Encryption? Yeah, We Do That

Encryption facilities, challenges, and choices on System z

Agenda

- Tour System z encryption facilities
- Survey available IBM products
- Briefly discuss third-party technologies (not products)
- Examine criteria for making intelligent selections
  - ***Not*** judging/comparing products *per se!*

Some Fundamental Points about Encryption

- Encryption is not ***fun***
  - Any encryption project involves some (or a lot) of work!
- Encryption does not make your job easier
  - Even once implemented, it’s one more thing to keep track of
- Encryption should not advertise itself
  - Done right, encryption is invisible to the users
- Encryption is difficult and complex
  - Unless you have a PhD in math, prepare to not understand many of the details

So Why Would Anyone Want to Encrypt?

- Regulatory compliance
  - HIPAA, GLBA, Red Flag, Sarbanes-Oxley, et al.
- Recovery from a breach
  - "Do something so this can't happen again!"
- General hygiene (breach prevention)
  - It could happen to you…
- ***Not*** encrypting may risk company's future
  - But doing it badly is worse than not doing it at all (data loss!)

Do We Really Need to Care?

- **Mainframes are secure** – we all “know” this
  - Not inherently true
  - Reflects decades of rigid change control theology
  - Aided by historically lagging mainframe Internet connectivity
- ***Not*** something you want to bet your job on!
  - Mainframes are increasingly connected to the ’net
  - Inside-the-firewall connections also offer attack vectors
  - Partnering often means data travels far from home
  - Outsourcing means other companies share floorspace, hardware

So You Need To Encrypt Some Data…

- Where will the data live?
  - Network
  - DASD
  - Tape
  - Flash drives
  - DVDs
  - Punched cards
  - Smoke signals
- These are ***different***, require different solutions

Narrowing the Problem

- On mainframes, DASD and tape are the concerns
  - Network traffic: Use SSL (or Connect:Direct or scp or sftp)
  - Flash drives, DVDs: Not a z problem
  - Punched cards: Hopefully no longer a z problem!
  - Smoke signals: Call your CE
- DASD and tape are “data at rest”
  - But are still largely ***different*** problems from each other

Hardware vs. Software

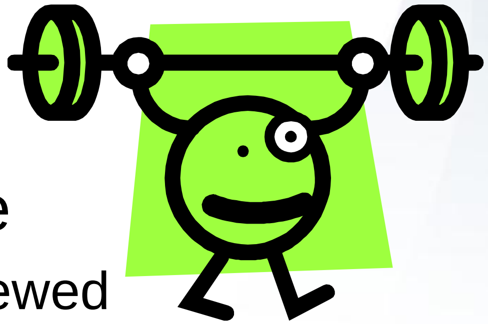
- Encryption can be performed by
  - Software routines using everyday instructions
  - Software using specialized instructions
  - Hardware: instructions, millicode, HSMs, external servers
- The U.S. government considers encryption a “munition”
  - Places restrictions on its export
  - Includes some hardware facilities, software packages
  - Availability thus limited in some countries

A Word about “Point” Solutions

- Many products include some form of encryption
  - Outlook encrypts stored mail by default
  - Many products encrypt passwords internally
- Not necessarily secure
  - May use weak encryption
  - Are keys sufficiently managed/stored/protected?
- Such point solutions can proliferate
  - Suddenly you have 27 solutions for 27 slightly different problems
  - No commonality, management nightmare

Encryption “Strength”

- Encryption **“strength”** refers to the likelihood that an attacker can “break” encrypted data
  - Typically tied to bit length of encryption key
  - Exponential: 128-bit key is 2**64 times as strong as 64-bit
  - See “Understanding Cryptographic Key Strength” on `youtube.com/user/VoltageOne` for a good discussion/illustration



- The encryption community is collaborative
  - Research, algorithms are published, peer-reviewed
  - Cryptographers look for weaknesses in each other’s work

Proving Encryption Strength

- Cryptographers “cheat” in attacker’s favor when analyzing
  - Make assumptions like “attacker has multiple known examples of encrypted data and matching plaintext”
  - Also assume they’ll know plaintext when they find it, and that the encryption algorithm is known
- “Weaknesses” reported are often largely theoretical —only NSA could really exploit
  - Huge amounts of time, brute-force computing power required
  - E.g., recent AES “weakness”: ¼ the previous strength, so 2 billion years to crack, not 8 billion…

More About Proving Encryption Strength

- This “cheating” ensures encryption strength is real*
  - This approach increases security for all
  - By the time an algorithm is accepted as a standard and implemented in products, confidence is high
  - Even if a weakness is later discovered, it’s likely largely theoretical/impractical for most to exploit
- Makes it easy to spot the charlatans
  - Companies whose proprietary algorithms are ***not*** peer-reviewed
  - Also look for claims like “unbreakable encryption”, or focus on key length rather than standards-based cryptography

*** Well, as real as the smartest minds in the business can make it!**

IBM System Facilities

System z and z/OS encryption capabilities

IBM Common Cryptographic Architecture

- CCA "…provides a comprehensive, integrated family of services that employs the major capabilities of the IBM coprocessors"
  - In other words, common APIs across different platforms
  - Makes it easier to port skills across systems
  - Also smart since IBM HSMs work on multiple hardware
- Offers robust functionality
  - Symmetric and asymmetric encryption operations
  - Key generation, import, and export
  - PIN generation, random number (entropy) generation
  - etc.

Integrated Cryptographic Services Facility

- Integrated Cryptographic Services Facility (ICSF)
  - z/OS implementation of CCA
  - Started Task provides crypto interfaces to crypto card
  - Requires hardware facilities for some functions
- Active area for IBM development
  - New ICSF levels often appear between z/OS releases
- Mostly just a toolkit, however
  - Requires "roll-your-own" work to build encryption solutions

SSL on System z

- SSL (Secure Sockets Layer), aka Transport Layer Security, is transport-layer network traffic encryption
  - Does "handshake" with partner, determines shared trust
  - Generates key to encrypt traffic for duration of session
  - Uses asymmetric encryption and certificates during handshake
- SSL is standard technology
  - Used for https, secure SMTP, others
  - TCP-only, so some services cannot use it

SSL on System z

- System SSL is IBM's SSL implementation
  - Part of z/OS Cryptographic Services Base element
  - Same underlying code used on z/VM, z/VSE
  - z/TPF uses OpenSSL (same functionality)
- Robust, well-documented API
  - **`GSK`*`xxxxx`*** members in **`SYS1.SIEALNKE`** on z/OS

IPSec on System z

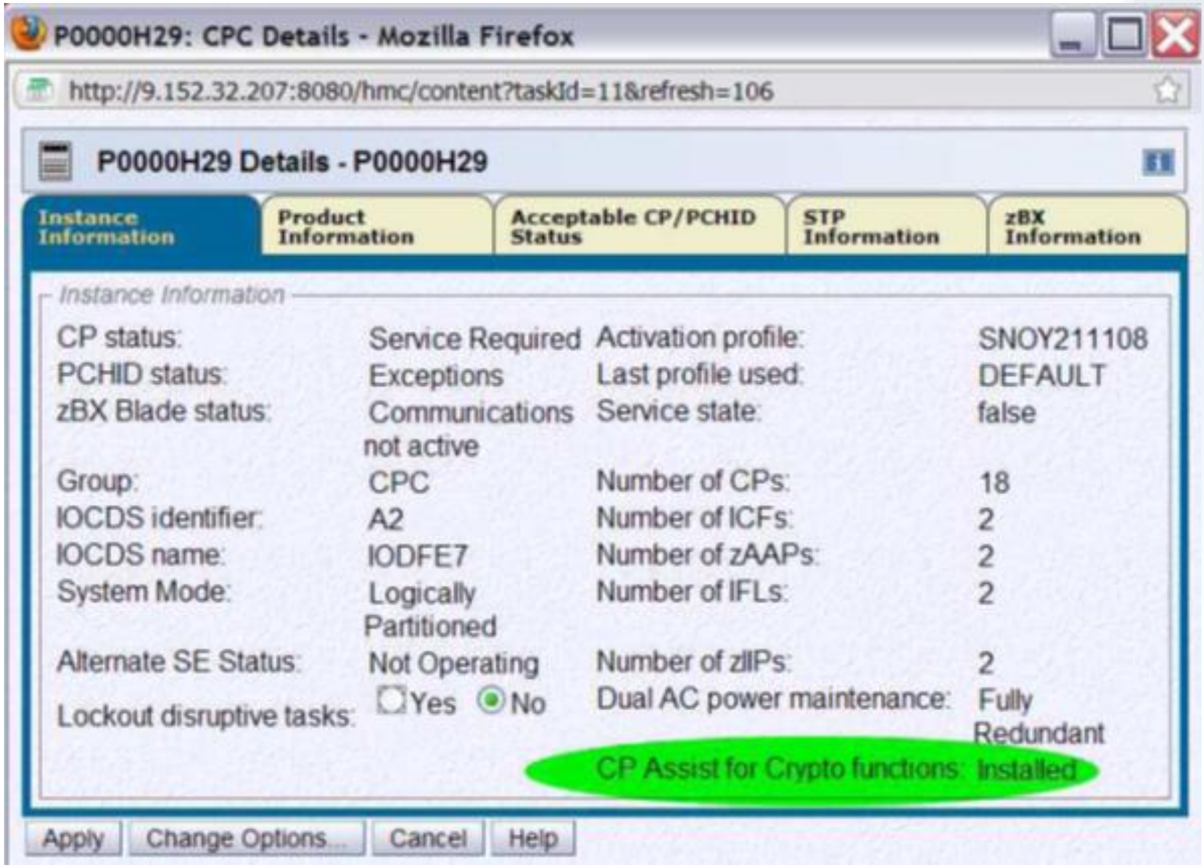
- IPSec is an IP-layer protocol for securing traffic
  - Does certificate-based authentication of partner, ~like SSL
  - IPSec works with any protocol, any application
  - Seen as slightly less secure than SSL, but more general
  - Useful for tunneling host-to-host traffic
  - For example, commonly used by VPNs
  - Can also be used at application layer (IKE mode)
- Implemented in z/OS TCP/IP
  - IPSec can be offloaded to zIIP
  - Linux for System z includes IPSec too
  - z/VSE, z/VM, z/TPF not playing here (yet?)

CPACF

- Central Processor Assist for Cryptographic Functions
  - Commonly pronounced "see-paff"
  - Single-instruction implementations of AES, DES, etc.
  - Combination of silicon and millicode
- Introduced with z9 in 2005
  - Additional functionality came on z10
  - zEnterprise adds still more
- CPACF reduced AES-256 CPU by 60% in our tests
  - Pretty significant if you're doing a lot of encryption

CPACF Enablement

- CPACF is free but enabled via Feature Code 3863
  - One of those munitions unavailable in countries we don't like
- "How do I tell whether CPACF is enabled?"
  - HMC display
  - Bits in CCVT (Crypto CVT)



Crypto Express2 and Crypto Express3

- Crypto Express: IBM Cryptographic Security Module
  - AKA “Hardware Security Module” or HSM
  - Same core technology as 4764/4765 HSMs for other platforms
  - Tamper-proof, secure crypto operations via add-in card
  - Validated to FIPS 140-2 Level 4 (highest level of validation)
- Crypto Express3 is current, replaced Crypto Express2
  - Which itself replaced PCI X Cryptographic Coprocessor (4758)
  - Similar functionality, improved RAS etc. with each generation
  - Various models with varying number of interfaces

CEX, CEX, and More CEX!

- A single CEC can have up to eight CEX installed
  - Each CEX contains two interfaces
  - Except -1P models for BC machines, which have one
- Each interface can be configured two ways:
  - As cryptographic coprocessor (CEX2C, CEX3C)
  - As SSL accelerator, for RSA operations (CEX2A, CEX3A)
- CEX also support “User-Defined Extensions”
  - Custom operations, created by IBM (for $), installed on CEX
  - Used by banks, for example, for custom PIN derivation

SSL Handshake Performance

- As a CEX2C/3C, CEX still helps with SSL
  - IBM results using z196 Model 754 (4 full-speed engines)

Method	ETR	CPU%	Crypto%
Software	1204	100	n/a
8 CEX3C	14457	95.24	92.3
4 CEX3A	14429	99.72	80.7

  - With (plenty of) CEX, more than 10x improvement
  - CEX3A is about double CEX3C!
  - CPU utilization 100% without CEX, lower with

CKDS and PKDS

- ICSF can populate/manage two special data sets
  - **CKDS**: Cryptographic Key Data Set
  - **PKDS**: Public Key Data Set
  - Each contains encryption keys
  - Used by many products
- Keys can be stored in CKDS/PKDS in encrypted form
  - Encrypted ("wrapped) by CEX using Master Key stored in CEX
  - Master Key is entered using ICSF panels or Trusted Key Entry (TKE) workstation feature
  - Master Key is ***never*** known to z/OS: only to CEX

CKDS, PKDS, and Secure Key Operation

- When an encrypted key from CKDS/PKDS is used:
  1. Application fetches key from *x*KDS
  2. Calls ICSF with data and encrypted key
  3. ICSF calls CEX
  4. CEX decrypts key with Master Key
  5. CEX performs operation on data
  6. Crypto result returned to ICSF, thence to application
  - ***Plaintext keys never reside in System z memory***
- This is known as **Secure Key** operation
  - Not ***super***-slow, but must do I/O to CEX, etc….
  - Suboptimal for large amounts of encryption

Protected Key Operations

- ICSF added **Protected Key** in 2009
  - FMID HCR7770
  - Hybrid solution, providing (most of) “Best of both worlds”
  - Exploits combination of CPACF and CEX (via ICSF)
- Stored keys in z/OS are still encrypted
  - CEX call decrypts key, re-encrypts with “wrapping key”
  - Copies wrapping key to protected HSA memory
  - Wrapped key returned and used on CPACF calls

Review: Key Operation Modes

- Clear Key
  - Keys stored unencrypted, CPACF performs operations
  - Fastest but least secure
- Secure Key
  - Keys stored encrypted, CEX decrypts key, performs operation
  - Slowest but most secure
- Protected Key
  - Keys stored encrypted, CEX decrypts key, re-encrypts with "wrapping key", returns wrapped key
  - CPACF performs operations
  - "Most of the performance with most of the security"

CPACF and Crypto Express Support

- All IBM operating systems support CPACF and CEX
  - z/OS ICSF uses CPACF or CEX as appropriate/available
  - z/VM guests can use CPACF, be given CEX access
  - z/VSE supports CPACF and CEX (no RSA Secure Key)
  - z/TPF supports CPACF, CEX as RSA/SSL accelerator
  - Current Linux for System z distros fully support both

ICSF and SAF (RACF, ACF2, Top Secret)

- SAF can control ICSF
  - CSFSERV resource class
  - If not activated, no controls over ICSF
- CKDS/PKDS are special to SAF (RACF, ACF2, TSS)
  - Each record (each key) is secured separately
  - Controlled by CSFKEYS resource class

Voltage security

Misconception: "CEX is Always Good"

- Easy assumption to make: "Using CEX is always faster"
  - Not true: CEX mainly for ***security*** not ***performance***
  - ***Certain*** operations (SSL/RSA) are faster
  - ***Most*** operations are slower: ICSF must do I/O to CEX
- For everyday cryptography (besides SSL handshakes):
  - Best **performance**: CPACF
  - Best **security**: Crypto Express
- CEX ***might*** be cheaper CPU-wise with large data blocks
  - Still slower wall-clock, unless CPU really, really overloaded

Approaches and Criteria

"They all claim they'll solve all our problems!?!"

Hardware or Software?

- Hardware:
  - Avoids system load, since encryption is offloaded
  - Typically does not require code changes
  - ***But*** narrower applicability – works or doesn't in given use case
  - Cannot provide Separation of Duties controls (discussed next)
- Software:
  - May be expensive to buy
  - Can use significant system resources to run
  - ***But*** broader solution: can be added to any application

Separation of Duties

- Separation of Duties (SoD) is important for real security
  - Means “need to know” required for decryption
  - E.g., just because you’re a DBA, you do not need to see SSNs
  - Without it, protection (and compliance) often difficult/impossible
- Fully transparent solutions fail to provide SoD
  - E.g., if table accesses automatically decrypted, no SoD
  - Must be ***some*** form of credential/access control in the process

Separation of Duties: The Reality

- Implementing true SoD requires application changes

"You can have peace. Or you can have freedom. Don't ever count on having both at once."

— Robert A. Heinlein

  - You can add security, or you can avoid changing applications
  - People always ***<u>want</u>*** to avoid having to change applications
  - Understandable but unrealistic: no "magic bullets"

Key Management

- Key management equally critical
  - What if you need data off a tape ten years from now?
  - Can you access keys in DR scenarios?
- Robust, flexible key management is a must
  - Key management involves three primary 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?
- What about distributing keys?
  - How do you distribute keys among isolated networks?
  - What about partners? How do they get required keys?
- Too many solutions focus on the encryption algorithm
  - Key management is harder and equally critical

Voltage security

IBM Encryption Products

System z and z/OS Hardware and Software from IBM

Encrypting Hardware

- IBM encrypting tape drives: TS1130, TS1140
  - Whole-tape encryption
  - Most useful for protecting backups
  - Tivoli Key Lifecycle Manager (“TKLM”, aka IBM Security Key Lifecycle Manager for z/OS) manages keys
- Encrypting disk array: DS8000
  - Whole-DASD encryption
  - Protects data in shared environments
  - Also removes worries when DASD decommissioned
- Performance impact of this encryption is minimal
  - Alas, so is utility, other than specific use cases listed above

InfoSphere Guardium Encryption Expert

- Whole database encryption
  - Formerly IBM Data Encryption for IMS and DB2 Databases
- Significant performance impact
  - Up to 400% more CPU per IBM, even with CPACF
  - Keys are stored in CKDS
  - Can use Protected Key or Secure Key (CEX) if required
- Limited value
  - Performance hit often unacceptable
  - Most regulations require Separation of Duties

Encryption Facility for z/OS

- File-level encryption
  - Described "…encrypt sensitive data before transferring it to tape for archival purposes or business partner exchange"
  - Includes no-charge decryption client (unsupported)
  - Can also compress data before encryption
  - Uses "System z format" or OpenPGP (various algorithms)
- Useful tool for specific purposes targeted
  - OpenPGP includes asymmetric algorithms
  - Could be integrated into existing processes
  - z/OS only, further limiting applicability
- Same product available for z/VSE

IBM® Sterling Connect:Direct®

- Automated, secure file transfer between systems
  - Formerly Sterling Commerce Connect: Direct
  - Formerly Sterling Network Data Mover
  - Formerly Systems Center NDM
  - Still commonly called “NDM”
- Mature, powerful product
  - Think “FTP or scp, only more programmable and secure”
  - Backbone of many companies’ daily operations

ISV Encryption

Approaches and Options

Hardware or Software?

- Same criteria as with IBM products
  - Hardware avoids system load, but narrower applicability
  - Software can be expensive to buy/run, but broader solution
- Separation of Duties is important
  - Without it, protection (and compliance) often difficult/impossible
- Key management equally critical
  - What if you need data off a tape ten years from now?
  - Can you access keys in DR scenarios?

Hardware Solutions

- Various hardware options
  - Tape drives: Oracle (SUN [STK]), Hitachi/HP
  - DASD: Usual suspects (EMC, (SUN [STK]), Hitachi/HP)
  - Network level: more choices than you can count…
- Need to understand the problem being solved
  - Hardware can be a fine solution to a specific problem!
  - But usually not a general answer: some/most data not eligible

Software Solutions

- z/OS encryption products fall into three categories
  1. Very narrow, "point" solutions (e.g., file encryption)
  2. SaaS/SOA/SOAP (web services) remote server-based
  3. Native (with or without hardware exploitation)
- Do you want to manage dozens of point solutions?
  - Or one enterprise solution?
  - Also see *Enterprise Encryption 101* at www.share.org/Portals/0/Webcasts/2012%20Webcasts/Getting%20Started.wmv or http://bit.ly/wtMriL

Voltage security

Point (Narrow) Software Solutions

- Plenty of “encrypt a file” products available
  - Typically include weak key management, if any
  - Intended to encrypt data prior to backup or partner exchange
  - Some are specific to tape backup (e.g., FDRCrypt)
  - Useful to solve specific point problems
- Many choices
  - Rocket Software
  - CA Technologies
  - Code Magus
  - OpenTech
  - PKWARE
  - Innovation Data Processing

SOA (Web-based) Software Solutions

- Server (real or virtual) installed on your network
  - z/OS applications pass data to server, returned en/decrypted
  - SaaS: Transaction uses SSL, many use SOAP
  - Requires minimal software on host
- Weaknesses:
  - Performance: SSL connections involve overhead, delay
  - System z folks often uncomfortable with operations “out there”
  - Effective as z/OS point solution, if performance acceptable
- Several choices
  - Protegrity
  - Safenet
  - Liaison Techologies (formerly nuBridges)

Native Software Solutions

- APIs to add to existing applications
  - Make sure usable from all environments, e.g., CICS
  - Language support may be limited
  - Implementation can be complex
  - Some exploit CPACF, some do not
- Again, varied choices:
  - RSA (EMC): C/C++ and Java APIs
  - CFXWorks, Entrust: Java-only APIs
  - Redvers Consulting: COBOL-only API
  - Prime Factors, Advanced Software Products Group: general-purpose APIs

Making Intelligent Choices

First Step: Understand the Problem

- “We need some encryption” isn’t sufficient
  - To protect what?
  - From whom?
  - What else will this of necessity affect?
- Requires executive sponsorship
  - Otherwise expect to fail
  - Nobody ***wants*** to do encryption!
- Expect a successful implementation to spread
  - Picking a very limited solution now may lead to regrets later

Security-Related Questions

- Is algorithm strong, peer-reviewed?
  - No real reason to use anything but AES
  - Asymmetric use cases should usually use “wrapped” AES
- Does it support hardware assists?
  - Improves performance
  - Eliminates side channel risks
- Is key management part of the solution?
  - Must keys be stored multiple places, secured independently?
  - Include key rollover requirements, if needed
  - Long-term historical key access is nothing to fool with!

Operational/Deployment Questions

- Is implementation cost reasonable?
  - Not just $$$, time and effort are even larger costs
  - Consider having to train tens/hundreds of different developers
- Is implementation under ***your*** control?
  - Can your folks do most/all of the real work?
  - Must they develop crypto expertise to exploit it?
  - Or is “product” really a Professional Services play by vendor?
- Is it multi-platform?
  - If this is a known requirement, it’s a very important one
  - Even if it isn’t, what happens if/when encryption use spreads?

Voltage SecureData

Voltage SecureData

- Voltage SecureData: Yet Another Encryption Product
  - With some key differences, of course!
- Available on z/OS, Windows, Linux, HP/UX, AIX, more…
  - Built on platform-agnostic codebase (easy to port)
  - Can add platforms quickly as customers require them
  - Exploits HSMs (and CPACF, Crypto Express)
  - ASCII/EBCDIC issues handled transparently

Voltage SecureData

- Complete suite of options:
  - APIs for application integration
  - z/OS Started Task-based encryption server
  - Bulk data encryption tools for scripting/data masking (z/FPE, CL)
  - SOA server for legacy/lightweight platforms
  - Tokenization supported via SOA for sites that require it

SecureData Simple API | **z/Protect** | **z/FPE, SecureData CL** | **SecureData SOA**

Voltage SecureData z/Protect

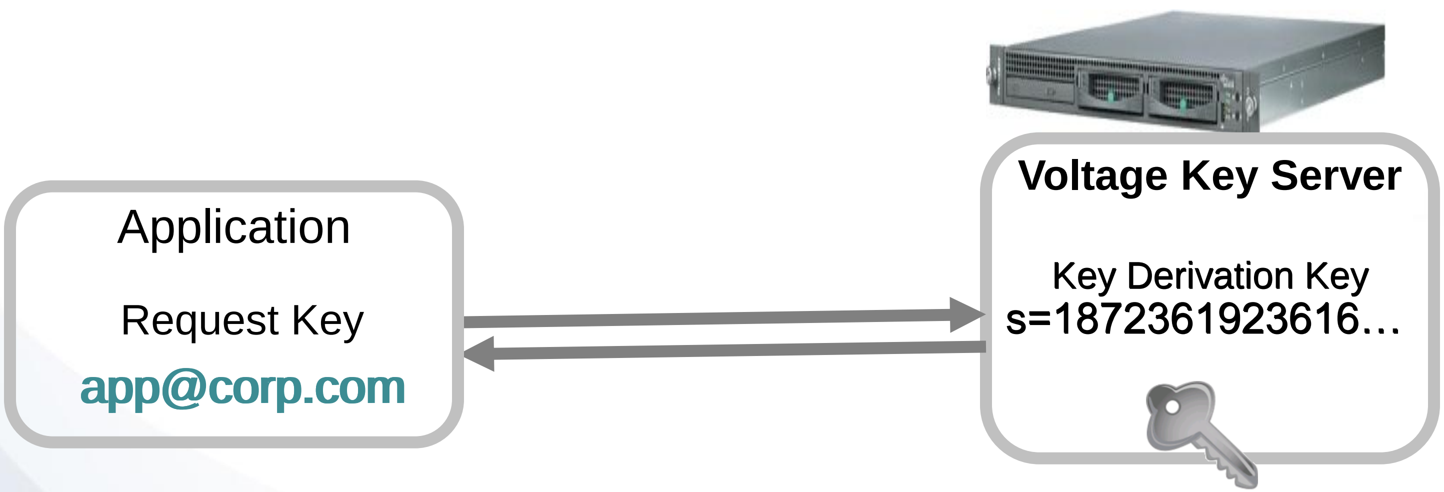
- Complete z/Protect code to perform encryption:

```
call vshprot using CRYPTID, ssn, length returning rc.
```

- **<u>Cryptid</u>** rhymes with "lipid"
  - Defined in z/Protect Started Task configuration
  - Combines **all aspects of encryption** into 1- to 64-byte name
- Cryptids allow complete centralized control
  - Tell application programmers "Use the Cryptid named **XYZ**"
  - Administrator changes Cryptid definition for key rollover, etc.
- ***The simplest encryption API available anywhere***
  - Makes encryption much less difficult for applications teams

Key Management

- Voltage 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 server configuration changes
  - Key requests are authenticated: separation of duties



Voltage SecureData Benefits

- FPE minimizes implementation difficulty
  - Databases require no schema changes
  - Most applications require minimal or no code changes
- Persistent encryption prevents accidental leakage
  - Compensating controls only cover holes you know about
- True separation of duties
  - DBAs can do their jobs, no access to PII without authorization
- z/Protect revolutionizes integration of encryption
  - Orders of magnitude simpler than any other solution

Conclusion

Summary

- System z is a full player in the encryption world
  - Industry-leading hardware assists, HSM capabilities
- Many encryption approaches exist
  - Suitability depends on specific use cases
  - But be careful, encryption use tends to spread!
- IBM, vendors offer varied products
  - Some quite powerful, some very limited
- Voltage SecureData is available on many platforms
  - Enterprise-strength, proven in largest encryption projects

Questions?

Phil Smith III

703.476.4511 (direct)
phil@voltage.com
www.voltage.com