Learning from a Flaw in a Naive-Bayes Masquerade Detector

Kevin S. Killourhy Roy A. Maxion
Dependable Systems Laboratory
Computer Science Department
Carnegie Mellon University
Pittsburgh, Pennsylvania 15213 / USA

Abstract

As security professionals relegate more and more security-critical tasks to machine-learning algorithms, there is a growing risk that trust in automation will overlook a critical flaw from time to time. We share our experience in having discovered a flaw that enabled an attacker to masquerade as another user without detection. We describe how we came to suspect a vulnerability in a machine-learning-based detection algorithm, how we confirmed the existence of the vulnerability through rigorous and controlled experimentation, and how we came to fortify the detection algorithm against certain adversarial behaviors. We illustrate a danger faced by machine learning in the computer-security domain, and we discuss the strategies we found useful when investigating that danger. We hope that the lessons we learned will be useful to other researchers using machine learning in adversarial environments.

1 Introduction

A masquerader is someone who gains access to another person's account, and pretends to be that person, perhaps accessing sensitive information or emails, or modifying critical files. Masquerade-detection systems usually work by learning the normal behavior of a potential victim, and by identifying an interloper's masquerading behavior as anomalous with respect to the victim's normal behavior. We investigated the effectiveness of the naive-Bayes machine-learning algorithm in detecting differences between, for example, the UNIX commands typed by different users, where a detector might flag one user's preference for `emacs` over another user's `vi`.

A curiosity arose during a comparative evaluation of naive-Bayes' performance on two types of command-line input: truncated (containing just UNIX command names) and enriched (including flags and other arguments as well) [3]. We observed that certain masqueraders always escaped detection; we called these *super-masqueraders*. From a machine-learning perspective, these super-masqueraders were a bit of a mystery. From a security perspective, they were a significant concern, since such masqueraders might be capable of evading detection at will.

2 Related work

Several machine-learning algorithms predated the use of naive Bayes in masquerade detection. Lane and Brodley [2] developed an algorithm to learn the "characteristic sequences" of a user's commands, and to alarm upon seeing uncharacteristic sequences. Schonlau et al. [7] evaluated six different machine-learning algorithms for detecting masqueraders in UNIX command-line data. Maxion and Townsend [4] introduced naive Bayes, and compared it to these earlier algorithms. Each of these investigations measured performance in aggregate terms (e.g., miss and false-alarm rates). They did not consider whether an adversarial masquerader could thwart the detector and evade detection.

The machine-learning literature has considered some general conditions under which naive Bayes might underperform. Mitchell [5], in a standard reference for naive Bayes, described how an m-estimate, or pseudocount, could augment the algorithm to prevent very rare events from having undue influence. Domingos and Pazzani [1] showed that naive Bayes performed well, even when its assumptions about the conditional independence of the data were violated. Rish et al. [6] demonstrated that data entropy is a good predictor of naive-Bayes performance. While masquerade detection has benefitted from these insights, conditions arose in our domain that were not addressed by general machine-learning analyses. For instance, our domain has adversarial masqueraders who can adapt their behavior (i.e., their own commands) to escape detection.

3 Experiment summary

Our problem was: to discover what enabled the super-masqueraders to evade naive Bayes; to explain why the detector was failing; and to fortify it against such failures.

Examining the super-masqueraders. To determine the root of the super-masquerader phenomenon, we performed an error analysis on previous experimental results obtained by Maxion [3]. We examined the commands used by each of the two super-masqueraders (out of 50), and noticed that they both used rather eccentric commands. For instance, both super-masqueraders used a peculiar invocation of `rlogin` that none of their victims ever used; and these were commands that naive Bayes had never seen in any victim data. Based on this observation, we hypothesized that never-before-seen commands (NBSCs) suppress alarms by naive Bayes. When a block of commands is typed into a user's account (either by the account owner or by a masquerader), naive-Bayes reviews the similarity of those commands to those previously typed by the account owner, and to those typed by other users on the same system. Commands similar to the account owner's commands will not cause an alarm; commands similar to other users' commands will cause an alarm. Our suspicion was that NBSCs would not raise alarms.

Explaining the NBSC effect. To explain why NBSCs cause detector failure, we ran naive Bayes on a series of carefully constructed, synthetic data sets corresponding to a range of factors that could affect naive Bayes' decisions about a block of commands. Examples of these factors are the number of NBSCs, the similarity of the rest of the block to commands typed by the account owner and by other users, the number of other users, etc. For each of these factors, we identified a range of possible values (e.g., from least-similar to most-similar), and synthesized 13,200 data sets covering all combinations of factors. Running naive Bayes on these data sets confirmed that, regardless of other factors, increasing the number of NBSCs decreased the rate at which naive Bayes alarmed. Having demonstrated the effect of NBSCs, we were able to analyze the detector's internal decision procedure to understand the mechanism by which NBSCs lowered the likelihood of an alarm (the naive Bayes score equation is biased against alarming).

Fortifying the algorithm. It is easy for a masquerader to introduce NBSCs at the UNIX command line by creating aliases or symlinks with strange names, turning common commands into NBSCs (e.g., using a nonsense string like `hzeowco` as an alias for `rm`). To respond to this threat, we implemented an NBSC detector that could operate in conjunction with naive Bayes. The result was a 35% decrease in overall cost of error using the combined detector as opposed to using standard naive Bayes (cost measured as the sum of the miss and false-alarm rates). More importantly, the previously noted super-masqueraders were easily detected, and no other masquerader was able to evade detection more than 34% of the time (which is still a concern but substantially better than 100% evasion).

4 Lessons learned

The error of a machine-learning algorithm is often expressed as an average (e.g., a miss rate). Such aggregate measures of performance hide the fact that some missed events are more important than others. While Maxion found that using enriched commands produced better average results than using truncated commands [3], we would not recommend using an unfortified naive-Bayes detector in that environment. The importance of average miss rates is diminished when adversaries can induce misses at will. Evaluations must move beyond performance averages, since they omit an important part of the story. Even worst-case estimates are insufficient, because domain knowledge is required to see if a worst-case situation arises by chance, or through masqueraders' manipulations.

We attribute our success in this investigation to a thorough analysis of classifier errors, coupled with domain knowledge. Chronic failures, induced by phenomena like the super-masquerader, were evidence of an underlying vulnerability, but the same could be true of any classification error. Only error analysis can determine whether an error represents an exploitable flaw. Domain knowledge helps to determine whether a theoretical vulnerability can be exploited in practice. For instance, while NBSCs are easy to introduce at the UNIX command line, if our detector were analyzing system calls (where there is a fixed, known set of possible values), an adversary could not easily introduce never-before-seen system calls.

While domain knowledge is an important investigatory tool, common problems across domains may have common solutions. For instance, some years ago much of the spam evading our spam filters contained nonsense strings that caused Bayesian spam filters to fail. Modern spam filters have been adapted to detect this attack, and a similar adaptation might also fortify Bayesian masquerade detectors against NBSCs.

Our experience suggests a useful way to integrate synthetic-data experiments into computer-security research. We found synthetic data to be invaluable in exploring detector performance under conditions that are rare in practice. When the detector failed under rare conditions, we applied domain experience to determine whether an adversary could fabricate the conditions responsible for evading detection.

5 Summary

We described our experience uncovering flaws in an application of machine learning to masquerade detection. We addressed that flaw through deep analysis of classifier errors, domain knowledge, and controlled experiments with synthetic data. We hope that these experiences illustrate the challenges of machine learning for other security researchers, and that the lessons we learned prove as useful to them as they were for us.

References

[1] Pedro Domingos and Michael Pazzani. On the optimality of the simple Bayesian classifi er under zero-one loss. *Machine Learning*, 29:103–130, 1997.

[2] Terran Lane and Carla E. Brodley. Temporal sequence learning and data reduction for anomaly detection. *ACM Transactions on Information and System Security*, 2(3):295–331, Aug 1999.

[3] Roy A. Maxion. Masquerade detection using enriched command lines. In *International Conference on Dependable Systems and Networks (DSN-03)*, pages 5–14, Los Alamitos, CA, 22-25 June 2003. IEEE Computer Society Press. San Francisco, CA.

[4] Roy A. Maxion and Tahlia N. Townsend. Masquerade detection using truncated command lines. In *International Conference on Dependable Systems and Networks (DSN-02)*, pages 219–228, Los Alamitos, CA, 23-26 June 2002. IEEE Computer Society Press. Washington, D.C.

[5] Tom Mitchell. *Machine Learning*. McGraw-Hill, 1997.

[6] Irina Rish, Joseph Hellerstein, and Jayram Thathachar. An analysis of data characteristics that affect naive Bayes performance. Technical report, IBM T.J. Watson Research Center, 30 Saw Mill River Road, Hawthorne, NY 10532, 2001.

[7] Matthias Schonlau, William DuMouchel, Wen-Hua Ju, Alan F. Karr, Martin Theus, and Yehuda Vardi. Computer intrusion: Detecting masquerades. *Statistical Science*, 16(1):58–74, Feb 2001.