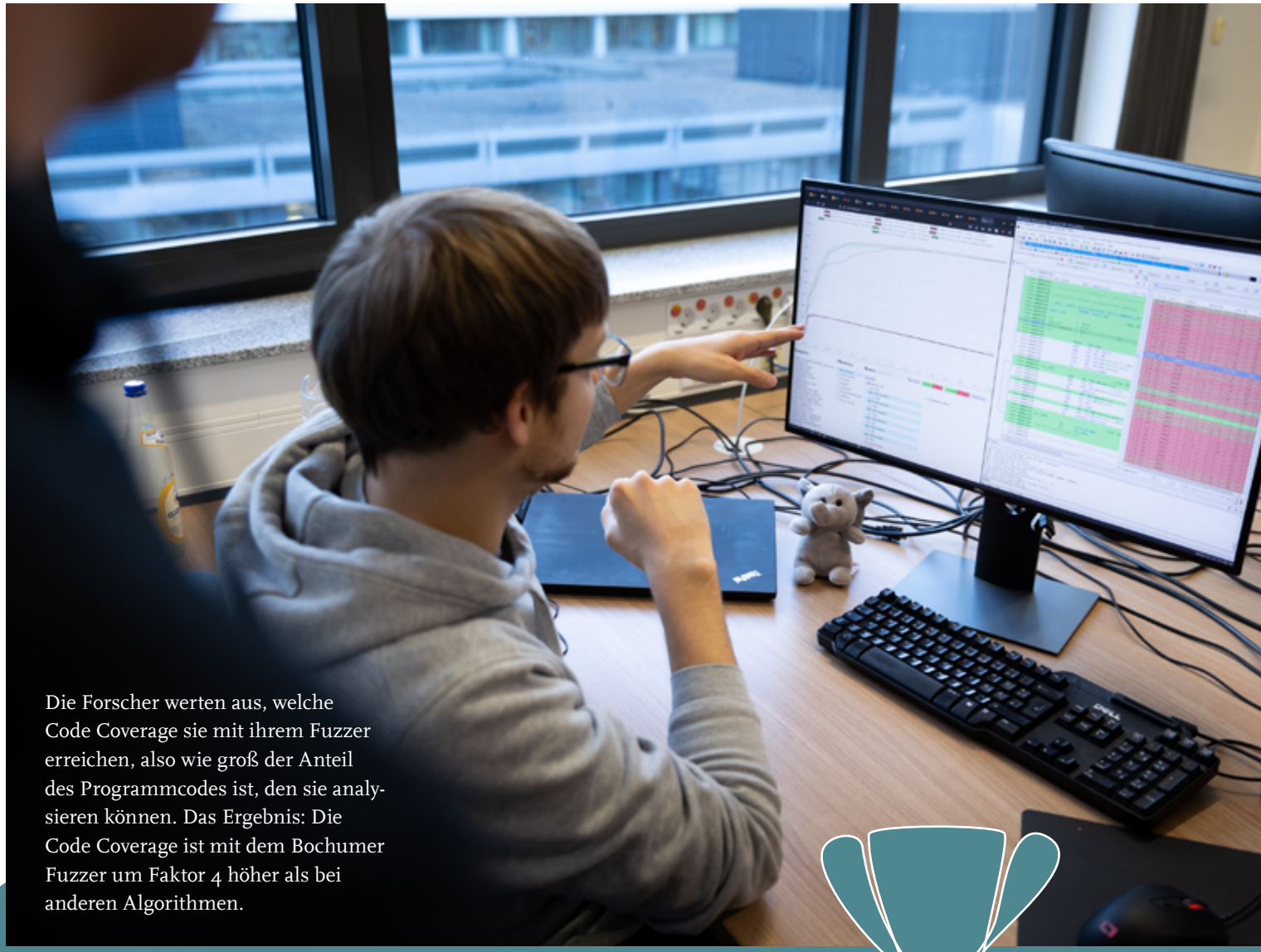
Zusammen mit Kolleginnen und Kollegen aus Santa Barbara und Amsterdam testete das Bochumer Team 77 Firmwares mit Fuzzware. Im Vergleich zu herkömmlichen Fuzzing-Methoden sortierten sie bis zu 95,5 Prozent der möglichen Inputs aus. Trotzdem gelang es ihnen, mit dem Fuzzware-System in der gleichen Zeit bis zu dreimal mehr von dem Programmcode zu checken wie mit herkömmlichen Verfahren. Dabei fand die Gruppe auch neue Schwachstellen, die mit anderen Fuzzing-Methoden unentdeckt geblieben waren.

„Man findet eigentlich immer was“, weiß Thorsten Holz. „Wenn ein System noch nie mit Fuzzing getestet wurde, dann gibt es darin auch unentdeckte Schwachstellen.“ Gerade bei eingebetteten Systemen ist es für Programmiererinnen und Programmierer nahezu unmöglich, einen perfekten Code auf die Beine zu stellen. „Um mit der Hardware von eingebetteten Systemen sprechen zu können, muss man eine Low-Level-Programmiersprache nutzen“, erklärt Tobias Scharnowski. Programmierer können in vielen Bereichen nicht auf Codeschnipsel zurückgreifen, die für andere Anwendungen entwickelt wurden. Sie müssen ihren Code von Grund auf neu aufbauen. Gerade Randfälle – Zustände, in denen sich das System selten befindet – werden dann eventuell nicht bedacht. „Für unsere Fuzzer sind diese Zustände aber leicht zu analysieren“, sagt Scharnowski. „Sie können daher helfen, die Systeme robuster zu machen.“ Gefundene Schwachstellen melden die Forscher an die Hersteller und tragen so zu mehr Sicherheit in Industrie, Glühbirnen, Kühlschränken und Co. bei.

Text: jwe, Fotos: ms

Tobias Scharnowski ist Doktorand am Horst-Görtz-Institut für IT-Sicherheit.

Thorsten Holz war viele Jahre einer der Principal Investigators des Exzellenzclusters CASA.



Die Forscher werten aus, welche Code Coverage sie mit ihrem Fuzzer erreichen, also wie groß der Anteil des Programmcodes ist, den sie analysieren können. Das Ergebnis: Die Code Coverage ist mit dem Bochumer Fuzzer um Faktor 4 höher als bei anderen Algorithmen.

Im Gespräch

„EIN RAUNEN GING DURCH DIE MENGE“

Softwarefirmen freuen sich, wenn Forschende Fehler in ihrem Code finden, bevor es Angreifer tun. Sie richten sogar Wettbewerbe für die Fehlersuche aus. Das Bochumer Team hat schon reichlich Preisgelder eingeheimst.

Herr Scharnowski, in Ihrem Bereich ist öfter mal von Bug Bounties die Rede. Was muss man sich darunter vorstellen?
Es ist eine Art Bonusprogramm von Softwareunternehmen für das Aufspüren von Schwachstellen. Je schwerwiegender die Schwachstelle ist, die man entdeckt, desto höher der Gewinn. Manche Hersteller schreiben sogar Wettbewerbe aus.

Haben Sie schon einmal teilgenommen?
2020 habe ich mit Kollegen beim Pwn2Own-Wettbewerb in Miami mitgemacht. Er wurde von verschiedenen Herstellern aus dem industriellen Sicherheitsbereich ausgelobt. Es ging um Geräte, die industrielle Anlagen steuern. Wir haben unter anderem das sogenannte DNP3-Protokoll angegriffen, das für die Kommunikation zwischen den Steuersystemen eingesetzt wird, beispielsweise im kritischen Energiesektor. Wir haben es als einzige geschafft, die höchste Kategorie für diese Aufgabe zu erreichen, und konnten komplette Kontrolle über das Programm gewinnen.

Das klingt nach einem besonderen Erfolg.
Ja, das war ein besonderes Erlebnis. Bei dem Wettbewerb waren verschiedene Ziele ausgelobt worden, und zu Beginn wurde verkündet, welches Team welches Ziel in Angriff nehmen würde. Als unsere Idee vorgestellt wurde, ging ein Raunen durch die Menge.

Und was war der Gewinn?
Wir haben zu dritt 87.500 US-Dollar Preisgeld bekommen. Das gibt uns nun die Freiheit, Software und Equipment für unsere nächsten Abenteuer dieser Art zu kaufen.

jwe

REDAKTIONSSCHLUSS

Die Hasen im CASA Universe sind aufgeschreckt: Der scheinbar gut gesicherte Zugang zum Karottenvorrat von Hase Mark wurde gehackt und alle Wintervorräte geraubt. Die mutige Häsin Betty macht sich daraufhin auf die Suche nach Unterstützung im nahegelegenen CASA Hub C – einem geheimnisvollen Ort, der Lösungen für digitale Sicherheit bereithalten soll. So beginnt das Abenteuer von Häsin Betty, der Protagonistin des ersten Wissenschaftscomics des Exzellenzclusters CASA. Gemeinsam mit Betty lernen die Leserinnen und Leser bei ihrem Streifzug durch den Research Hub die Forschungsschwerpunkte und Herausforderungen kennen, mit denen sich die Wissenschaftlerinnen und Wissenschaftler im Forschungsbereich Hub C „Sichere Systeme" tagtäglich beschäftigen. Wie Sie alle Comics der Reihe kostenlos lesen können, erfahren Sie unter:

↗ *casa.rub.de/outreach/wissenschaftscomics*

Auflösung
DEEPFAKE-QUIZ
Folgende Gesichter sind echt:
1a, 2a, 3b, 4a, 5b, 6a

IMPRESSUM

HERAUSGEBER: Exzellenzcluster CASA und Horst-Görtz-Institut für IT-Sicherheit der Ruhr-Universität Bochum in Verbindung mit dem Dezernat Hochschulkommunikation der Ruhr-Universität Bochum (Hubert Hundt, v.i.S.d.P.)

REDAKTIONSANSCHRIFT: Dezernat Hochschulkommunikation, Redaktion Rubin, Ruhr-Universität Bochum, 44780 Bochum, Tel.: 0234/32-25228, rubin@rub.de, news.rub.de/rubin

REDAKTION: Dr. Julia Weiler (jwe, Redaktionsleitung); Meike Drießen (md); Lisa Bischoff (lb)

FOTOGRAFIE: Michael Schwettmann (ms), Dammstr. 6, 44892 Bochum, Tel.: 0177/3443543, info@michaelschwettmann.de, www.michaelschwettmann.de

COVER: Sashkin – stock.adobe.com

BILDNACHWEISE INHALTSVERZEICHNIS: Michael Schwettmann

GRAFIK, ILLUSTRATION, LAYOUT UND SATZ:
Agentur für Markenkommunikation, Ruhr-Universität Bochum,
www.einrichtungen.rub.de/de/agentur-fuer-markenkommunikation

DRUCK: LD Medienhaus GmbH & Co. KG, Van-Delden-Str. 6-8, 48683 Ahaus, Tel.: 0231/90592000, info@ld-medienhaus.de, www.ld-medienhaus.de

AUFLAGE: 4.700

BEZUG: Die reguläre Ausgabe von Rubin erscheint zweimal jährlich und ist erhältlich im Dezernat Hochschulkommunikation der Ruhr-Universität Bochum. Das Heft kann kostenlos abonniert werden unter news.rub.de/rubin. Das Abonnement kann per E-Mail an rubin@rub.de gekündigt werden. Die Sonderausgabe 2023 ist erhältlich beim Horst-Görtz-Institut für IT-Sicherheit. Interessierte können sich per E-Mail an hgi-presse@rub.de melden.

ISSN: 0942-6639