Guarding the source code against inside attacks: a case study of user access patterns

Li Sun, Serdar Boztas, Kathy Horadam and Asha Rao
RMIT University
Australia
li.sun@rmit.edu.au

Abstract

For software companies, the source code and the people are the most valuable assets. Employees are responsible for developing the source code, but paradoxically, may also pose the greatest potential threat. A single disgruntled employee can cause untold damage, by for example selling the source code to a competitor or inserting a malicious backdoor. Detecting and preventing inside attacks is an unsolved problem. Our proposed approach is to detect when a software engineer deviates from their usual work patterns and then match these anomalies against signatures of inside attacks. The first step is to establish a baseline of the normal access patterns of software engineers to source code. To this end we consider a case study of a commercial software project, analysing how 282 software engineers access the source code in their daily work over a period of three years. We obtained 3 years of access logs to a real SVN source code repository by software engineers at a commercial IT company (CA Technologies). In the study, we used the DEX graph database management system to store data and used a visual clustering algorithm (VAT) to analyse user access behaviour.

From the visual clustering, communities of users are grouped together based on the projects they access and the networks they use. It is also noted that there are different daily access patterns between users with different roles in the enterprise. This suggests that the role of the user could be used as a key attribute of the user profile to be used in an insider threat detection system. This preliminary study will provide a good baseline for an insider threat detection system.

Keywords

Insider threat, Source code access management, SVN log analysis, User source code access behaviour, Visual clustering algorithm
Guarding source code against inside attacks: a case study of user access patterns

Li Sun, Serdar Boztas, Kathy Horadam and Asha Rao
School of Mathematical and Geospatial Sciences,
RMIT University

ASWEC 2013
Melbourne, 4th June

Overview

• Project motivation and aims
• Approach to detect insider attacks
• A case study within CA
• Conclusion
• Further work
Project motivation and aims

• Insider attack to the source code is a serious risk faced by software companies
  – Source code is the most valuable asset
  – Insider attacks are rare but cause disproportionately large damage to the company
• Ernst & Young Global Information Security Survey:
  – 64% of organisations rated leaking of sensitive data as a top 5 IT risk
  – 50% of organisations planning to increase their spending on DLP
• Types of insider attacks on the source code
  – IT sabotage
  – Theft of IP
• Project aim:
  – Devise a solution which can detect and prevent insider attacks, based on the real-time analysis of log data

Approach to detect insider attacks

• Identify high risk assets
• Obtain data on usage patterns from a wide range of sources
• Model legitimate behaviour of users
  – General models
  – Individual models
• Model different types of insider attacks
• An attack is detected when users’ behaviour deviates from normal patterns and correlates against patterns seen in threat models
A current study within CA: source code repository

- Access logs to the SVN repository for a CA product
- 700 million lines of access logs
- Covering a period of 3 years 2008-2011
- 282 users
- 3 million files accessed
- Data anonymised with respect to usernames, filenames, hostnames and IP addresses

SVN log analysis

1. SVN log
2. Log parser
3. Build the graph database (DEX)
4. Query the database
5. Query result analysis
6. VAT: virtualize the user access pattern
Low level SVN log

- 1.1.1.1 - bot1 [06/Mar/2011:06:39:19 +1100]
  "PROPFIND /repos/a/trunk/modules/b HTTP/1.1" 207
  363 ""SVN/1.4.6 (r28521) neon/0.25.5"
  - IP address of the client (machine)
  - Userid of the person requesting the document
  - The time that the request was received
  - The request line from the client (action & resource)
  - The status code
  - The size of the object returned to the client
  - The client browser

Data management (DEX)

- DB stats:
  - 700M nodes
  - 3500M edges
  - 313GB of data
Extracting higher level commands

- Aim: extract user actions (e.g. checkout, commit) from log lines
- A single user action may generate 100’s of lines of logs
- Reconstructing user actions is ambiguous
- Approach:
  - Reverse engineer user commands by testing on a sample repository
  - Group closely timed log lines from same user as “commands”
  - Classify as “read” or “write” commands
  - Resource = repository + project + file + version + tag
- New database:
  - 700 million logs -> 9 million commands (27k writes)
  - 1.9 million resources -> 246k files -> 223 projects

High level command database

- DB stats:
  - 6 GB of data
  - 281 Users
  - 8987666 Commands
  - 2 repositories
  - 223 projects
Visualizing the user access patterns

- VAT: A Tool for Visual Assessment of (Cluster) Tendency
  - NxN dissimilarity matrix
    - \( D = 1 - \frac{(U_1 \cap U_2)}{(U_1 \cup U_2)} \)
  - Other effective dissimilarity measures
  - Re-order
    - Randomly pick one object, e.g., \( U_3 \)
    - Find its closest object, \( U_2 \)
      - \( D_{12} > D_{43} > D_{23} \)
    - And so on, get \( U_4 \)
    - \( D_{12} > D_{42} \)
  - New order: \( U_3, U_2, U_4, U_1 \)
- Dissimilarity value
  - 0: least dissimilar, identical items (black)
  - 1: most dissimilar, totally different items (white)

Finding 1: User communities by accessed project
Finding 2: Different types of users’ daily access patterns

Software architect

Forward developer

Sustaining developer

Build script

Conclusion

• User communities based on access patterns do exist.
  – Build models of normal user behaviour, models of community
    behaviour in an insider threat detection system.

• Users with different roles have different types of access
  patterns.
  – Build models of role behaviour in an insider threat detection
    system.
  – Manage employees’ access to their source code
    • by role
    • by communities
Future work

• Further analysis of the source code repository user access pattern
  – Alternative distance functions – e.g. weighting by frequency
  – Improvements to event extraction algorithms
  – Explore different user access patterns

• Apply user access patterns in source code management
  – Build the baseline model of the insider threat detection system
    • An attack is detected when users’ behaviour deviates from normal patterns and correlates against threat models
    – Identify the user communities and fine tune their access privileges.

• Explore user access patterns on other datasets

Acknowledgments

• This research was supported and sponsored by CA Labs, the research division of CA Technologies.

• The project was funded by Australian Research Council (ARC) under an ARC linkage grant at the School of Mathematical and Geospatial Sciences, RMIT University.
Thanks

• Any questions?