**Title: Sandbox or Sieve?: Blocking Computer Malware by Closing Holes in Security Sandboxes,** 399 words

**Published (scheduled):** Building Bridges, 2016 edition

**Awarded:** Outstanding Abstract Award

**Presenter:** Joshua DuFault, El Camino College

**Mentor:** ProfessorSolomon Russell

Information is valuable: stolen banking data can cost a retiree his life savings and stolen personal information can cost a worker her good credit record. In all, cybercrime costs the world over 100 billion dollars a year (Anderson et al.). Antivirus programs have not been able to keep up with the massive scale of digital attacks ("Application"); a superior approach is to isolate potentially dangerous code with reduced privileges in a process called sandboxing. Unfortunately, several commonly used sandboxing implementations introduce vulnerabilities that can be exploited by malware to break out of the sandbox.

One currently used but easily defeated sandboxing model is to run multiple windows on the same desktop with restricted tokens, security identifiers that give an application restricted rights. However, a malware compromised program can use a shatter attack to break out of this type of sandbox (Kashyap and Wojtczuk). Some researchers claim that this vulnerability can be mitigated by using a security feature called job objects, which further restrict a program's access (Close, Karp, and Stiegler). This is incorrect. Although job objects can restrict malware's access to an individual target program, they cannot prevent access to global resources such as keyboard hooks that allow system-wide keylogging (Leblanc).

Although a commonly used security feature, the desktop alone can be bypassed easily because windows securable objects, such as desktops, cannot be restricted to individual programs; they are restricted to individual user tokens. The common solution is to launch an application process thread on an alternate desktop with restrictions preventing it from accessing any desktop. Normal applications do not work with these restrictions. Integrity levels can prevent data theft out of memory but are too few to be used effectively and do not prevent file theft.

My experiment tested launching an application as an alternate user on an alternate desktop with an inheritable deny Access Control List (ACL) for the alternate user placed on Windows Explorer. This sandbox was then compared to existing sandbox models. The ACL automatically propagated to all applications that Explorer launched. This sandbox successfully prevented data theft, keylogging, and shatter attacks while existing general purpose sandboxes were defeated. It also proved possible to protect sensitive data in a different sandbox by denying the main user access to programs launched on the alternate desktop. If future versions of Windows replaced Windows 10's cosmetic desktop feature with this sandbox it would greatly enhance protection against data theft.

**Works Cited**

Anderson, Ross, Chris Barton, Rainer Bohme, Richard Clayton, Michel Eeten, Michael Levi, Tyler Moore, and Stefan Savage. "Measuring the cost of cybercrime." *The Economics of Information Security and Privacy*. 2013. Web. 12 Feb. 2016.

"Application Whitelisting Using Software Restriction Policies." *National Security Agency*. 1 Aug. 2010. Web. 15 Nov. 2015.

Close, Tyler, Alan Karp, and Marc Stiegler. "Shatter-proofing Windows." *Hewlett-Packard Laboratories.* 9 May 2005. Web. 15 Nov. 2015.

Kashyap, Rahul, and Rafal Wojtczuk. "Application Sandboxes: A Pen-Tester's Perspective." *Bromium Labs.* 23 Jul. 2013. Web. 15 Nov. 2015.

Leblanc, David. "Practical Sandboxing: Techniques for Isolating Processes." *Black Hat USA 2007*. Caesars Palace, Las Vegas, NV. 2 Aug. 2007. Presentation.