Crisis-Tested Data Controls

Lessons from Major Breaches

Chris Burroughs

August 14, 2025

Agenda

- **Reality Check**
- **Case Studies**
  - NotPetya
  - Equifax
  - Scatter Spider
  - Solarwinds
  - Sony
- **The New Security Paradigm**
- **Your Action Plan**

About me

The Harsh Reality Check

Assume breach

Plan for breach

Build for breach

NotPetya – When Trust Becomes Your Weakness

- Attackers compromised MEDoc
- Used it to push malicious updates to 12,000+ companies
- Combined multiple exploits: EternalBlue, Mimikatz, credential harvesting
- Within hours, it spread globally through legitimate business connections

This wasn't about patching faster or better firewalls. This was about TRUST VALIDATION

🔑 Key Takeaways

- **Zero Trust Your Software Updates**
  - Implement code signing verification for ALL software updates
  - Create isolated update testing environments

How many of you can guarantee that every piece of software in your environment came from a verified, trusted source?

And how long would it take you to recover if you lost everything?

  - If NotPetya taught us anything, it's that lateral movement kills
- **Rethink your recovery architecture**
  - Deploy immutable, air-gapped backups
  - Take offline snapshots of AD
  - Calculate your catastrophic RPO - Can you rebuild from zero?
  - Create workstation golden images and software install libraries

Equifax - The Asset You Didn't Know You Had

- 147 million people impacted due to one unpatched vulnerability.
- Apache Struts vulnerability announced March 7, 2017
- Equifax identified the vulnerability in their systems
- Breach occurred May-July 2017
- The real problem: They found the vulnerability in some systems, but missed this one

You can't protect assets you don't know exist.

🔑 Key Takeaways

- **Asset Discovery is Your Foundation**
  - You can't protect what you don't know exists
  - Implement automated asset discovery tools
  - Map data flows, not just network connections
  - Track application components: JVM versions, libraries like Log4j and Struts
  - Manage application lifecycle religiously - know what's end-of-life

Can you tell me every system that has access to your most sensitive data? Not just the ones that supposed to have access - the ones that actually do?

- **Data Minimization is Your Best Friend**
  - Why did they have 40 years of credit data online?
  - Archive what you don't need
  - Encrypt everything else with different keys

SCATTERED SPIDER - THE HUMAN FIREWALL FAILS

- Scattered Spider used social engineering to compromise employee accounts
- Bypassed MFA through SIM swapping and social engineering
- Moved laterally through cloud environments
- Accessed sensitive customer and employee data

Your identity verification processes were designed for a world where attackers didn't have access to personal information, voice synthesis, and professional social engineering teams.

Key Takeaways

- **Identity is the New Perimeter**
  - Implement behavioral biometrics
  - Monitor for impossible travel scenarios
  - Use risk-based authentication
  - Restrict service accounts to non-interactive and monitor. Change when someone with elevated privileges leaves
- **For real - implement least privilege**
  - Restrict admins to only those systems they need

Your help desk can probably reset the agency head's password with a phone call and some basic personal information. Think about that.

- **Social Engineering Has Gone Professional**
  - Train for specific attack scenarios, not generic phishing
  - Implement out-of-band verification for sensitive requests
  - Create "code words" for financial or data access requests
- **Zero Trust Your Help Desk**
  - Most dangerous entry point in modern organizations
  - Implement strict identity verification protocols
  - Record and review all password reset requests

SOLARWINDS - THE SUPPLY CHAIN NIGHTMARE

- Attackers compromised SolarWinds' build environment
- Injected malicious code into legitimate software updates
- Distributed malware to 18,000+ customers including Fortune 500 and government agencies
- Maintained persistence for 9+ months

How do you implement security when the security tools themselves can't be trusted?

🔑 Key Takeaways

1. **Vendor Risk ≠ Product Risk**
   - Audit vendor security practices, not just compliance certificates
   - Monitor vendor security incidents that might affect your environment
2. **Behavioral Analysis for Software, Not Just Users**

The more integrated and automated your security stack becomes, the more vulnerable you are to supply chain attacks. How do you balance efficiency with resilience?

   - Practice isolating affected vendor tools without breaking operations
   - Maintain alternative tools/processes for critical functions

SONY - THE PLAINTEXT PASSWORD CATASTROPHE

- Lulzsec hacking group accessed systems through a common SQL injection attack
- Over one million customer passwords were unencrypted
- Personal data of 77 million PlayStation Network users compromised
- Services down for 23 days
- Cost: Over $171 million

Your Code IS Your Security Control

🔑 Key Takeaways

- **Secure Coding Isn't Optional Anymore**
  - Every developer needs security training, not just the "security team"
  - Implement secure coding standards and enforce them
  - Code reviews must include security checks
- **Defense in Depth for Data Storage**

Sony's breach could have been minimized by implementing password hashing - a technique that was already 20 years old at the time.

  - Design data storage assuming complete database compromise
  - Separate authentication data from personal data
  - Use different encryption keys for different data types

New Security Paradigm

Data-Centric Security Architecture

- Classify data based on business impact and regulatory requirements
- Encrypt data at rest, in transit, and in processing
- Implement DLP that actually works

Assume Breach Design Principles

- Design systems assuming internal compromise from day one
- Implement network micro segmentation
- Deploy deception technology, like honeypots, strategically

Behavioral Analytics Over Rule-Based Detection

- Monitor for deviation from established normal patterns
- Focus on data access behaviors, not just network traffic
- Implement conditional access policies based on risk context

Identity-First Security Controls

- Treat identity verification as your primary security boundary
- Implement passwordless authentication (Okta Verify, Windows Hello, etc.)
- **SMS MFA is not secure** - upgrade to app-based or hardware tokens
- Monitor for identity compromise indicators

Continuous Validation

- Verify trust continuously, not just at login
- Implement adaptive access controls
- Monitor for privilege escalation attempts

Supply Chain Security

- Implement software bill of materials
- Monitor vendor security postures continuously
- Plan for vendor compromise scenarios

Action Plan

Week 1

- Map your crown jewel data
- Inventory your software stack
- Audit your software supply chain
- Review cloud data

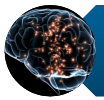
Month 1

- Implement data classification and handling procedures
- Review your permissions using principle of least privilege
- Review and update help desk identity verification procedures
- Test your incident response

Month 3

- Architect network segmentation project
- Implement CyberArk for privileged accounts and use a key vault
- Assess vendor security practices beyond compliance certificates
- Conduct tabletop exercises based on these specific breach scenarios

The Non-Negotiables


Know your data

Assume compromise


Monitor everything


Verify constantly


Plan for failure

Secure your supply chain


Manage your configurations

Closing Challenge


odga
Office of Data
Governance
and Analytics