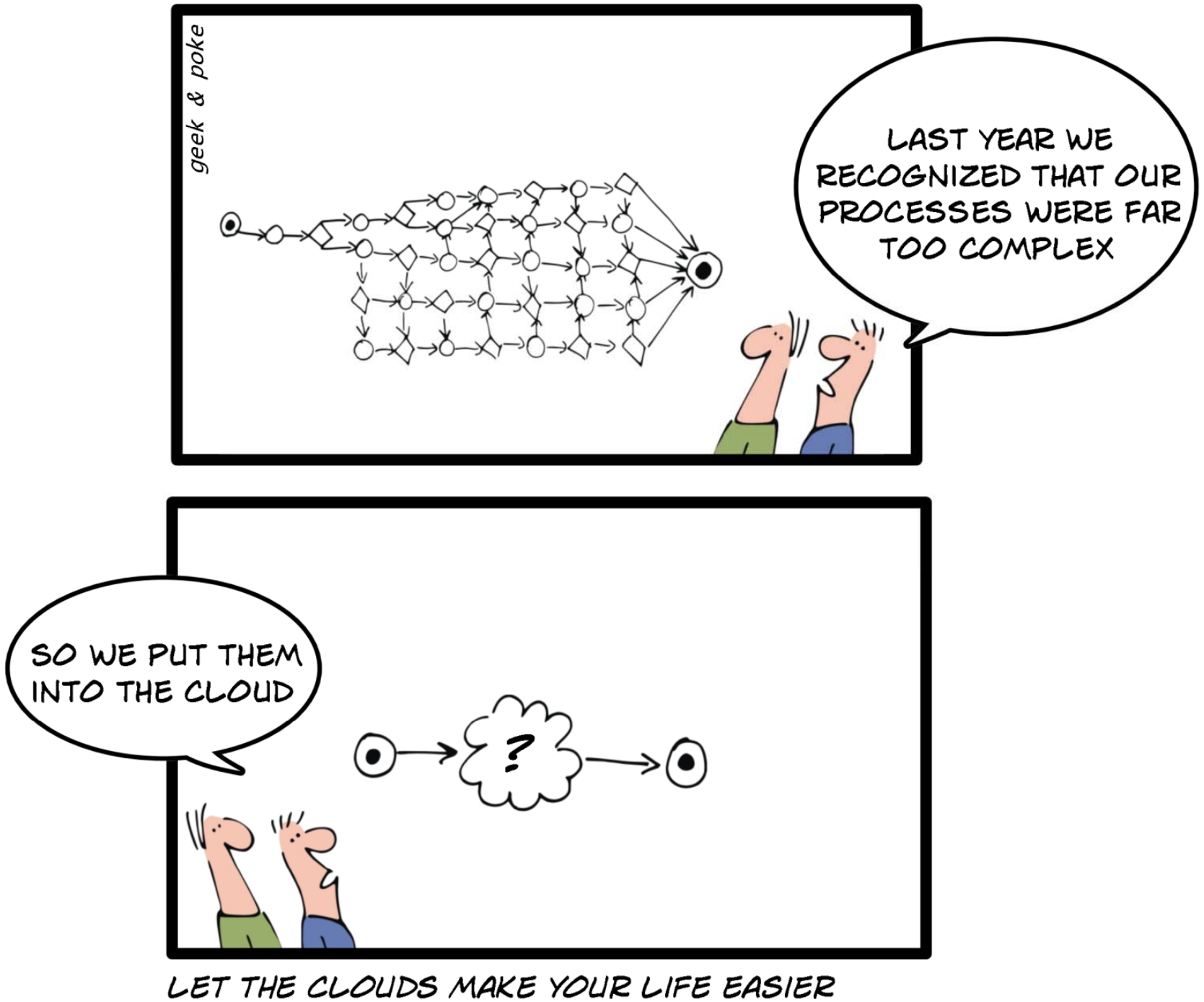


LET THE CLOUDS MAKE YOUR LIFE EASIER

Enterprise Encryption 101

Phil Smith III

Voltage Security, Inc.

August 2010

Copyright Information

Agenda

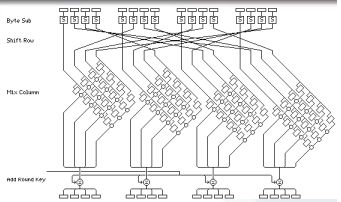
- Why we're here
- Why encryption is difficult and scary
- The five Ws of encryption
- Encryption key management: the "other" gotcha
- A realistic approach to enterprise encryption
- Example: Voltage SecureData

Why We're Here

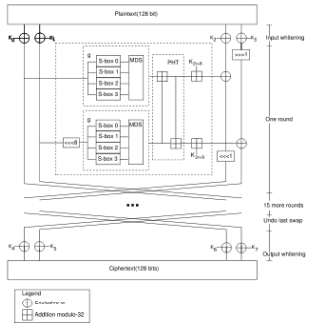
- Encryption is on many folks' minds these days
  - C*x*Os, CISOs are saying "Gotta encrypt stuff ***now!***"
- Breaches are in the news
  - Heartland, TJX, RBS WorldPay, et al.
- Many sites have implemented several point solutions
  - Different platforms, different problems…not interoperable!
- DLP (data leakage prevention) is not foolproof
  - If it's leaked but encrypted, you care a ***whole lot*** less!
- The h4xx0rs are out there…
  - …and they're getting smarter and more creative
- Internal breaches are increasing
  - Gartner et al. agree: 70%++ breaches are internal

Enterprise Encryption In Sixty Minutes

Encryption Is Difficult

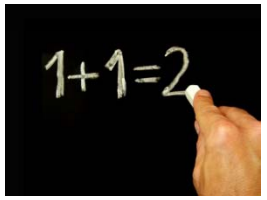


- Lots of different technologies
  - Hardware-based, software-based, hardware-assisted
  - DES, TDES, AES, Blowfish, Twofish, CAST, PGP, GPG … !
- Companies have ***lots*** of data in ***lots*** of places
  - Much of it probably of unknown value/use
  - The sheer volume is daunting
- Difficult to imagine how to get started
  - Easier to stick your head in the sand and hope it goes away
- For mainframe folks, it's even easier to (try to) ignore
  - System z OSes are traditionally more secure than distributed



Encryption Is Scary

- Most of us don't understand the technologies
  - Math classes were a looong time ago
- It changes constantly
  - We hear "DES has been broken, use AES"
  - What does that mean? Is DES useless? Is AES next to fall?
- Lots of snake-oil salesmen in encryption
  - www.meganet.com touts "unbreakable encryption"
- Easy to decide encryption is unapproachably complex
  - Like buying your first house, or doing your own taxes…



IRS
Department of the Treasury
Internal Revenue Service

The Five Ws of Encryption

- ***Why*** encrypt data?
- ***What*** should be encrypted?
- ***Where*** should it be encrypted?
- ***When*** should it be encrypted?
- ***Who*** should be able to encrypt/decrypt?
- ***How*** will you encrypt it?

Why Encrypt?

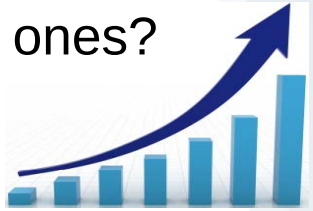
- Every company has data to protect
  - NPPI, PII, or just PI
  - Customer information
  - Internal account information
  - Intellectual property
  - Financial data
- Every company moves data around
  - Backup tapes
  - Networks
  - Laptops
  - Flash drives
  - Data for test systems

Why Encrypt?

- Different media have different issues
  - Very few backup tapes get lost…but it does happen
  - Networks get compromised fairly regularly
  - Laptops are lost or stolen ***every day***
  - Flash drives are disposable nowadays
- Different media types mean different levels of risk
  - Deliberate, targeted network breaches are obvious concern
  - Missing backups ***probably*** won't be read
  - Missing laptops ***probably*** won't be analyzed for PII
  - Found flash drives are probably given to the kids

Why Encrypt?

- Breaches happen!
  - 2009: 498 2010: 315 to 6/15/2010 (Identity Theft Resource Center)
  - A healthy increase…and what about undetected/small ones?
  - Can you afford to bet your job/business?



- Data encryption is ***not*** a luxury
  - Claimed cost per compromised card is $154–$215!!!*
  - Heartland breach: 130M cards; TJX: 94M cards
  - Do the math…

* Source: Ponemon Institute
$154 = negligent inside
$215 = malicious/criminal act

Why Encrypt?

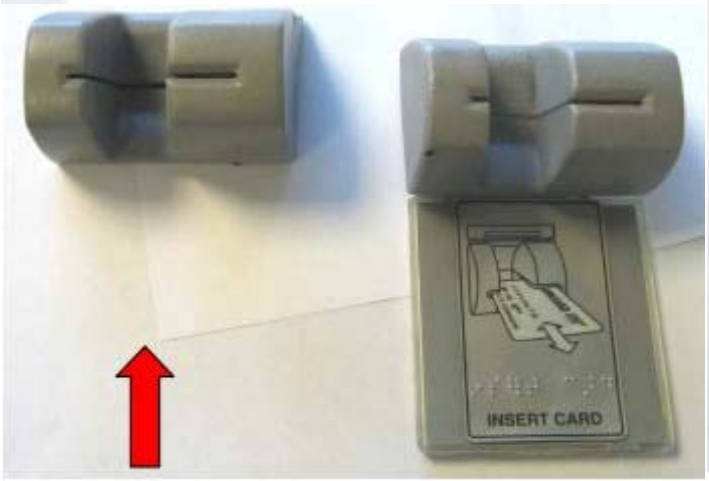
- Data breach sources:
  - 73%: external
  - 18%: insiders
  - 39%: business partners
  - 30%: multiple parties

Source: Verizon Business, 2009 Data Breach Investigations Report

- But insider breaches far more expensive:
  - External attack costs averages $57,000
  - Insider attacks average $2,700,000!

Why Encrypt?

- Commonalities:
  - 66%: victim unaware data was on system
  - 75%: not discovered by victim
  - 83%: not “highly difficult”
  - 85%: opportunistic
  - 87%: avoidable through “reasonable” controls
- Causes:
  - 62%: attributed to a “significant error”
  - 59%: from hacking or intrusions
  - 31%: used malicious code
  - 22%: exploited vulnerability
  - 15%: physical attacks



The real card reader slot.

The capture device

The side cut out is not visible when on the ATM.

Why Encrypt?

- The law is catching up with the reality
  - PCI DSS (Payment Card Industry Data Security Standard)
  - Red Flag Identity Theft Rules (FACTA)
  - GLBA (Gramm-Leach-Bliley Act)
  - SB1386 (California)
  - Directive 95/46/EC (EU)
  - HIPAA
  - etc.

- PCI DSS not only requires data encryption, but also:
  - Restrict cardholder data access by business need-to-know
  - This is called **separation of duties**

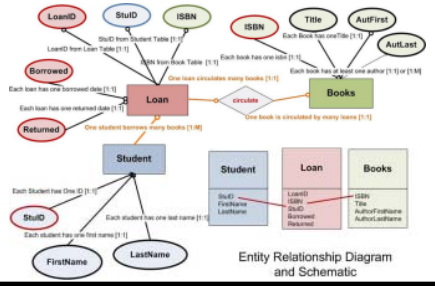
What To Encrypt?

- Everything! (Well, maybe not…)
  - Performance, usability, cost are barriers
  - Partners likely use different encryption technology
  - Changing ***every*** application that uses the data is prohibitive
- No single answer
  - Laptops, flash drives: at least PII, probably all data
  - Backup tapes: all data
  - Whole-database encryption possible but not a good answer

What To Encrypt?

- Whole database encryption fails on several counts
  - Can impose unacceptable performance penalty
  - Prevents data compression, using more disk space etc.
  - Violates separation of duties requirements
  - Better to just encrypt the PII (whatever that is)!

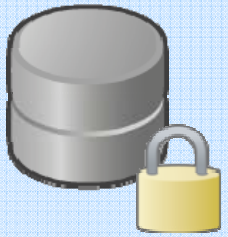
- What about referential integrity and other data relationships?
  - Database 1 & database 2 both use SSN as key
  - If you encrypt them, encrypted SSNs better match!
  - Else must decrypt every access, and indexes useless



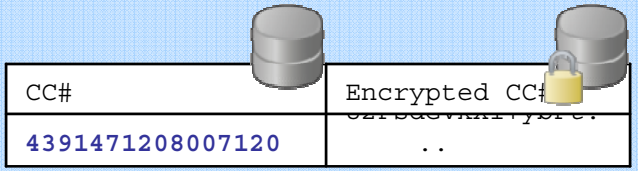
Entity Relationship Diagram and Schematic

Application & Database Encryption Today:
Four Approaches

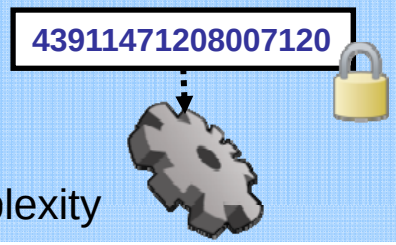
- Whole Database Encryption
  - Encrypt all data in DB—slows ***all*** applications
  - No granular access control, no separation of duties
  - No security of data ***within*** applications



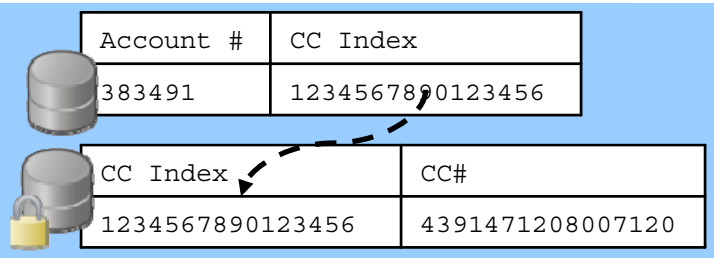
- Column Encryption Solutions
  - Encrypt data via DB API or stored procedure
  - Hundreds of tables and views, restricts change
  - No data masking support and poor separation of duties



- Traditional Application-level Encryption
  - Encrypt data itself via complex API
  - Requires DB schema/application format changes
  - High implementation cost plus key management complexity



- Lookaside Database (aka “Tokenization”)
  - CC# indexed, actual CC# in protected DB
  - Requires online lookup for ***every*** access
  - Can require major application redesign



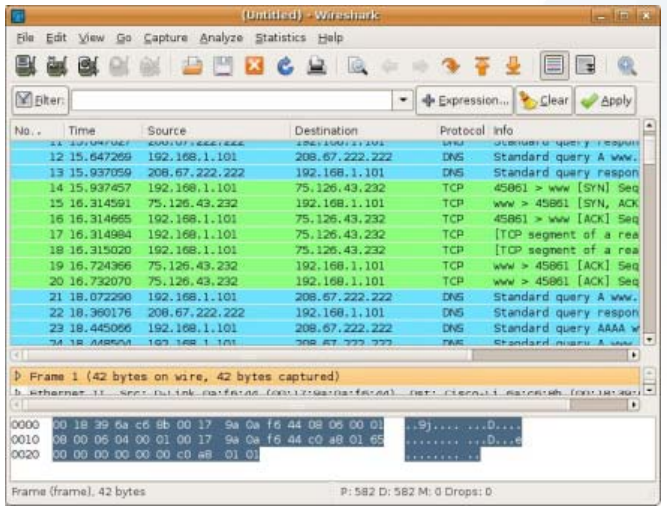
Voltage security

Where To Encrypt?

- Different question than “what”:
  - Data **at rest** and **in motion**
- Data at rest
  - “Brown, round, and spinning” (DASD of all types)
  - On tape (backup or otherwise)
- Data in motion
  - Traversing the network

Where To Encrypt?

- Data in motion particularly troublesome
  - How do you know if it's been sniffed as it went by?
- Data at rest ***somewhat*** easier
  - Intrusion detection systems fairly effective (if installed and configured, and if someone actually checks the logs)
  - ESMs very effective on z/OS (if administered correctly)
- Different issues, thus different criteria!



Voltage security

When To Encrypt?

- Ideally, data is encrypted as it's captured
  - By the data entry application, or the card swipe machine

- In reality, it's often done far downstream
  - The handheld the flight attendant just used—is it encrypting?
  - Did last night's restaurant encrypt your credit card number?
  - If the data goes over a wireless network, is it WEP? WPA?
- "Doing it right" is harder: more touchpoints
  - Easier (if less effective) to say "Just encrypt at the database"
  - Avoids interoperability issues (ASCII/EBCDIC, partners)

Who Can Encrypt/Decrypt?

- Usual question is: who ***decrypts***?
  - Who should have the ability to decrypt PII?
- Should your staff have full access to all data?
  - Many unreported (or undetected) internal breaches occur
- What if someone leaves the company?
  - How do you ensure their access is ended?
- What if an encryption key is compromised?
  - Can you revoke it, so it's no longer useful?
- PCI DSS et al. ***require*** these kinds of controls
  - This is a big deal—***not*** trivial to implement

PCI Security Standards Council™

How Will You Encrypt Data?

- Hardware? Software?
  - Many options exist for both

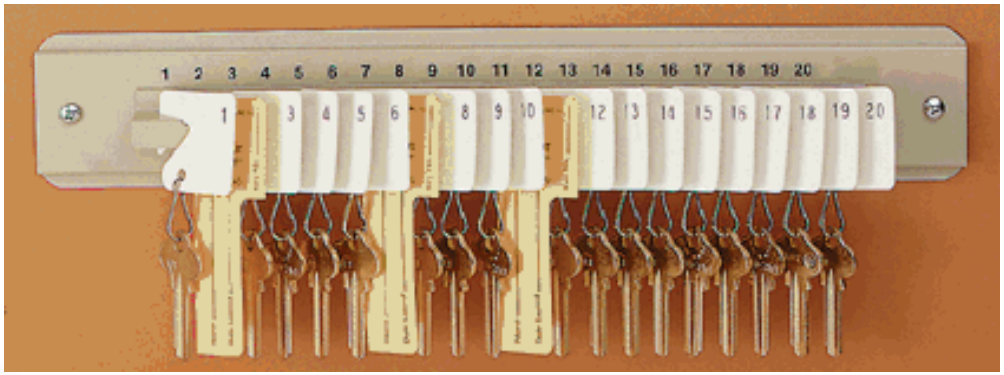
- Is a given solution cross-platform?
  - If not, you ***must*** decrypt/re-encrypt when data moves
- AES? TDES? Symmetric? PKI?
  - Many, ***many*** choices exist—too many!

How Will You Encrypt Data?

- Different issue: How do you get from here to there?
  - 100M++ data records—how to encrypt without outage?
  - “Customer database down next week while we encrypt”?!
- What about data format changes?
  - Encrypted data usually larger than original
  - Does not compress well (typically “not at all”)
  - Database schema, application fields expect current format
  - ***Can you change everything that touches the data?***
  - (Should you need to?)

Key Management

- "Encryption is easy, key management is hard"
  - Ultimately, encryption is just some function applied to data
  - To recover the original data, you need key management
- Three main key management functions:
  1. Give encryption keys to applications that must protect data
  2. Give decryption keys to users/applications that correctly authenticate according to some policy
  3. Allow administrators to specify that policy: who can get what keys, and how they authenticate



Key Management

- Key servers generate keys for each new request
  - Key server must back those up—an ongoing nightmare
  - What about keys generated between backups?
  - Maybe punch a card every time a key is generated…

- What about distributed applications?
  - How do you distribute keys among isolated networks?
- What about partners?
  - If you distribute encrypted data, how do they get the keys?
- “Allow open key server access” not a good answer
  - Suggest it, watch network security folks’ heads explode

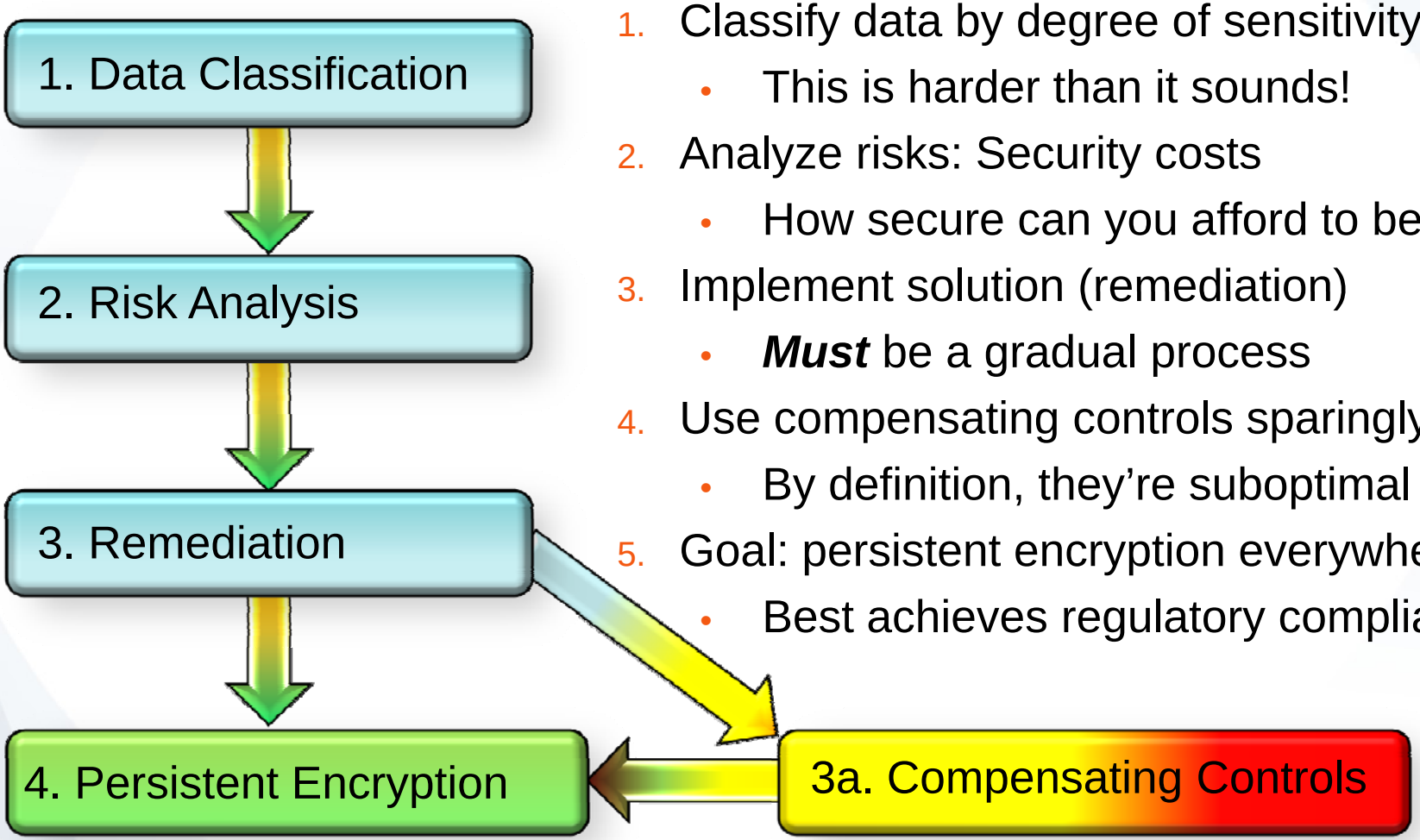
Getting There From Here: A Realistic Approach

A Realistic Approach: Take A Deep Breath

- Investigate encryption, now or soon
  - Better now than ***after*** breach
  - That light at the end of the tunnel ***is*** a train!

- Understand that choices have far-reaching effects
  - Data tends to live on for a very long time
- Expect to use multiple solutions
  - Backups, laptops, databases all have different requirements
  - “Right” answer differs
  - E.g., for backups, hardware-based solution; for customer database, column-based encryption

A Realistic Approach: High-Level Roadmap



1. Classify data by degree of sensitivity
   - This is harder than it sounds!
2. Analyze risks: Security costs
   - How secure can you afford to be?
3. Implement solution (remediation)
   - ***Must*** be a gradual process
4. Use compensating controls sparingly
   - By definition, they're suboptimal
5. Goal: persistent encryption everywhere
   - Best achieves regulatory compliance

A Realistic Approach: Key Steps

- **Key:** Involve stakeholders across the enterprise
  - "No database is an island": multiple groups use the data
  - Partners, widespread applications need access too…
- **Key:** Find a "starter" application
  - Generating test data from production is a good beachhead
  - If you "get it wrong", you haven't lost anything "real"
- **Key:** Designate data by sensitivity:

  Red: Regulated (legally required to be protected)

  Yellow: Intellectual property or other internal (unregulated)

  Green: Public
  - Each requires a different level of isolation/encryption

A Realistic Approach: Proof of Concept

- Encrypt a representative database
  - "Database" could be DB2, IMS, VSAM, flat file...
- Update application(s) that access it
  - You know what all your applications do, right? ☺
- Validate performance, usability, integrity
  - Encryption is ***not*** free: may see significant performance hit
- Demonstrate to other groups
  - Invite discussion, counter-suggestions
- Once (if!) project approved, request executive mandate
  - Otherwise, some groups may simply not participate

A Realistic Approach: Finishing the Job

- Doing ***all*** databases/applications takes time
  - Expect glitches
  - Perhaps most difficult: understanding data relationships
  - Table A and Table B seem unrelated, but aren't
- Lather, rinse, repeat…
  - Each database will have its own issues/surprises


Voltage
security
Voltage SecureData

Voltage SecureData

- Voltage SecureData: Yet Another Encryption Product
  - With some key differences, of course!
- Available on z/OS, Windows, Linux, z/Linux, HP/UX, AIX
  - Built on platform-agnostic codebase (easy to port)
  - Can add platforms quickly as customers require them
- Complete suite of options:
  - Toolkit (APIs) for application integration
  - Bulk data encryption tools for scripting/data masking
  - SOA server for legacy/lightweight platforms
  - Tokenization supported via SOA for sites that require it

Voltage SecureData

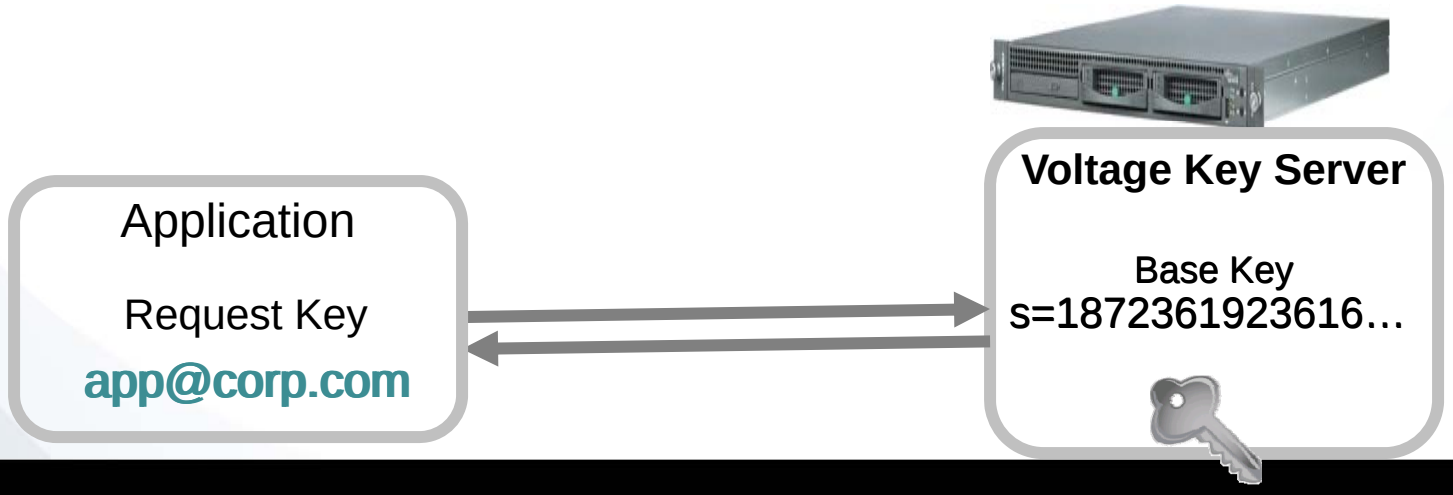
- Provides **Format-Preserving Encryption** (FPE)
  - Data encrypted with FPE has ***same format*** as input
  - Encrypted SSN still 9 digits; name has same number of characters; credit card number has same number of digits…
  - Avoids database schema changes, most application changes
  - Most applications can operate ***on the encrypted data:*** Less than 10% of applications need actual data
- FPE is proposed mode of AES
  - Google “ffx mode” or look for “FFX” on http://csrc.nist.gov/groups/ST/toolkit/BCM/modes_development.html
  - Peer-reviewed, well-established—not snake oil!

Voltage SecureData: Cross-Platform

- ASCII/EBCDIC handled automatically
  - Data converted to UTF-8 before encryption/decryption
  - Stored in native format on host (ASCII or EBCDIC)
  - Possible because character sets are deterministic (FPE!)
  - Result: z/OS is a full partner in protected data management
- Encrypt/decrypt ***where the data is created/used***
  - Avoids plaintext data ever traversing the network

Voltage SecureData Key Management

- Simplified key management eases most headaches
  - Keys are generated dynamically based on **identity**
  - Enables multiple key servers, serving ***same*** keys
  - Allows geographic/network isolation
  - Requires backup ***only*** when key server configuration changes
- Key request authentication allows separation of duties
  - Users/applications without access ***cannot*** get keys
  - Voltage SecureData makes full compliance much easier



Data Masking

- Application testing needs realistic datasets
  - Fake sample datasets typically too small, not varied enough
- Best bet: Use production data…**but:**
  - Test systems may not be as secure
  - Testing staff should not have full access to PII!

Data Obfuscation Today: Four Approaches

- Random Data
  - Replace data with random values
  - Destroys referential integrity
  - Can result in collisions
- Shuffling
  - Shuffle existing data rows so data doesn't match
  - Breaks referential integrity
  - Can still leak data, since values are "real"
- Fake data tables & rules
  - Consistently map original data to fake data
  - Provides referential integrity, reversibility
  - Massive implementation costs & security risks
- Weak, breakable encryption
  - E.g., stream ciphers, alphabetic substitution
  - Not secure – easily reversible by attacker
  - Key management challenges

- **IBM Optim**
- **Applimation**
- **Informatica**
- **CompuwareFile Aid**
- **Camouflage**

- All fit into these "legacy" approaches
- Need another database to manage rules/mappings – more risk, effort, etc.!
- Must run process to create test data

Voltage SecureData for Data Masking

- Answer: Use encryption to mask (anonymize) test data
  - With FPE, encrypted production data is perfectly usable for test
  - No extra steps required!
- Or can create test data on demand (subset, etc.)
  - Further protects test environment from possible internal breach
  - If random key used, data cannot be decrypted
  - Alternatively, use actual key, decrypt only to verify results/diagnose issues
  - Can even re-encrypt production encrypted data

Voltage SecureData

- "Rolling" keys is required by PCI DSS, other standards
  - Means re-encrypting with new key, invalidating old key
  - Required periodically, if trusted staff leaves, if breached, etc.
- With most encryption solutions, this is a nightmare
  - With SecureData, can re-encrypt on-the-fly
  - Or encode key version in encrypted data
  - In any case, separation of duties through identity-based key provisioning makes it easy to revoke user's access

Reduced Audit and Risk Scope

- Persistent encryption prevents accidental leakage
  - Compensating controls only cover holes you know about
  - Integrate with existing monitoring and scanning tools
- True separation of duties
  - DBAs can still do their jobs, no access to “Red” data without authorization
- Role-based access model allows granular data policies
  - CSR only sees last 4 of credit card; fraud investigator sees all 16
  - Full re-use of identity/access management systems

Using Voltage SecureData

- SecureData Toolkit
  - APIs callable from LE languages
  - Simple: one call to initialize, one call for each encryption/decryption, one call to terminate
- z/FPE and the SecureData CL
  - Scriptable tools for z/OS (z/FPE) or distributed (CL)
  - Both built as Toolkit applications
- z/FPE
  - Runs against flat files, or as user exit
  - Uses customer-written code (Rexx or LE) to control operation

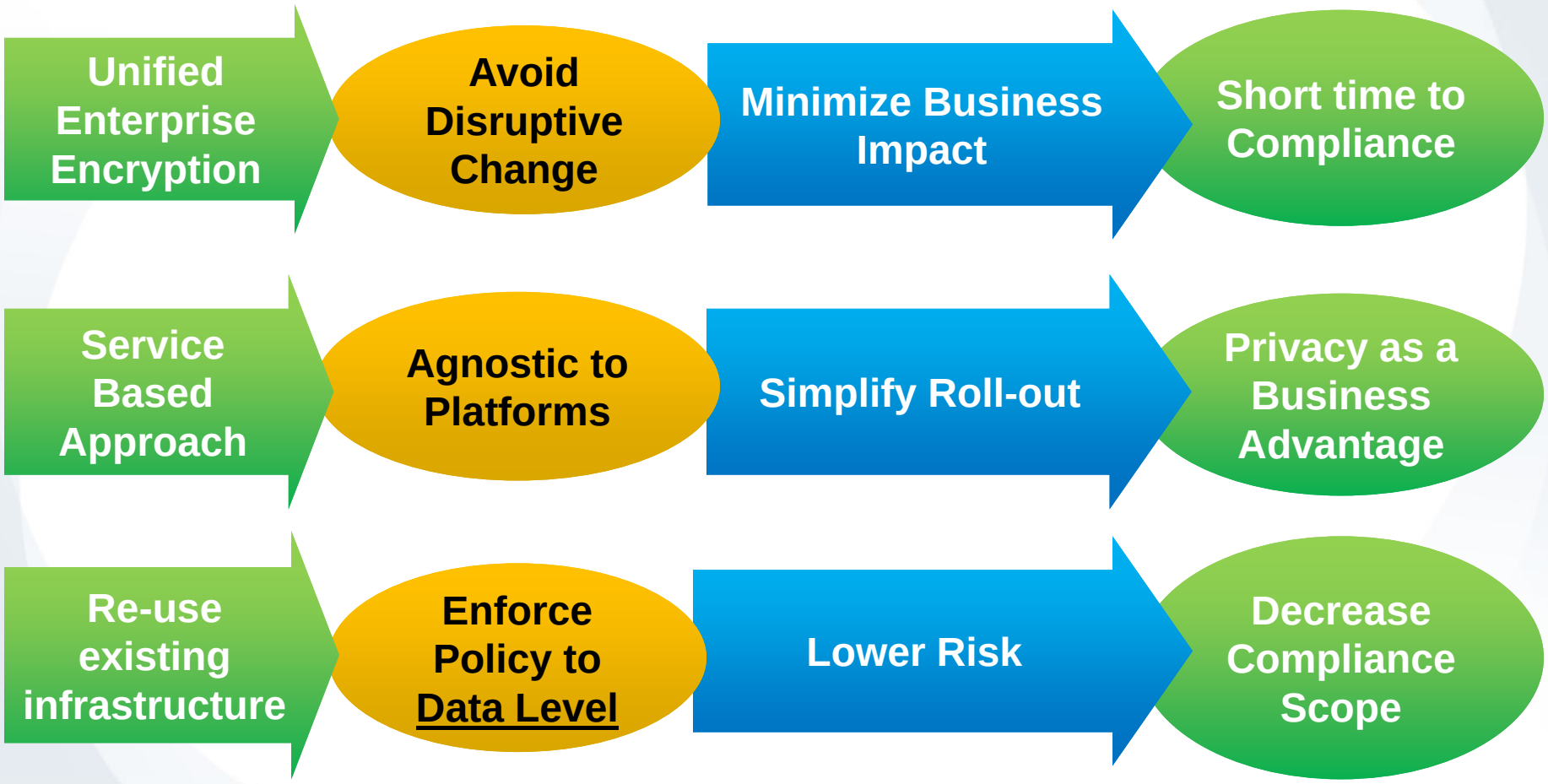
Voltage SecureData Advantages

- Meets all data protection requirements
  1. Persistent protection of any data type/field agnostic of database
  2. Full segregation of duties between data, administrators, applications, and permitted users, with full audit trail
  3. One solution for both persistent data protection and data masking/de-identification
  4. Full dynamic central key management — no key storage/backup
  5. Supports existing identity management /authorization systems

Enterprise PII Privacy with Voltage SecureData

Use Case	Business Driver	Data-centric Business Benefit & Cost Savings
Data protection	Enterprise privacy compliance, Fast, low-cost PCI Compliance	Reduce audit scope, Automate repetitive compliance processes
Data masking for test/QA	Reduce costs with compliant outsourcing and off-shoring	Simple, immediate data de-identification
Securing mobile app data	Capture payments or customer data at point of sale	Embrace new platforms – mobility adoption e.g. iPhone
Securing partner data	Legal and contractual obligations	Extend the enterprise without losing control

Data-centric Approach Benefits Summary




Voltage
security
Summary

Conclusion

- Encryption is not a luxury, not optional today
- A complex topic, but one that ***can*** be tamed
- Many solutions exist
- Different data/media require different solutions

- Voltage SecureData solves many of the problems for data at rest and data in motion
  - Not a solution for whole-disk, whole-tape encryption
  - The best solution for existing data, existing applications

Voltage security

Encryption Resources

- InfoSecNews.org: email/RSS feed of security issues
  http://www.infosecnews.org/mailman/listinfo/isn
- Voltage security, cryptography, and usability blog
  http://superconductor.voltage.com
- Bruce Schneier's CRYPTO-GRAM monthly newsletter
  http://www.schneier.com/crypto-gram.html
- RISKS Digest: moderated forum on technology risks
  http://catless.ncl.ac.uk/risks
- US Computer Emergency Response Team advisories
  http://www.us-cert.gov/cas/signup.html
- Tracking breaches: http://datalossdb.org/ and
  http://www.privacyrights.org/ar/ChronDataBreaches.htm

Questions?

Phil Smith III
703.476.4511 (direct)
phil@voltage.com
www.voltage.com