Pragmatic Security

Marc Donner
2008 October 23
V3
(for presentation to ISTS Dartmouth)

Numerous commentators through history have noted that brute strength, as in a fortress with thick high walls, is never a reliable guarantor of security. In this talk I will take you through some experiences from years of operational practice in industry, experiences of mine and of various others, that illustrate some of the pragmatic challenges in security. I will attempt to demonstrate to you that dramatic improvements can be achieved with simple expedients and why many of the powerful techniques that we study are destined to be ineffective.
Problem: Subway Graffiti
During the 1970s and 1980s the New York City transit system was afflicted by an epidemic of rail car defacement. Young vandals painted the insides and outsides of the cars. The city tried all sorts of tricks, including draconian restrictions on the sale of spray paint, severe punishment for the offenders they caught, and redesigning the interiors of the cars so that paint wouldn’t stick. Nothing worked and it looked like the city was doomed to destruction. That is, until someone studied the behavior of the vandals and discovered that virtually all of the damage was being done in the rail yards where the trains were stored when not in service. A little fencing, and a little patrolling and the problem was history.
Two Shaggy Dog Stories

- Trading floor security
- Access control hygiene
Context:
- Early 1990s
- 600 desktop computers
- <50,000 square feet (50,000 Square Feet is less than 100 square feet per person. An 8x6 desk takes up 48 square feet, or more than half)
- Up to four 21” screens per desk

The foreground station has seven screens. Notice the three blind mice.

Trading day – 7:30 AM to about 4:30 PM
Crowded, busy, and noisy on the trading floor during the day ...
completely dead at night
Trading Desk
Early 1990s Trading Floor

- 600 desktop computers
- less than 100 square feet per person
- Up to four 21” screens per desk for real-time data
- Most important transactions over the phone
- Trading day – 7:30 AM to about 4:30 PM
  – Many markets open, systems active 24 x 7
  – “Follow the sun” trading activity
- Crowded, busy, and noisy on the trading floor during the day, dead at night
What I Found On The Trading Floor

- 80% of passwords succumbed to crack
- 50% of accounts had the same password
- A tour of the trading floor produced hundreds of post-it notes with passwords on them

- System uptime was fabulous – time between reboots was commonly nine to twelve months!
  - Users didn’t know their passwords
  - Floor support didn’t have root password

- No one cared about security
Security incident – market open, floor empty, “someone” enters a bad trade. Unwinding the bad trade cost many millions.

[For instance – the Mizuho securities bad trade http://news.bbc.co.uk/2/hi/business/4512962.stm in January of 2005 (not a security incident, but it shows that a bad trade can be catastrophic) ... meant to sell one share at 61,000 Yen (about $5K) , instead sold 61,000 shares at 1 Yen (about one cent) for a loss of $300 million. This happened because the conventional data entry order in Japan is < security, quantity, price> while in the US it is <security, price, quantity> (or vice versa) ... the trader involved was Japanese but visiting the US and using a US system. The system prompted him twice that he was making a questionable trade and demanding confirmation that he wanted to proceed. He clicked on the proceed button twice despite the warnings.]
Know Thy Password

• Need it to log on
  – ... once per year
• How about a screen lock?
  – Make them use their password regularly
  – Secure the screen when use is away
• Voluntary screen lock exhortations?
  – LOL
OK, xlock and xautolock

- Xlock locks the screen
- Xautolock fires up xlock when the keyboard and mouse are idle for too long
- But but but
  - Market data watching is main use ... little or no keyboard or mouse activity
  - Unlocking everytime the user has been idle for more than ten minutes is onerous
Solutions

• Introduce the notion of “local business day” as a configuration file in the system.
  – Once xlock is unlocked during the local business day, it stays unlocked for the rest of the day.

• Make xlock’s screen blanker be transparent
  – Thus only really locking the keyboard.
Trojan Horse, Anyone?

- One last flaw in xlock ...
  - Unlock by logged-on user OR by root
  - ... hmm ...
    - Floor support staff do not have root password

- OK, change one line of code and instead of root, we expect the password of the “xlock” user
Final Rollout

• Memo to all trading floor staff:
  – Misuse of your account hits YOUR P&L
  – It’s from the head of trading, not from some security dweeb (me)
  – We’ll be implementing automatic screen locking next week

• Xautolock becomes part of standard user profile (we had standardized them already)

• No user complaints 😊
Elapsed Time

• Educating senior trading floor management and negotiating memo
  – Six months
• Fixing up xlock
  – One week
• Writing and distributing the memo
  – Two weeks
• Completion of rollout of xlock to all desktops
  – Three weeks
Mainframe Access Control Hygiene

In the 1990s I was running a strategic planning exercise to figure out what to do with a major legacy back office system.

The head of system security came to talk to me. He told me that Internal and External Audit had just informed him that they would no longer audit the access control system on the mainframe.
Why Was He Terrified?
The Policy-Implementation Feedback Loop

- Application
- Authorization Database
- Audit
- Policy Statement
- Security Administration
- Enterprise Managers
Application Authority

- `<UserID, AccountRange, Privilege>`
  - Where `AccountRange` was literally a range of integers
  - Where `Privilege` was one of some large number of functions (methods, sort of) in some application

- The permissions database now had nearly 700,000 assertions (rows)
  - Governing about 2,000 employees
The Old Audit Procedure

• Select a few hundred assertions
• Analyze them for reasonableness
• If they passed, certify the entire 700,000
• If they failed, fix them and certify the entire 700,000

• What’s wrong with this approach?
First Try

- Hire some fancy security consultants
- Give them the statement of the problem
- Pay them a bunch of money
- Get a pretty report back that says ...

- No can do.
Next Try

• Role-Based Access Control
  – That’s what all the researchers say is the right answer

• First problem:
  – Can’t buy a Role-Based Access Control System

• Second problem:
  – There’s no empirical evidence that RBAC will actually work or is actually better
Atoms were the raw assertions from the original authorization database.

Molecules are the < AssetGroup, UserGroup, PrivilegeGroup > tuples here.
Second Problem

• We mined the existing permission database looking for structure that indicated the existence of roles.

• (Paper: Donner, Nochlin, Shasha, and Walasek – “Algorithms and Experience in Increasing the Intelligibility and Hygiene of Access Control in Large Organizations”)
It really ought to be concave down. Why isn’t it? Well, the algorithm that we used was lossless, so any noise in the setup of the original database was lovingly preserved in the RBAC model we would create. Thus two people with the same role whose permissions had diverged in error would be kept distinct over time.
Remaining Open Questions

• Can we remove noise from the system and get the downward concavity we expect?
• Can we find an RBAC vendor?
• When we started building an RBAC system for production, there was a big push to make the asset descriptions dynamic – pass by name and evaluate at run time. Is this necessary?
• Have we actually eased the Audit Problem?