# Anomalous Taint Detection (Extended Abstract)*

Lorenzo Cavallaro[1] and R. Sekar[2]

[1] Department of Computer Science, University of California at Santa Barbara, USA.
[2] Department of Computer Science, Stony Brook University, USA.

**Abstract.** We propose *anomalous taint detection*, an approach that combines fine-grained taint tracking with learning-based anomaly detection. Anomaly detection is used to identify behavioral deviations that manifest when vulnerabilities are exercised. Fine-grained taint-tracking is used to target the anomaly detector on those aspects of program behavior that can be controlled by an attacker. Our preliminary results indicate that the combination increases detection accuracy over either technique, and promises to offer better resistance to mimicry attacks.

## 1 Introduction

A number of approaches have been developed for mitigating software vulnerabilities. Of these, learning-based anomaly detection has been popular among researchers due to its ability to detect novel attacks. Although the basic assumption behind anomaly detection, which states that attacks manifest unusual program behaviors, is true, the converse does not hold: unusual behaviors are not necessarily attacks. As a result, anomaly detection techniques generally suffer from a high rate of false positives, which impact their practical deployment.

Recently, fine-grained taint-tracking has become popular in software vulnerability defense. Its strength lies in its ability to reason about the degree of control exercised by an attacker on data values within the memory space of a vulnerable program. This enables the development of security policies that can, with high confidence, detect dangerous uses of such "tainted" data in security-critical operations. This technique is capable of defeating a wide range of attacks, including code injection, command injection and cross-site scripting[1]. Its main drawback is the requirement for manual policy development, which can be hard for some classes of attacks, e.g., non-control data attacks[2] and directory traversals.

We propose a new taint-based approach in this paper that avoids the need for policies by leveraging an anomaly detector. By targeting the anomaly detector on tainted data and/or events, our approach can avoid a large fraction of false positives that occur due to benign anomalies, i.e., behavioral deviations that aren't under the attacker's control.

* This research was supported in part by an NSF grant CNS-0627687, and performed while the first author was a PhD student from Università degli Studi di Milano, Italy visiting Stony Brook University.

[1] See, for instance, XU, BHATKAR and SEKAR, *"Taint-enhanced Policy Enforcement: a Practical Approach to Defeat a Wide Range of Attacks,"* USENIX Security Symposium, 2006.

[2] These attacks corrupt security-critical data without subverting control-flow. Chen et al ("*Non-Control-Data Attacks Are Realistic Threats,*" USENIX Security Symposium, 2005) showed that they can achieve the same results as code injection attacks, while evading many code injection defenses.

## 2 Anomalous Taint Detection

Our starting point is a system-call based program behavior model, e.g., the one used by Forrest et al ("A Sense of Self for Unix Processes," IEEE Security and Privacy '96). We enhance this model with information about system call arguments and taint. As in Bhatkar et al ("Dataflow Anomaly Detection," IEEE Security and Privacy '06), this learning technique leverages the control-flow context provided by system-call models.

Our technique learns information about system calls (or other interesting functions) and their arguments at multiple granularities. At a coarse granularity, it learns whether an event's argument is tainted. At a finer granularity, it learns whether structure fields (or array elements) are tainted. Furthermore, we also generate application-specific taint-enhanced profiles, such as expected maximum and minimum argument lengths, structural inference with character class mapping, and longest common prefix models.

We briefly illustrate our technique[3] using a format-string vulnerability in the `WU-FTPD` program. This program elevates its privileges temporarily, and then uses the following code snippet to revert its privilege to that of a normal user. Chen et al demonstrated a non-control data attack that overwrites `pw->pw_uid` field with zero. As a result, the server does not revert to user privilege.

```
1  FILE *getdatasock(...) {
2     ...
3     seteuid(0);
4     setsockopt(...);
5     ...
6     seteuid(pw->pw_uid);
7     ...
8  }
```

Our approach can detect this attacks in two ways. First, the attack causes this `seteuid`'s argument to be tainted, whereas the argument is untainted under normal operation. Second, the attack causes deviations in the structure of a (tainted) argument to a `printf`-like function. While the latter method is tied to the specifics of the underlying vulnerability, the former technique is able to detect the effect of corruptions that may be caused by other vulnerabilities as well.

By leveraging on taint information, our approach is less vulnerable to mimicry-like attacks than, for instance, a learning-based anomaly detection approach which relies *only* on statistical properties of the observed data, e.g., Mutz et al ("Anomalous System Call Detection," ACM TISSEC 2006). With a purely learning-based approach, if a limited number of authenticated users were observed during training, then a mimicry attack would be possible that may allow an attacker to impersonate any one of these users.

We have been able to detect other non-control data attacks described by Chen et al using models that reason about the structure and/or lengths of tainted arguments. Our future work is aimed at (a) extending the technique to work on other attack types that require application-specific taint policies (e.g., directory traversals), and (b) deriving taint policies from the taint-enhanced behavioral models that can provide the basis for preventing (rather than just detecting) exploits.

[3] Addtional details can be found in CAVALLARO AND SEKAR, *"Anomalous Taint Detection"*, Tech Report SECLAB08-06 at `http://seclab.cs.sunysb.edu/pubs.html`.