Detecting web server take-over attacks through objective verification actions

Mark Vella[1] and Sotirios Terzis[2]

[1] University of Malta
[2] University of Strathclyde, Glasgow

Attacks targeting web servers pose a major security threat. Typically prone to a mix of infrastructure and application-level security vulnerabilities, they serve as the lowest hanging fruit for intruders wanting to gain unauthorized access to the entire host network. This is specifically the case for 'server take-over' attacks, whose immediate objective is to gain unauthorized remote access to the host server, for example through shell-spawning, backdoor-ing or botnet joining[3].

From attack/malware to exploit detection. The most common option to detect such attacks consists of recognizing attack packets or malicious binaries at the network and host levels. However, these detectors make use of misuse detection rules that can be easily evaded through obfuscation/polymorphism and are highly limited in matching attacks exploiting zero-day vulnerabilities. Recently, the research domain is witnessing a shift from this malware/attack-centered approach to one that focuses on the targeted application. Specifically, these what we call *exploit detectors*, operate by dynamically monitoring the execution of the potentially vulnerable application/platform for indications of successful exploits.

Analysis of runtime information either follows the classic misuse/anomaly detection dichotomy, such as recognizing known malicious system call sequences of parasitic malware [4] or detecting jump targets/control sequences considered tainted [5], as compared to recognizing anomalous HTTP-backend traffic pairings caused by SQL injection attacks [2], or goes one step further and outright change the execution environment to block successful exploit execution [3]. Overall, these detectors leverage the higher quality information obtained through dynamic analysis to generalize beyond exploits instances to an entire exploit category, and are also resilient to content obfuscation resulting in increased detection effectiveness. Furthermore, the dynamic verification of successful exploitation avoids false positives (FP) however at the cost of invasive instrumentation and high monitoring overheads. An aggregation of such detectors would be an obvious method to effectively protect from take-over attacks, though performance overheads and compatibility issues abound.

Problem definition. We aim for a dynamic analysis-based method that generalizes from known attacks over the objective dimension in order to detect web server take-overs. Specifically, the proposed solution is required to: 1) Combine

[3] http://www.symantec.com/security_response/publications/threatreport.jsp

multiple relevant exploit categories in its detection scope; 2) Translate the high-level objective description to a low-level one in terms of events associated with the execution of the vulnerable web application; 3) Retain the polymorphic/zero-day attack resilience and low FP properties of dynamic analysis detectors; and 4) Not increase overheads beyond that of individual exploit detectors.

Proposition. We propose a solution that: focuses on attack rather than normal content to avoid FP; combines known exploit categories from an attack objective dimension into a single solution through causal relations; is verification-based as per existing exploit detectors; relies on externally observable dynamic analysis events so as not to impose intrusive instrumentation; and uses modified LAMBDA [1] to translate between a high-level detection heuristic and its low-level counterpart for the take-over objective. The result is an objective verification approach that verifies the objective's success based on its pre-/post-conditions expressed in terms of dynamic analysis events, where: the preconditions are defined on process input, post-conditions are defined over events resulting from input processing associated with objective's attainment, whilst the verification actions confirm the causal relation between the input and the events. We modify the LAMBDA language to fit this approach so that it reflects the single-step attacks and objective focus, where pre- and post-conditions describe the monitored process's state, and verification and detection actions are fused together. The high-level detection heuristic "Attack input needs to inject code to setup the take-over and then either connects to a remote listening port, or start listening on a (newly-opened/reused) web port" gets the low-level translation:

objective *WWW_take-over*(HTTPReq, Platform, WebApp_Interpreter_List, Interpreter_List, WWWProcTree)
pre: $injectable\_netcode(HTTPReq, [Platform : Interpreter\_List]) \vee$
$codebase\_extending\_script(HTTPReq, WebApp\_Interpreter\_List)$
post: $net\_start\_listen(WWWProcTree, (Local\_IP, Local\_Port)) \vee$
$net\_connect(WWWProcTree, (Remote\_IP, Remote\_Port)) \vee$
$((create(File) \vee modify(File)) \wedge interpretable(File, WebApp\_Interpreter\_List))$
verification/detection: G_1
where: $action(G_1) = ((Local\_IP, Local\_Port) \in ipport\_pairs(Injectable\_netcode) \vee$
$(Remote\_IP, Remote\_Port) \in ipport\_pairs(Injectable\_netcode)) \vee$
$contains(File, codebase\_script\_blocks(Codebase\_extending\_script))$

The post-condition events can be used for the immediate recovery of a subverted system, whilst the HTTP request implicated in the precondition can be used to track the exploited vulnerability for long term recovery. The implementation relies on network/memory/disk forensic probes to supply the required events, and a purposely-built emulator to identify and extract information from potential instances of $Injectable\_netcode$ and $Codebase\_extending\_script$. Ongoing work concerns the implementation of an experimentation test-bed that provides real-world traffic and a range of successfully executing attacks that are representative of the take-over objective.

References

1. Cuppens, F., Ortalo, R.: Lambda: A language to model a database for detection of attacks. In: Recent advances in intrusion detection. pp. 197–216. Springer (2000)
2. Le, M., Stavrou, A., Kang, B.: Doubleguard: Detecting intrusions in multitier web applications. vol. 9, pp. 512–525. IEEE (2012)
3. Locasto, M.E., Wang, K., Keromytis, A.D., Stolfo, S.J.: Flips: Hybrid adaptive intrusion prevention. In: Recent Advances in Intrusion Detection. pp. 82–101. Springer (2006)
4. Srivastava, A., Giffin, J.: Automatic discovery of parasitic malware. In: Recent Advances in Intrusion Detection. pp. 97–117. Springer (2010)
5. Xu, W., Bhatkar, S., Sekar, R.: Taint-enhanced policy enforcement: A practical approach to defeat a wide range of attacks. In: Proceedings of the 15th USENIX Security Symposium. pp. 121–136 (2006)