Identify, Anonymize, and Encrypt your Data Repository PHI data

ALEX WALKER

Objectives

1. Identify columns containing PHI
2. Consider encryption options
3. Implement a proof of concept encryption system

A little bit about Me

- 10+ Years in MT space
- Vice President of Product Development
  - OpenGate
  - DrAuditor
  - DrDashboard
- DR Programming Supervisor @ MT
- Not a security expert
  - “know enough to be dangerous”

What Risks?

- Consider the breadth of data in your DR
- Consider the breadth of ETL out of DR
- Consider applications using DR data

- Are you scared yet?
- Hopefully we can talk through some ways to help you sleep at night

What is PHI?

Just to Review

1. Names
2. Geographical identifiers < State
3. Dates
4. Phone Numbers
5. Fax Numbers
6. Email Addresses
7. Social Security Number
8. Medical Record Number
9. Health Insurance Beneficiary Numbers
10. Account Numbers
11. Certificate/License Numbers
12. Vehicle Identifiers (license plate)
13. Device Identifiers
14. URLs
15. IP Address
16. Biometric Identifiers
17. Full face Photographs
18. Any other unique identifying number

Data At Rest in the Data Center

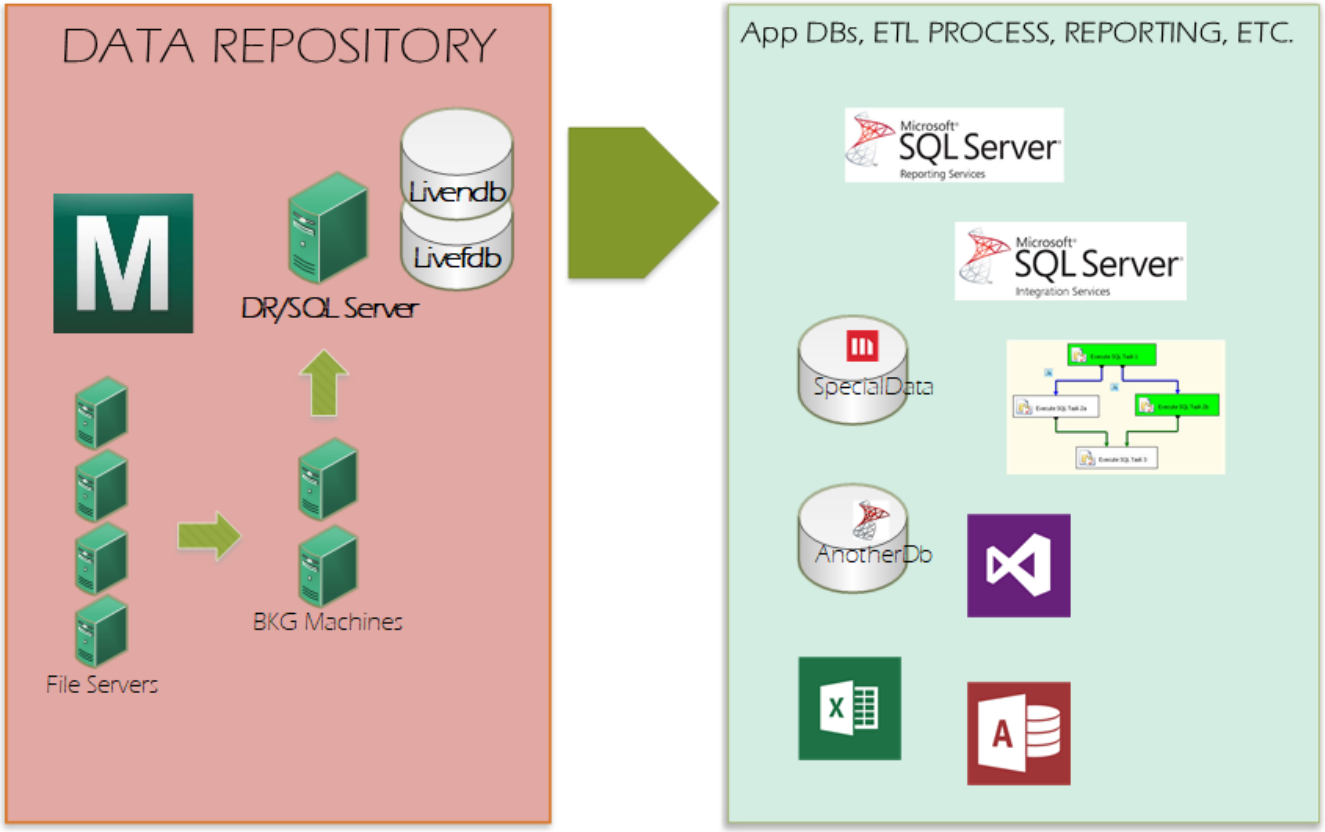
MEDITECH does not encrypt data at rest in the data center. Since Stage2 was an Interim Rule and then made Final, customers have asked about what MEDITECH will be doing to encrypt data at rest on file servers.

EHR Certification requires MEDITECH to ensure that we can pass a test script entitled "*Test Procedure for §170.314(d)(7) End-user device encryption.*" Note the title and script itself speak to "end-user devices."

With our ability to offer customers encrypted connections from end user devices to a file server or application server we and the entity are covered. The fact that when connections terminate, no PHI (Protected Health Information) is left behind also covers MEDITECH and the entity. The fact that the download routines have the capability to encrypt the file downloaded again covers MEDITECH and the entity.

There is no requirement in the legislation or test script to encrypt data on file servers or SANS. This is good news and the HIT Policy Panel took the advice of many who testified that system performance would be adversely affected if having to encrypt and de-encrypt each transaction in database structured EMRs.

What Can We Do?



PHI Identification

…AND DATA MAPPING TOO!

Let Meditech do the work!

- DDP 13151
  - Test Procedure for §170.314(d)(7) End-user device encryption

  - Creation of centralized Protected Health Information (PHI) temp file log and utilities.
  - Review of temp file creation and destruction.
  - Review of all applications to properly record patient id in log.

Let Meditech do the work!

- DDP 13151
  - Test Procedure for §170.314(d)(7) End-user device encryption

- If report output is downloaded to client
  - If report contains PHI – ENCRYPT temp file

Let’s Map those PHI Elements

- Identify MT elements flagged as PHI
- Cross reference to DR schema data
- Make decisions:
  a) Data is used for joins / searching (i.e. hashing)
  b) Data needs to be recoverable (i.e. decrypted)

Identify Elements in NPR

- NPR element attribute
  - NPR.SEG.attributes
  - NPR.SEG.ele.attribute.index = “PHI”
- NOT found in DR’s NPR application tables
- Have to use ‘other means’ to get:

```
SELECT NPR.SEG.ele.dpm, NPR.SEG.ele.seg,
NPR.SEG.ele.name
FROM NPR.SEG.attributes
where NPR.SEG.ele.attribute.index = "PHI"
```

Identify Elements in MAT

- MAT record in FocObj
  - FocObj.Fields.FieldPhi
- Found in DR's Foc application tables!
  - FocObj_Fields.FieldProtectedHealthInformation
  - Foc tends to be sparsely populated…

```
SELECT @OID(x), r.Rec, f.FldName
FROM FocObj.Idx x
JOIN FocObj.Records r
JOIN FocObj.Fields f
WHERE x.IdType = 'M' AND f.FldPhi = 'Y'
```

Cross Reference to DR

- NPR
  - Use DrTableMain / DrColumnMain
- MAT
  - Use DrTable_Main / DrTable_Columns

- Which of your queries access PHI?
- Which of your ETL process access PHI?

What are the shortcomings?

- How comprehensive is the schema?
  - Is it maintained?
- What about other sensitive data?
  - Payroll / Personnel Data
    - Massachusetts has privacy laws for this too
  - Accounts Payable
  - MIS Vendor Dictionary
    - Federal ID Number

Public Key Infrastructure

PKI Fundamentals

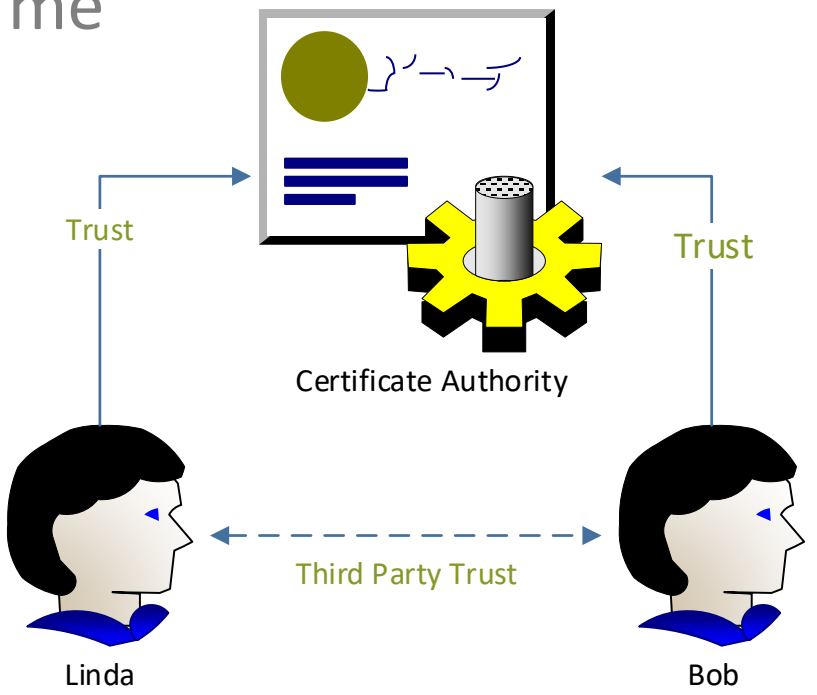
- Authentication – Digital Certificates
- Confidentiality – Encryption Algorithms
- Access Control – Public / Private Key Pairs
- Non Repudiation – Digital Signature
- Integrity – Message Hashing

Certificates

- Proof of Identification
- Contains
  - Serial Number
  - Subject
  - Issuer
  - Begin/End Date
  - Usage
  - Public Key
  - Signature Algorithm
  - Signature

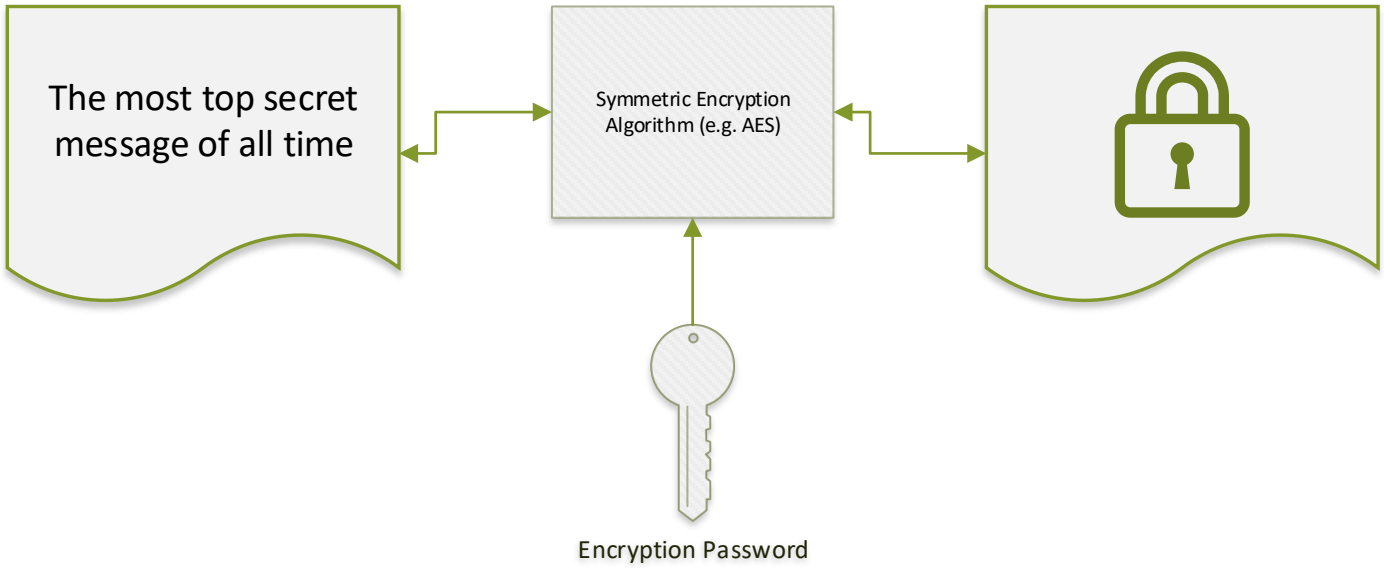
Establishing Trust

- Share with each other
  - Direct Trust
  - “I trust you and you trust me”
- Certificate Authority
  - Third Party Trust
  - “Should I trust her?”



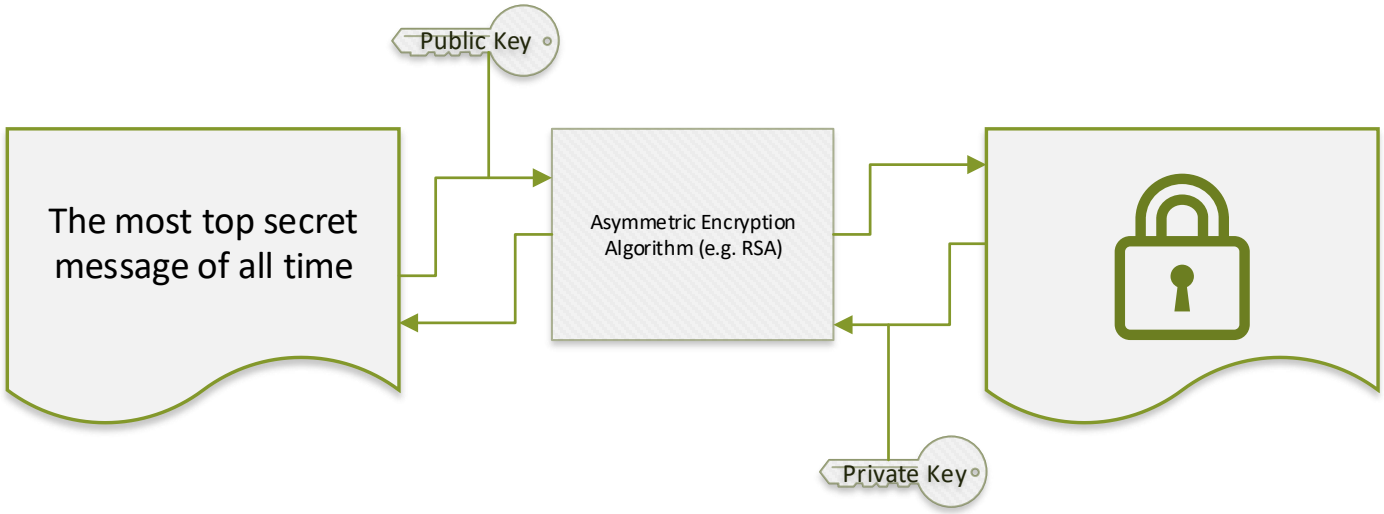
Symmetric Key Encryption

- Same key for encryption & decryption
  - Very fast
  - “keys to the castle”



Asymmetric Key Encryption

- Uses 2 Keys
  - Public Key – Share (encrypt)
  - Private Key – Protect (decrypt)
- Separate but mathematically related

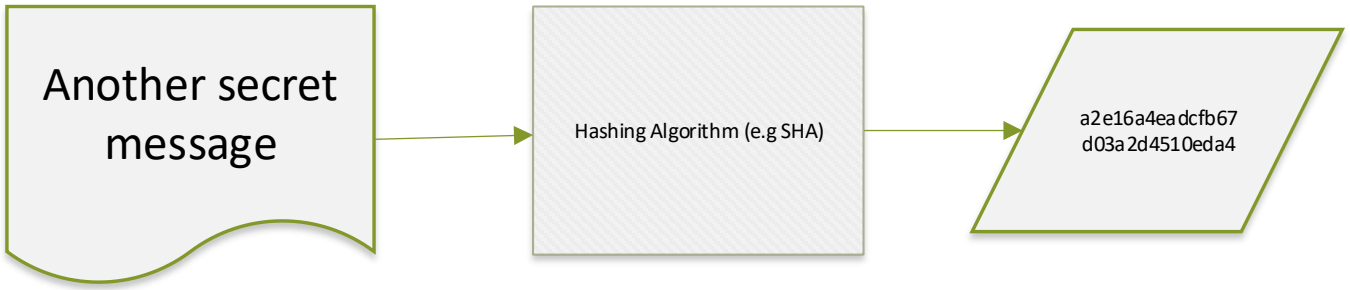


Digital Signatures

- Ensure message integrity
  - Certificates can include a set of signing keys
    - Public key for verification
    - Private key proves integrity
- Ensure that message creator
  - Can not deny having sent the message
  - Was actually created by claimed

Message Hashing

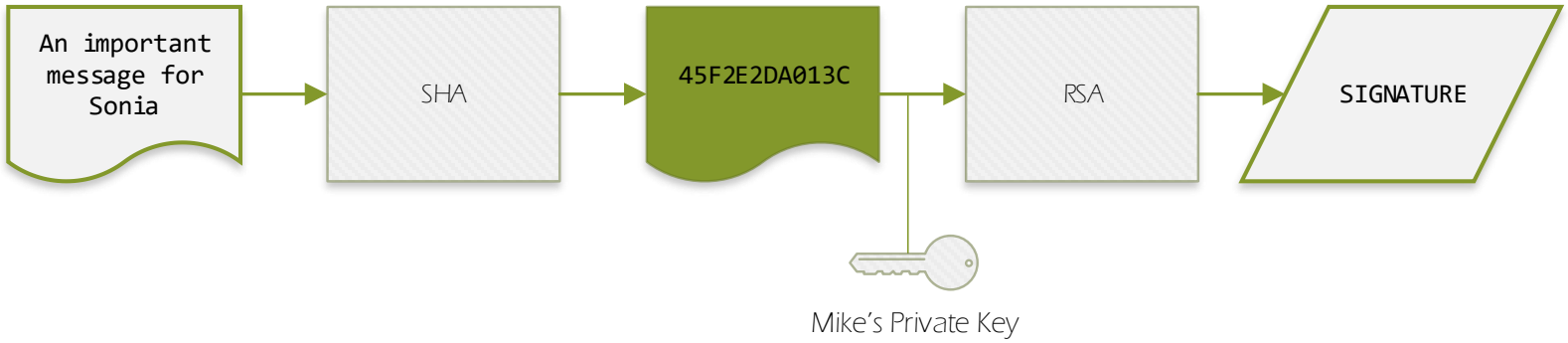
- Creates small, unique “signature” from data
- Is a one-way function
- Small changes input => Big changes output



Putting it all Together

- Mike wants to send a message to Sonia

1. Create a message
2. Create a hash of message
3. Use private signing key to sign hash

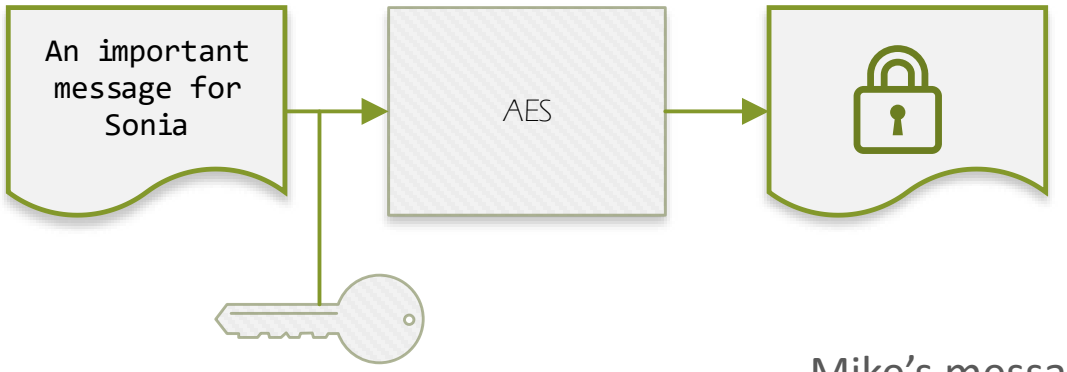


Hashing ensures integrity – message can not be changed without us noticing

Putting it all Together

- Mike wants to send a message to Sonia

4. Create one time session key
5. Symmetric key encrypt message using session key



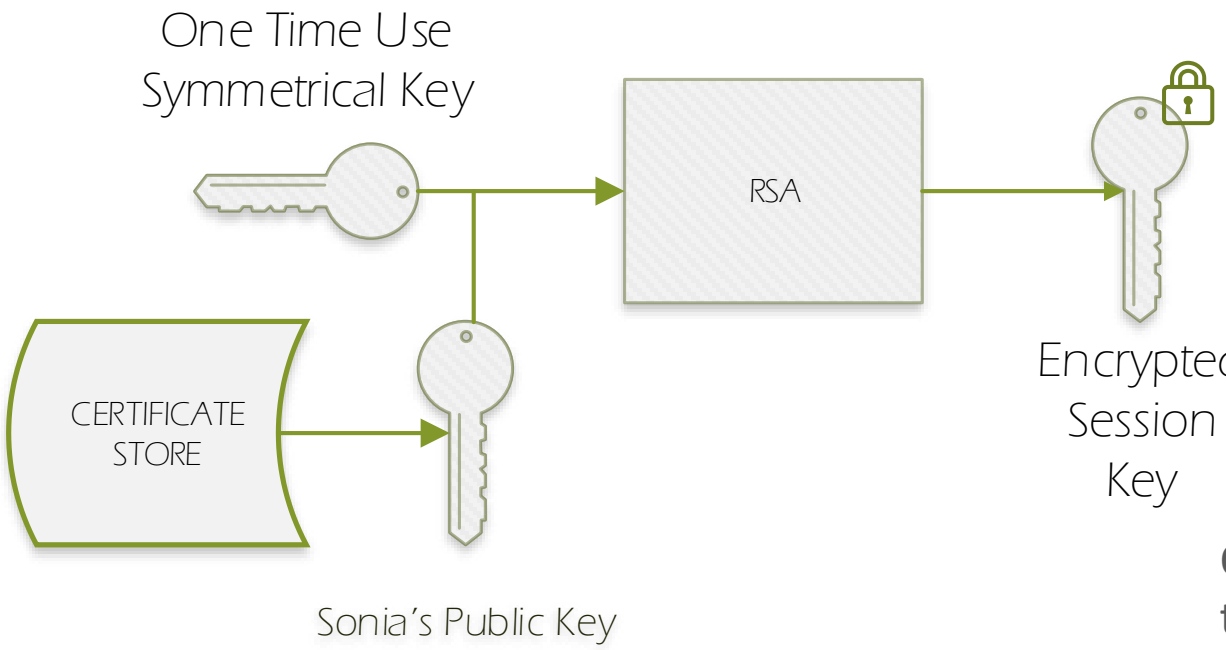
One Time Use Symmetrical Key

Mike's message content is now encrypted

Putting it all Together

- Mike wants to send a message to Sonia

6. Encrypt session key with SONIA's public key

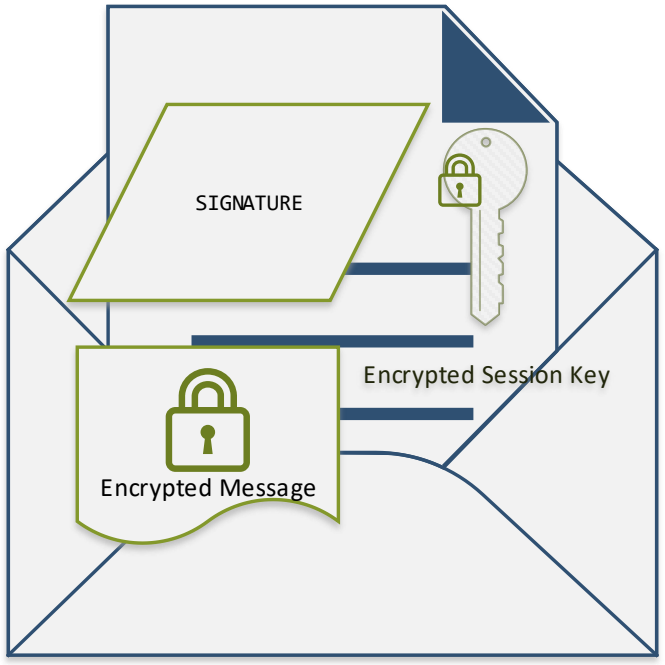


Only Sonia can decrypt this message now

Putting it all Together

- Mike wants to send a message to Sonia

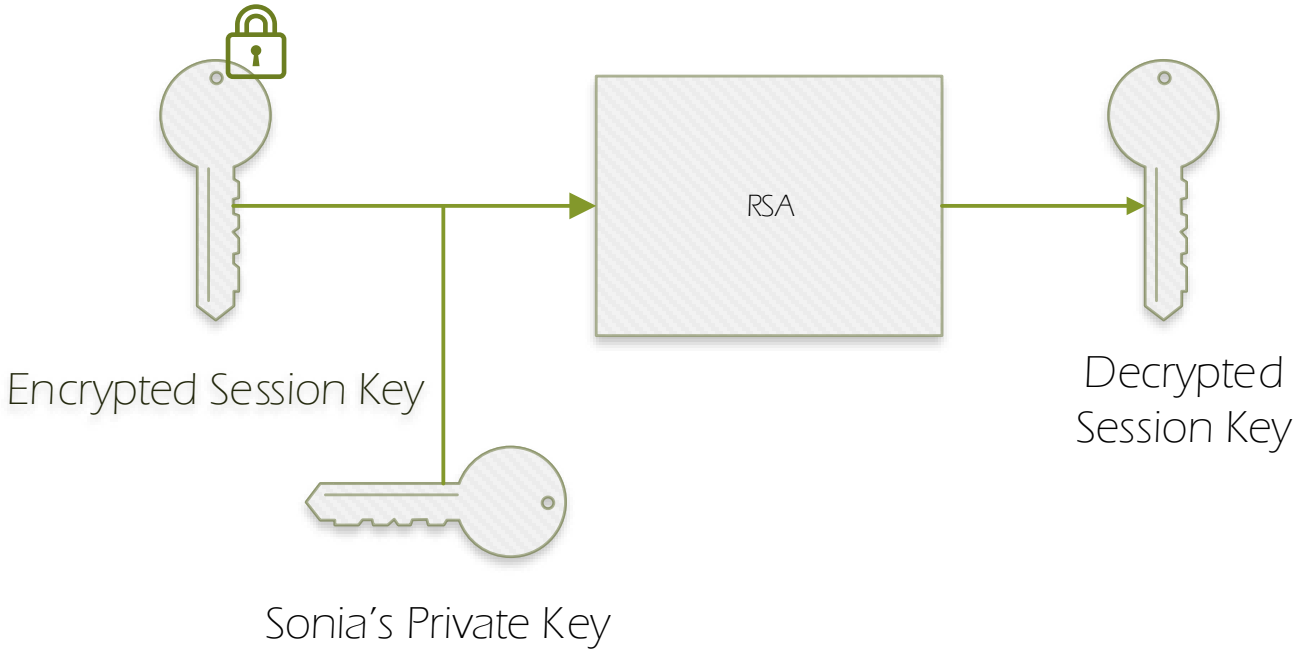
6. Mike can now send his message
   - **Signature**
   - **Encrypted message**
   - **Encrypted session key**



Putting it all Together

- Sonia wants to decrypt Mike's message

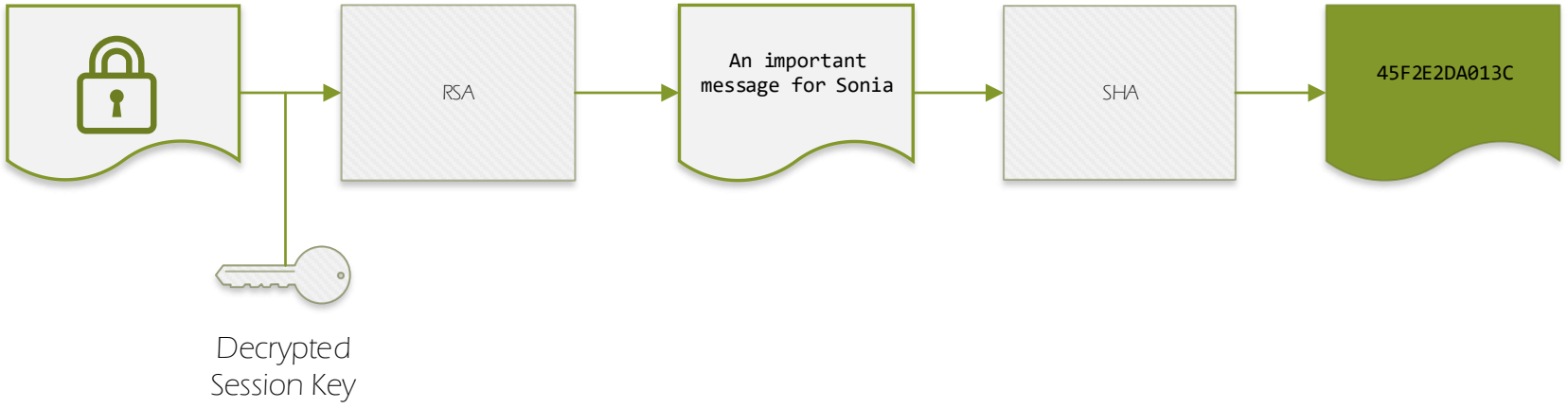
1. Decrypt the session key using her private key



Putting it all Together

- Sonia wants to decrypt Mike's message

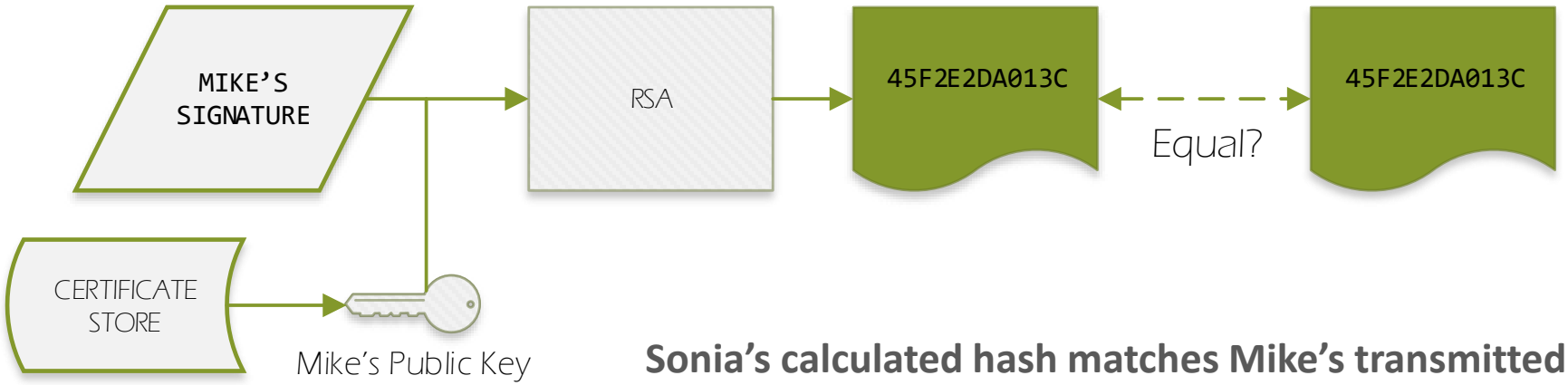
3. Decrypt the message using the decrypted session key
4. Compute a hash on the decrypted message content



Putting it all Together

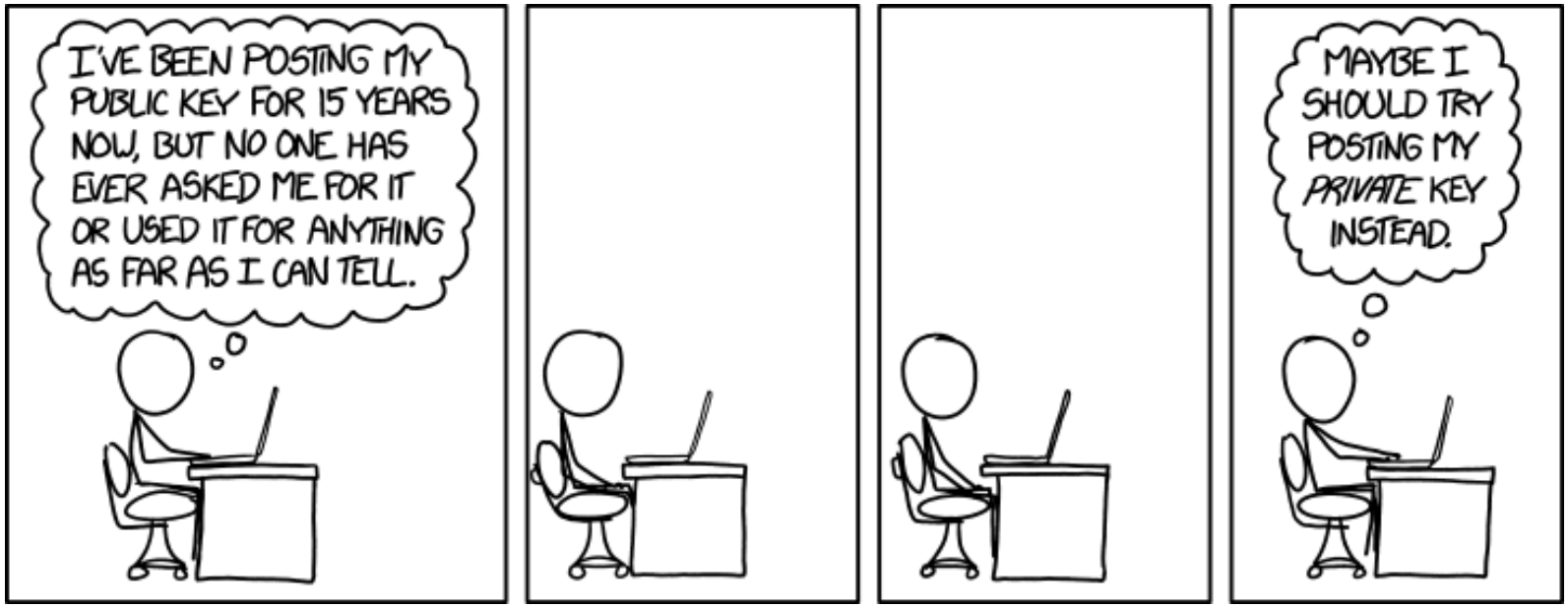
- Sonia wants to decrypt Mike's message

5. Decrypt Mike's signature
6. Compare Mike's transmitted signature hash to Sonia's calculated hash



Sonia's calculated hash matches Mike's transmitted!

1. Mike's message was never compromised
2. Mike is the author of the message



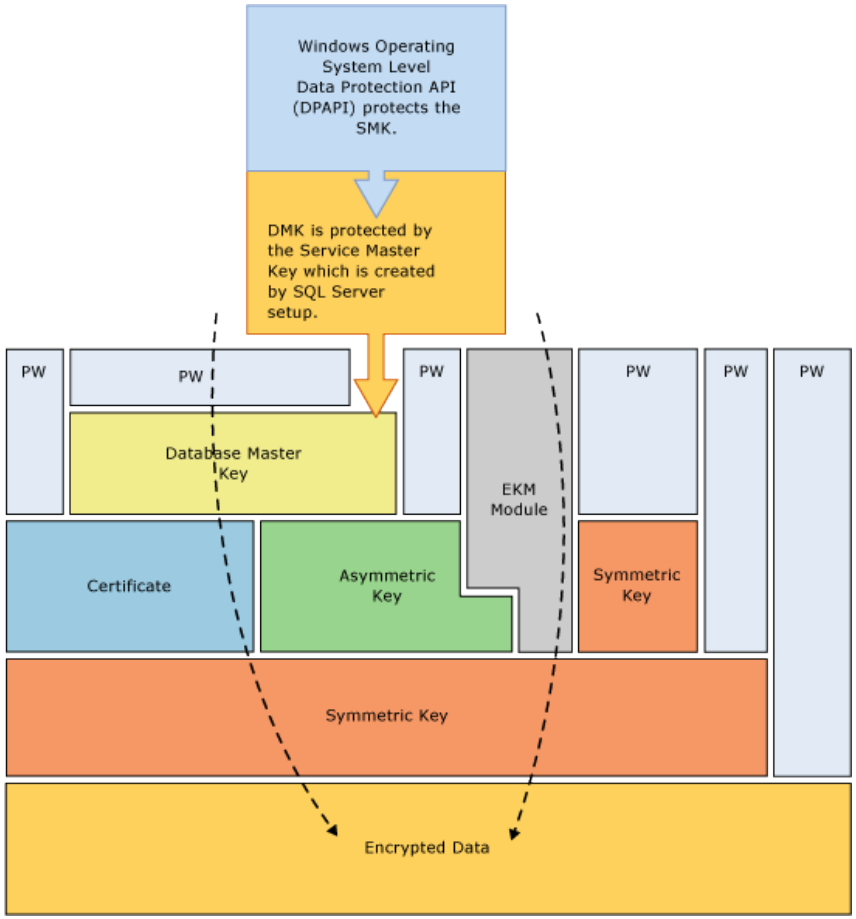
https://xkcd.com/1553/

SQL Server Encryption

SQL Server Encryption

- Column Level Encryption Functions
- Transparent Data Encryption
- Always Encrypted
  - Primarily for data at rest
  - Encryption is not access control

Encryption Hierarchy



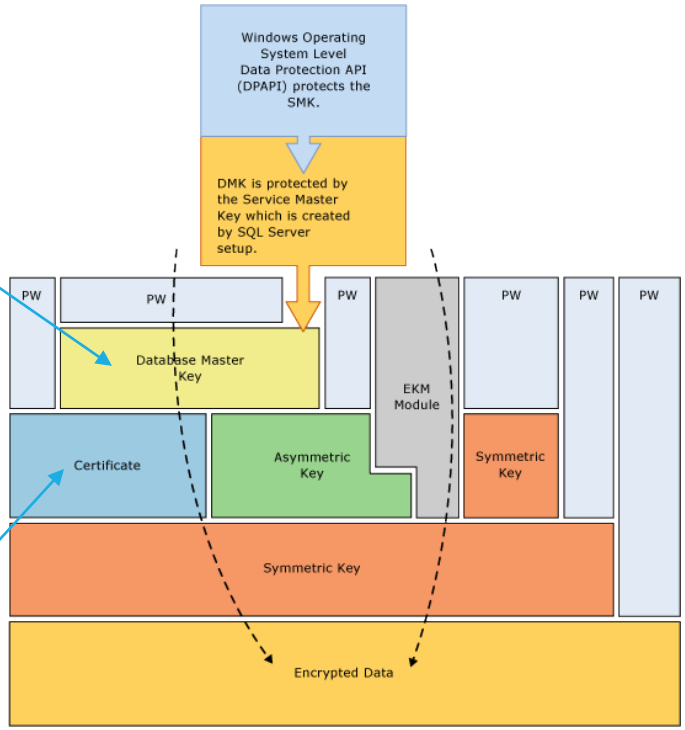
Column Level Encryption

1. Create the database master key

```
CREATE MASTER KEY
ENCRYPTION BY PASSWORD = '*******'
```

2. Create a self-signed Certificate

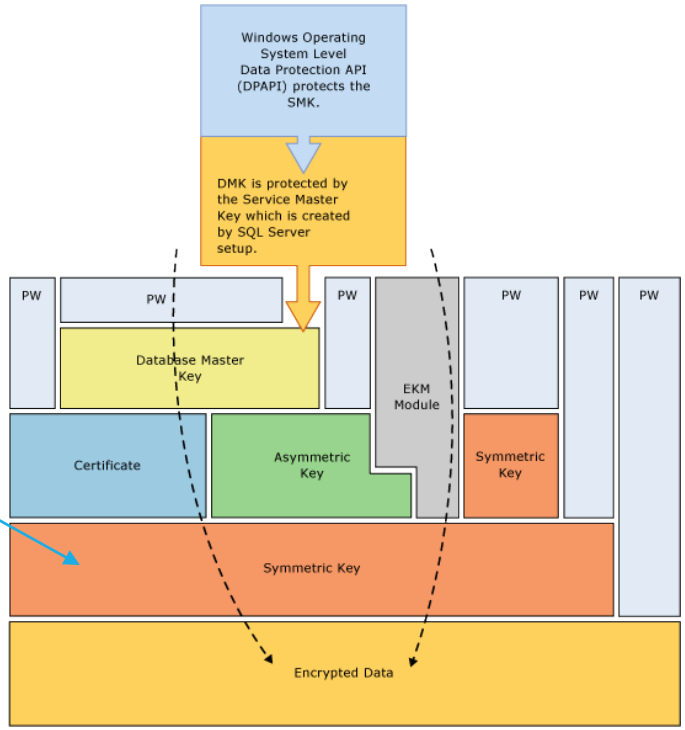
```
CREATE CERTIFICATE PhiProtect
WITH SUBJECT = 'Protect DR PHI'
```



Column Level Encryption

3. Create a symmetric key

```
CREATE SYMMETRIC KEY SymKey1
WITH ALGORITHM = AES_256
ENCRYPTION BY CERTIFICATE PhiProtect;
```



Column Level Encryption

4. Encrypt some data:
   - Open a session key
   - Use session key in encryption function
   - Close the session key

```
OPEN SYMMETRIC KEY SymKey1
DECRYPTION BY CERTIFICATE PhiProtect

INSERT MyEtlTable
SELECT [PatientID],
ENCRYPTBYKEY(KEY_GUID('SymKey1'),[Name]),
ENCRYPTBYKEY(KEY_GUID('SymKey1'),[SocialSecurityNumber])
FROM [livefdb].[dbo].[HimRec_Main]

CLOSE SYMMETRIC KEY SymKey1
```

Column Level Encryption

5. Decrypt some data:
   - Open a session key
   - Use session key in decryption function
   - Close the session key

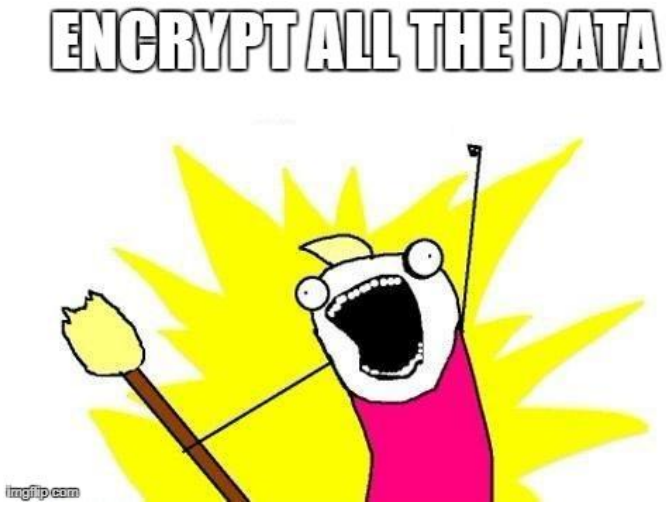
```
OPEN SYMMETRIC KEY SymKey1
DECRYPTION BY CERTIFICATE PhiProtect

SELECT PatientID,
CONVERT(VARCHAR, DECRYPTBYKEY(NameEncrypted)) as [Name],
CONVERT(VARCHAR, DECRYPTBYKEY(SsnEncrypted)) as [SocialSecurityNumber]
FROM MyEtlTable

CLOSE SYMMETRIC KEY SymKey1
```

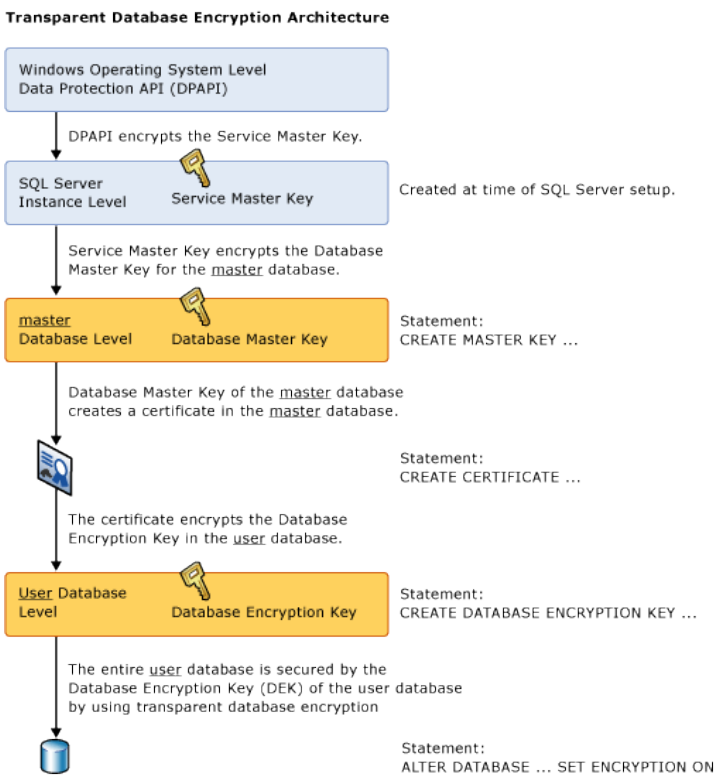
Transparent Data Encryption

- Encrypts data stored in physical files
- Encrypts transaction log files
- Data can not be restored without master key and certificate
- Queries are unaffected, encryption/decryption is transparent



Transparent Data Encryption

1. Create database master key
   - Stored within master database
2. Create a certificate protected by master key
3. Create database encryption key and protect
4. Set database to use encryption



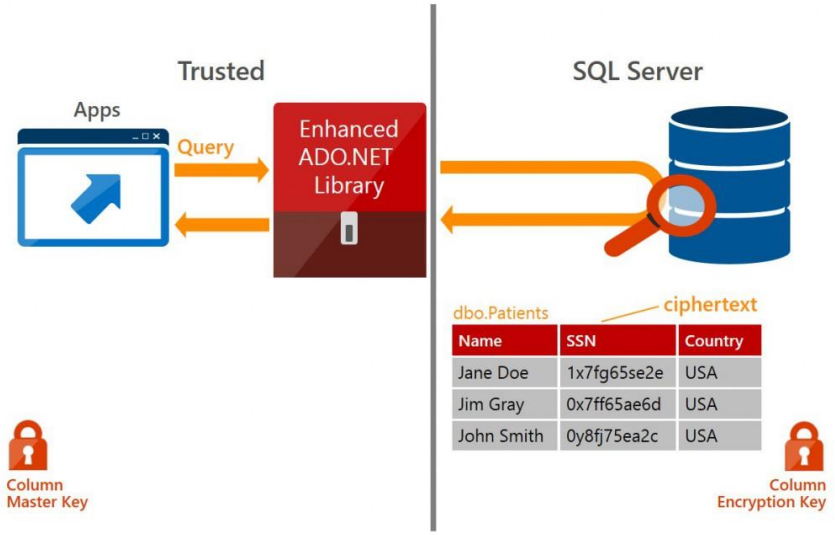
HTTPS://DOCS.MICROSOFT.COM/EN-US/SQL/RELATIONAL-DATABASES/SECURITY/ENCRYPTION/TRANSPARENT-DATA-ENCRYPTION

Transparent Data Encryption

```
USE master;
GO
CREATE MASTER KEY ENCRYPTION BY PASSWORD = '****************';
go
CREATE CERTIFICATE EtlDbCert WITH SUBJECT = 'Certificate for MyEtlDatabase';
go
USE MyEtlDatabase;
GO
CREATE DATABASE ENCRYPTION KEY
WITH ALGORITHM = AES_128
ENCRYPTION BY SERVER CERTIFICATE EtlDbCert;
GO
ALTER DATABASE MyEtlDatabase
SET ENCRYPTION ON;
GO
```

Always Encrypted

- Introduced with SQL Server 2016
  - Available in all versions SP1
- Client side technology
- Separates:
  - Data owners
  - Database admins
- Column level
  - Encryption keys
  - Master keys



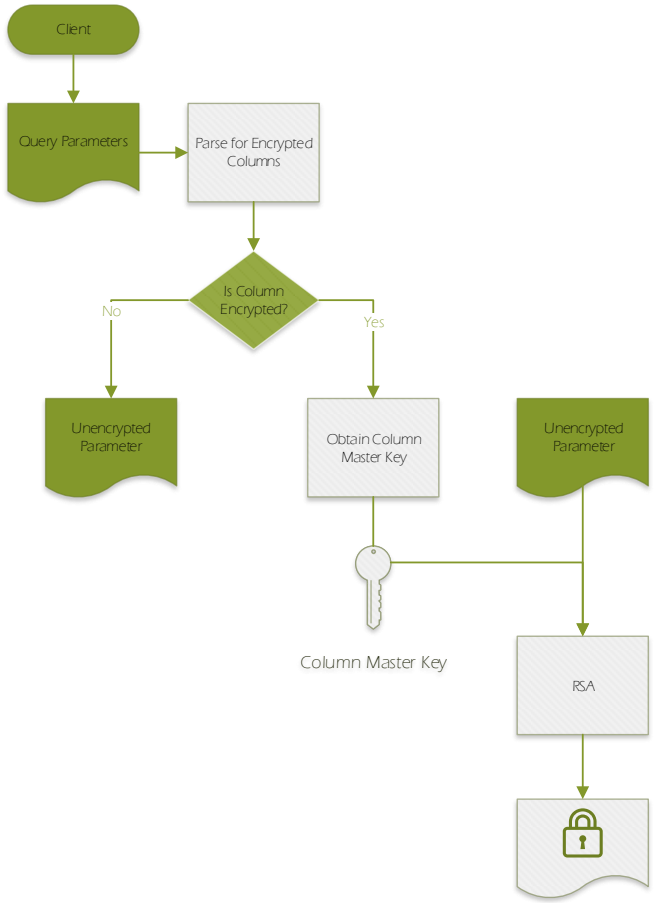
Always Encrypted

1. Use a wizard!!
2. Determine which columns to encrypt
3. Determine encryption type
   - Deterministic
   - Random
4. Generate Master Key
   - Windows Certificate Store
   - Azure Key Vault

Always Encrypted

- Client side must use enabled driver
- Must have access to certificate store
- Connection string contains:

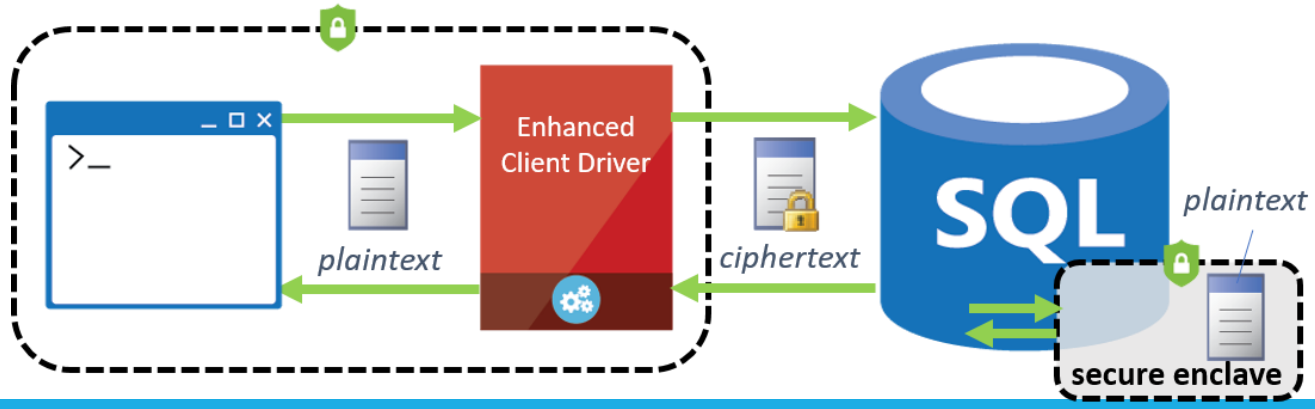
```
Column encryption setting = enabled
```



Always Encrypted Security Enclaves

Preview technology from Microsoft

- Major limitation is equality only comparison
- Security Enclaves
  - Like and Range Comparisons
  - Datatype conversions
  - Sorting operations



SQL Encryption Evaluation

Column level encryption allows for

- Fine grain control
- Lots of encryption options
- Data “in-flight” is never protected

Transparent Data Encryption

- Encrypts database files and logs
- Encrypts all data

Always Encrypted

- Fine grain control
- Uses client for encryption
- Not backwards compatible

Other things to consider

Connection Level Security

- Encrypt keyword
- Server certificate

Reporting Services

- Report server HTTPS option
- Report history snapshots
- Report execution snapshots

Adding Logging Capabilities

- Anyone can access encrypted data
- Log when data is decrypted

Proof of Concept

LET'S APPLY WHAT WE'VE LEARNED INTO A DEMO APPLICATION

The Goals

Identify PHI being used in the system

- Load PHI meta data from MT
- Cross reference to DR tables / columns

Add PHI Encryption / Decryption

- Use PKI and an internal Certificate Authority
- Encrypt PHI so that all users can have access
- Allow ‘special users’ to decrypt data for use
- Provide a method to create de-identified data (i.e. hashes)

Allow system to be ‘extended’

- Allow any application to provide encrypted data
- Allow any application to decrypt

The Goals

1. **Identify PHI being used in the system**

   Use MT metadata to identify

2. **Add Encryption / Decryption**

   Use hybrid Public Key Encryption (PKI)

   (Single certificate implementation)

3. **Allow system to be ‘extended’**

   Lots system / software can send us data

The Goals

Identify PHI being used in the system

- Load PHI meta data from MT
- Cross reference to DR tables / columns

```
1  SELECT  t.TableName, c.ColumnName, dt.[Name]
2  FROM [dbo].[DrTable_Columns] c
3  JOIN [dbo].[PhiElementsFoc] foc
4      ON c.ColumnObjectClass = foc.[Object]
5      AND c.ColumnRecord = foc.Record
6      AND c.ColumnField = foc.Field
7  JOIN [dbo].[DrDataType_Main] dt
8  on dt.DrDataTypeID = c.ColumnDataType_DrDataTypeID
9  JOIN [dbo].[DrTable_Main] t
10     ON c.DrTableID = t.DrTableID
11     AND c.SourceID = t.SourceID
12
```

00 % | 0 changes | 0 authors, 0 changes

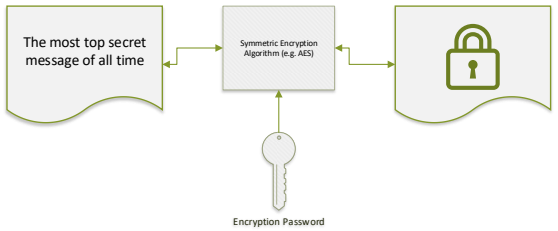
T-SQL | Results | Message

	TableName	ColumnName	Name
1	HimRec_Main	Age	varchar
2	HimRec_Main	Birthdate	datetime
3	HimRec_Main	BirthdateComputed	datetime
4	HimRec_Main	Name	varchar
5	HimRec_Main	SocialSecurityNumber	varchar

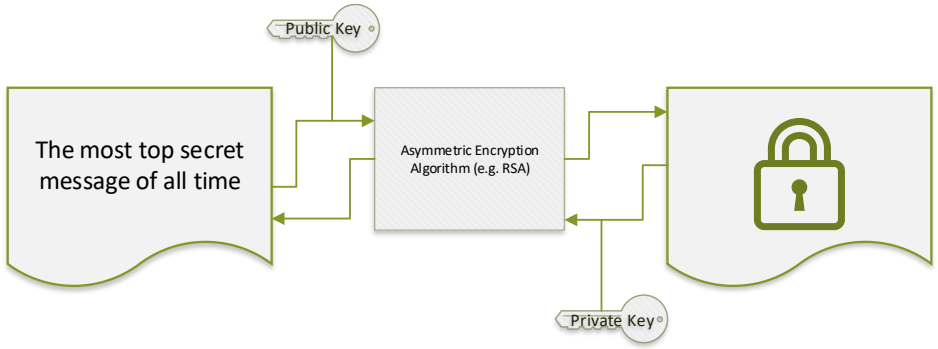
The Goals

PHI Encryption / Decryption Discussion

- Symmetric Key Encryption
  - Fast
  - Can encrypt / decrypt with same key



- PKI For Everything
  - Anyone can encrypt
  - Only special users can decrypt
  - Slow for large amounts of data
    - DR has LOTS of data!!



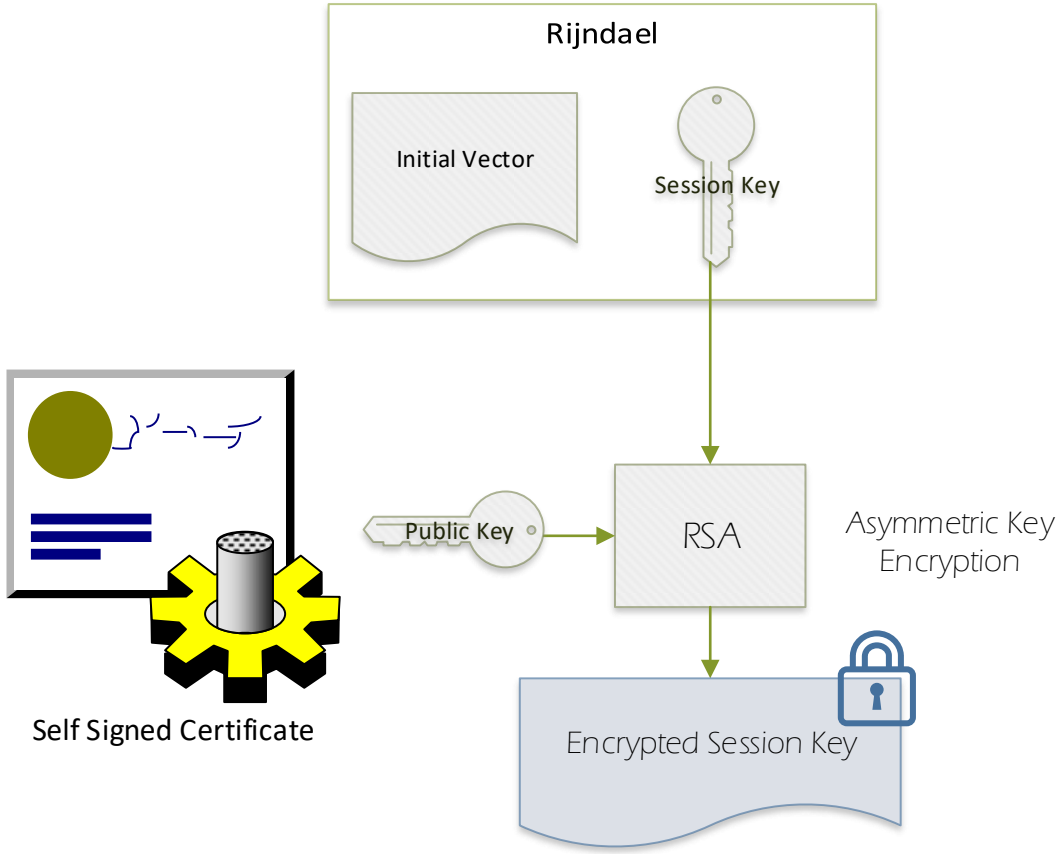
- Database Encryption
  - Difficult to share a subset of data w/o re-encrypting

The Implementation

DataProtect Assembly

ProtectBuffer

- Generate Rijndael IV and Key
- Asymmetric key encryption of Rijndael key / vector

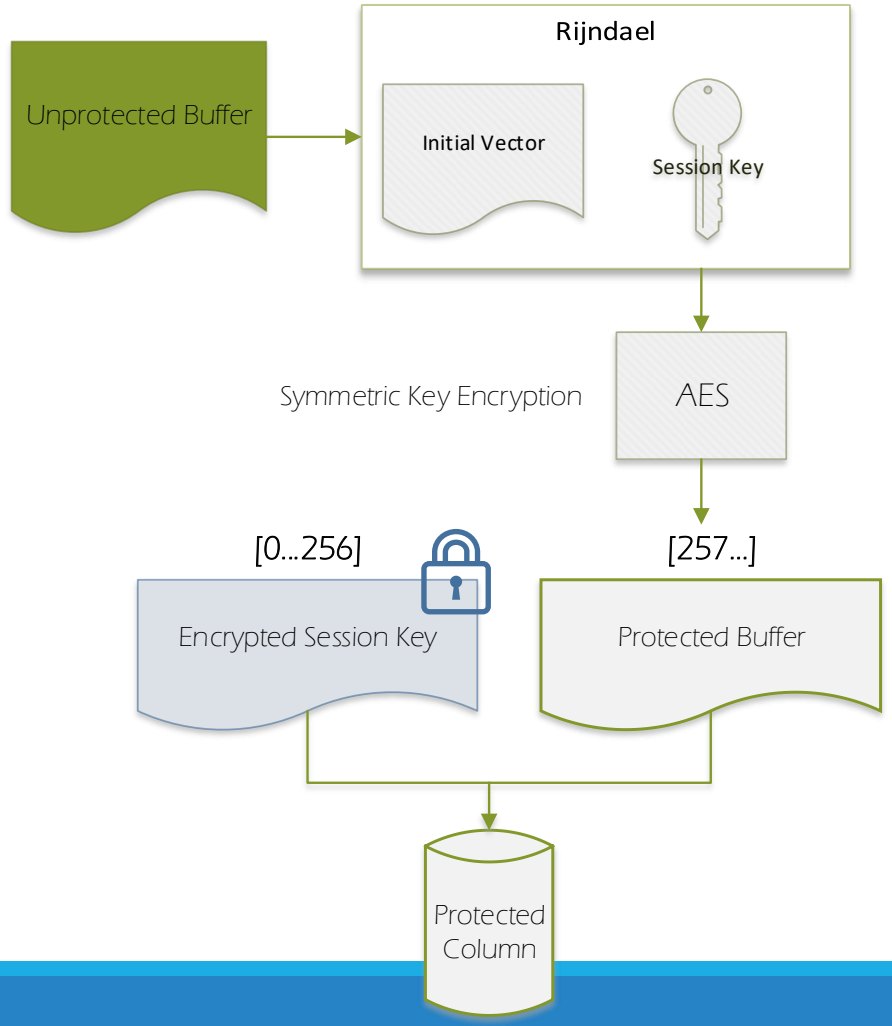


The Implementation

DataProtect Assembly

ProtectBuffer

- Use AES on data w/ Rijndael IV + Key
- Prefix encrypted session key

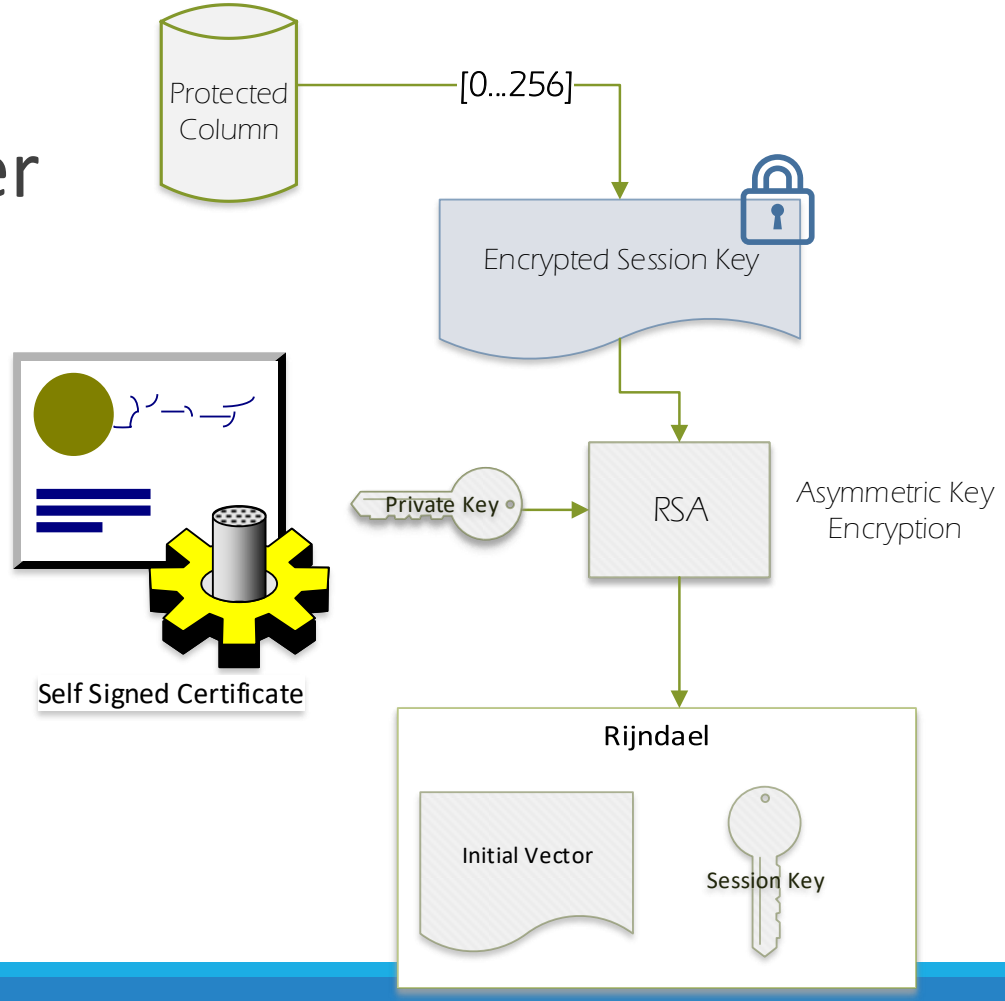


The Implementation

DataProtect Assembly

UnprotectBuffer

- Read prefix on protected column
- Decrypt using certificate **PRIVATE** key

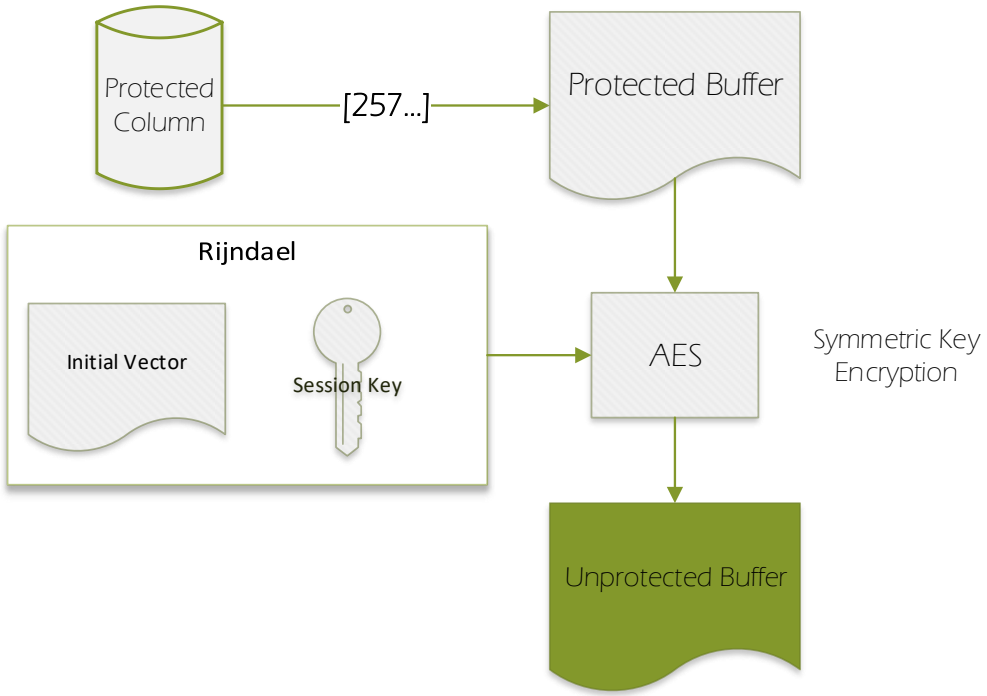


The Implementation

DataProtect Assembly

UnprotectBuffer

- Read protected remainder
- Decrypt using Rijndael IV + key



The Implementation

DataProtectDb

- Moves functions from DbProtect into SQL Database
- Uses SQL CLR to run .Net functions
  - grant assembly EXTERNAL_ACCESS in order to read certificates
  - Use self-signed certificate to sign assembly

- Putting it all together
  - Get PHI containing columns from our reference table
  - Protect PHI columns using dbo.Protect
  - Compute Hashes on columns used for lookups
  - Search for a row by comparing a hash calculated on an input

The Implementation

Other things to consider:

- Share the public key! Anyone can send us data!
- Anyone can view anonymized data – take advantage.
- How do we handle a decryption event / request?
- What are the trade-offs?

Thank you for your time!

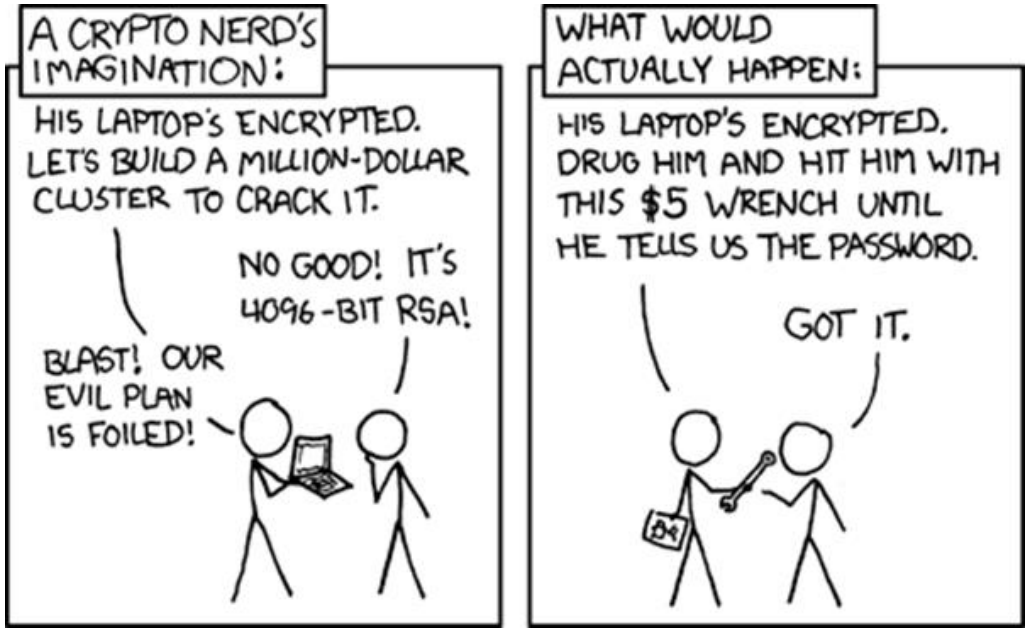
blue elm

Your Data ▪ Only Better

@BlueElm

/blueelm

blueelm.com



https://xkcd.com/538/

Sources / References:

Privacy and Security: Protect Electronic Health Information
https://customer.meditech.com/en/d/bestpractices/otherfiles/bp1460privacyandsecurity.pdf

Massachusetts 940 CMR 27.00: Safeguard of Personal Information
https://www.mass.gov/files/documents/2016/08/rv/940-cmr-27-00.pdf

An Idiots Guide to Public Key Infrastructure
https://www.giac.org/paper/gsec/2171/idiots-guide-public-key-infrastructure/103692

SQL Server Encryption Hierarchy
https://docs.microsoft.com/en-us/sql/relational-databases/security/encryption/encryption-hierarchy

Feature Spotlight: Transparent Data Encryption
https://blogs.msdn.microsoft.com/sqlsecurity/2016/10/05/feature-spotlight-transparent-data-encryption-tde/

SSMS Encryption Wizard – Enabling Always Encrypted in a Few Easy Steps
https://blogs.msdn.microsoft.com/sqlsecurity/2015/10/31/ssms-encryption-wizard-enabling-always-encrypted-in-a-few-easy-steps/

Enabling Confidential Computing with Always Encrypted using Enclaves (Early Access Preview)
https://blogs.msdn.microsoft.com/sqlsecurity/2017/10/05/enabling-confidential-computing-with-always-encrypted-using-enclaves-early-access-preview/

Encrypting Personal Health Data
http://blog.brucejackson.info/2009/09/encrypting-personal-health-data-part-1.html