Using Inputs and Context to Verify User Intention

He (Shawn) Shuang

A long-standing problem in Internet systems security is how a service can establish whether a request received from a remote client[1] is trustworthy, meaning that they originate from a proper user and is untempered during the transmission. Consider an online payment service where Alice wishes to pay $1 to Bob. However, an attacker, Mallory, wishes to forge a request that will cause Alice to pay $100 to Mallory. The most basic attack is for Mallory to steal login/password from Alice and impersonate Alice to the service. This attack is prevented by multi-factor authentication such as SMS message verification. However, authentication can be defeated if Mallory is able to find a vulnerability in Alice's client that allows Mallory to install malicious software (i.e., malware). The malware can tamper with the display and trick the user into clicking on something but use that click to trigger user unintended actions. For instance, the user clicks may be hijacked to trigger the transaction confirmation to Mallory [3, 5, 8]. Still, other proposals attempt to certify requests by checking for human interaction over hardware devices such as the mouse or keyboard [2, 4, 7]. However, they suffer from *semantic gap* [1, 6]—they can only assert that human interaction occurred around the same time as the request, but say nothing about the *context* in which that interaction occurred. Attackers can harvest user activities in other applications to get certification for the forged request.

The key property missing from all previous proposals is that human interactions, and the context in which they occur, are not securely bound to the request being received by the service. To solve this problem, we propose *Attested Interactions*, which not only attest that there is human interaction, but attest to the *input* and *context* of that interaction. For example, to prevent forgery of a payment request, the input could be the keyboard inputs of the user filling out the payment form and the mouse input of the user clicking the "Pay" button. Similarly, the user context could be a video or screen capture of the display showing the interaction of the user with the properly rendered website. These, when combined with information about the execution integrity of the client, allow the service to determine whether requests are legitimate or forged, in many of the situations that current solutions either do not address or only partially address.

There are several technical difficulties: 1) secure access to user inputs and context, 2) properly rendered user context, 3) execution integrity of service requests generation and 4) privacy concerns among the captured user data. To securely capture user context, Attested Interaction uses an external hardware device, such as an FPGA, that sits in between the user's machine and the display. Anything user *sees*, will be captured securely and sent back to the machine. To prevent malicious software from tampering with the captured context, Attested Interaction deploys a micro hypervisor to isolate the device and data from malicious software. The hypervisor also captures user inputs. Finally, to support the diverse requirement of context verification and request generation logic, Attested Interaction uses Intel SGX enclaves, a trusted computing mechanism. To protect privacy, all operations happen locally on user clients; no information is sent remotely. The verification result (a single bit) will be attest to the server cryptographically to prove that the request is consistent with user intention.

[1] We use the term *client* to denote any computing device, including smartphones, PCs and even embedded/IoT devices.

References

[1] P. M. Chen and B. D. Noble. When virtual is better than real. In *Proceedings of the Eighth Workshop on Hot Topics in Operating Systems*, pages 133–138. IEEE, 2001.

[2] W. Cui, R. H. Katz, and W.-t. Tan. Binder: An extrusion-based break-in detector for personal computers. In *Proceedings of the 2005 USENIX Annual Technical Conference*, Berkeley, CA, USA, April 2005. USENIX Association. URL https://www.microsoft.com/en-us/research/publication/binder-an-extrusion-based-break-in-detector-for-personal-computers/.

[3] Y. Fratantonio, C. Qian, S. P. Chung, and W. Lee. Cloak and dagger: from two permissions to complete control of the ui feedback loop. In *2017 IEEE Symposium on Security and Privacy (SP)*, pages 1041–1057. IEEE, 2017.

[4] R. Gummadi, H. Balakrishnan, P. Maniatis, and S. Ratnasamy. Not-a-bot: Improving service availability in the face of botnet attacks. In *Proceedings of the 6th USENIX Symposium on Networked Systems Design and Implementation*, NSDI'09, pages 307–320, Berkeley, CA, USA, 2009. USENIX Association. URL http://dl.acm.org/citation.cfm?id=1558977.1558998.

[5] L.-S. Huang, A. Moshchuk, H. J. Wang, S. Schecter, and C. Jackson. Clickjacking: Attacks and defenses. In *Presented as part of the 21st USENIX Security Symposium (USENIX Security 12)*, pages 413–428, Bellevue, WA, 2012. USENIX. ISBN 978-931971-95-9. URL https://www.usenix.org/conference/usenixsecurity12/technical-sessions/presentation/huang.

[6] B. Jain, M. B. Baig, D. Zhang, D. E. Porter, and R. Sion. Sok: Introspections on trust and the semantic gap. In *IEEE Symposium on Security and Privacy (SP)*, pages 605–620. IEEE, 2014.

[7] Y. Jang, S. P. Chung, B. D. Payne, and W. Lee. Gyrus: A framework for user-intent monitoring of text-based networked applications. In *NDSS*, 2014.

[8] L. Stefanko. Android trojan steals money from PayPal accounts even with 2FA on, dec December 2018. URL https://www.welivesecurity.com/2018/12/11/android-trojan-steals-money-paypal-accounts-2fa/.