UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
14/670,895	03/27/2015	Xun Wang	331792.02	7391

39254 7590 09/26/2019

Barta, Jones & Foley, P.C.
(Patent Group - Microsoft Corporation)
2805 Dallas Parkway
Suite 222
Plano, TX 75093

EXAMINER	
MILLS, PAUL V	

ART UNIT	PAPER NUMBER
2196	

NOTIFICATION DATE	DELIVERY MODE
09/26/2019	ELECTRONIC

Please find below and/or attached an Office communication concerning this application or proceeding.

The time period for reply, if any, is set in the attached communication.

Notice of the Office communication was sent electronically on above-indicated "Notification Date" to the following e-mail address(es):

docket@bjfip.com
usdocket@microsoft.com
uspto@dockettrak.com

UNITED STATES PATENT AND TRADEMARK OFFICE

BEFORE THE PATENT TRIAL AND APPEAL BOARD

Ex parte XUN WANG, ADRIAN EMIL STEPAN, and TIMOTHY DAVID EBRINGER

Appeal 2019-000612
Application 14/670,895[1]
Technology Center 2100

Before MAHSHID D. SAADAT, MARC S. HOFF, and JOYCE CRAIG, *Administrative Patent Judges*.

HOFF, *Administrative Patent Judge*.

DECISION ON APPEAL

STATEMENT OF THE CASE

Appellants appeal under 35 U.S.C. § 134(a) from a final rejection of claims 1–20.[2] We have jurisdiction under 35 U.S.C. § 6(b).

We reverse.

Appellants' invention concerns malware detection by emulation of virtualized obfuscated program code. An emulation engine emulates a program containing a mix of native code and custom (e.g., virtualized

---

[1] Appellants state the real party in interest is Microsoft Technology Licensing, LLC. App. Br. 1.

obfuscated) code. A custom emulation component handles custom instruction blocks, processing each such block into an intermediate language for emulation by the emulator component. A native component handles native instruction blocks. Spec. ¶ 5. A dynamic translation engine includes the capability to dynamically switch front-end translation components and thereby handle native executable code as well as any target code (e.g., custom bytecode or script) in the same virtual machine instance. Spec. ¶ 17.

Claim 1 is exemplary of the claims on appeal:

> 1. In a computing environment, a system configured to emulate a program comprising:
>
> an emulator component;
>
> a custom emulation component processing a custom instruction block from program code of the program into an intermediate language for emulation by the emulator component;
>
> a native component processing a native instruction block from the program code of the program and outputting instructions corresponding to the native instruction block to the emulator component; and
>
> mixed code processing logic dynamically switching between the custom
> emulation component and the native component as the system processes the program code responsive to detection of at least one jump instruction in the program code.

Appendix A B-1(Claims Appendix).

The Examiner relies upon the following prior art in rejecting the claims on appeal:

Gheorghescu et al. US 2006/0123244 A1 June 8, 2006

Guillot et al., Automatic binary deobfuscation, 6 J. Comput. Virol. 261–276 (2010).

Claims 1–20 stand rejected under 35 U.S.C. § 103(a) as being unpatentable over Gheorghescu and Guillot.

Throughout this decision, we make reference to the Appeal Brief ("App. Br.," filed May 28, 2018), the Reply Brief ("Reply Br.," filed October 31, 2018), and the Examiner's Answer ("Ans.," mailed September 21, 2018) for their respective details.

ISSUE

Appellants' arguments present us with the following issue:

Does the combination of Gheorghescu and Guillot teach or suggest mixed code processing logic dynamically switching between the custom emulation component and the native component responsive to detection of at least one jump instruction in the program code?

ANALYSIS

Independent claim 1 recites, in pertinent part, "mixed code processing logic dynamically switching between the custom emulation component and the native component as the system processes the program code responsive to detection of at least one jump instruction in the program code." Independent claims 10 and 16, similarly, recite "dynamically switching between processing by the custom emulation component and the native component responsive to detection of at least one jump instruction in the program code."

The Examiner implicitly finds that Gheorghescu does not teach this limitation, in that the Examiner does not cite to a teaching in Gheorghescu and treats this limitation in detail in the discussion of Guillot. Final Act. 6–7.

The Examiner cites sections 2.5-2.6, 4.3, and 4.5 of Guillot as teaching the limitation at issue. *Id.* The Examiner equates Guillot's teaching that "we are able to hook every call to the virtual machine, to proceed to the symbolic execution of the bytecode, to update both the context and the memory of the process and then return to the original code" to the claimed switch performed in response to "detection of at least one jump instruction." *Id.*

We do not agree with the Examiner's interpretation of Guillot. First, we do not agree that Guillot teaches dynamic switching responsive to *detection of a jump instruction*. We agree with Appellants that Guillot's teachings are limited to *manually* walking "line by line of a program flow to inject a break anywhere [they] see a call to a VM." Reply Br. 5. Second, Appellants argue, and we agree, that Guillot teaches that the alleged switch between custom emulation component and native component is responsive to a manually injected break at the entry point of a virtual machine. Reply Br. 4; Guillot § 4.1, Fig. 21. Such a manually injected break does not correspond to the claim requirement of *dynamic switching* responsive to "mixed code processing logic detecting a jump instruction in the program code being emulated." Reply Br. 5.

We conclude that the Examiner erred in rejecting claims 1–20 over Gheorghescu and Guillot. We do not sustain the § 103(a) rejection.

CONCLUSION

The combination of Gheorghescu and Guillot does not teach or suggest mixed code processing logic dynamically switching between the

custom emulation component and the native component responsive to detection of at least one jump instruction in the program code.

ORDER

The Examiner's decision to reject claims 1–20 is reversed.

REVERSED