

**DUKPT**

Índice

[1. Derived Unique Key Per Transaction (DUKPT) 3](#_Toc39500667)

[1.1. Visão Geral 3](#_Toc39500668)

[1.2. DES, 3DES e AES e DUKPT 4](#_Toc39500669)

[1.3. DUKPT Keys 4](#_Toc39500670)

[1.4. Base Derivation Key (BDK) 4](#_Toc39500671)

[1.5. Chave Initial Key (IKEY ou IK) 6](#_Toc39500672)

[1.6. Key Serial Number 6](#_Toc39500673)

[1.7. Transaction Keys 7](#_Toc39500674)

[1.8. Processamento pelo Adquirente 7](#_Toc39500675)

[1.9. Resumo das Operações DUKPT 8](#_Toc39500676)

[1.10. AES DUKPT 9](#_Toc39500677)

[1.11. Variant DUKPT 11](#_Toc39500678)

[1.12. DUKPT para Point-to-Point Encryption e Mobile Acceptance 12](#_Toc39500679)

[1.13. payShield 9000 na DUKPT 13](#_Toc39500680)

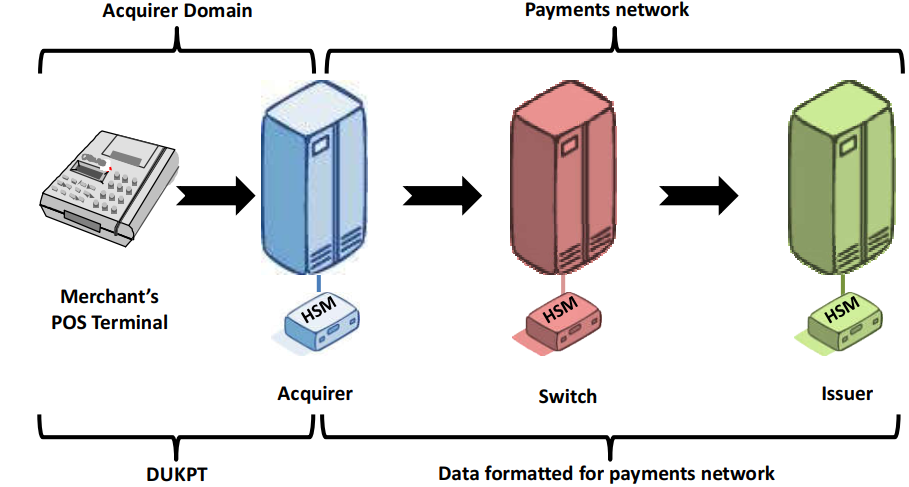
[2. Release Notes 14](#_Toc39500681)

# Derived Unique Key Per Transaction (DUKPT)

## Visão Geral

DUKPT (Derived Unique Key Per Transaction) é um esquema para gerenciar chaves de criptografia TDES (Tripe-DES) em um ambiente de pagamento com cartão. Tradicionalmente, é usado em terminais POS (ponto de venda) na América do Norte, mas a adoção está crescendo em outras regiões e para outras aplicações - e, portanto, o DUKPT está se tornando cada vez mais importante. O esquema DUKPT está especificado na norma ANSI X9.24-1: 2009.

O DUKPT é usado para criptografar PIN blocks, criptografar outros dados e autenticação de mensagens (MACing). DUKPT é normalmente utilizado entre comerciantes e seus adquirentes. O Adquirente ré empacotará os dados da transação para passar pela rede de pagamentos (que usa o gerenciamento de chaves Master/Session) antes de passá-los para uma central de pagamentos



A força da tecnologia DUKPT está no fato de que uma única chave é gerada e utilizada para cada transação, de modo que, se uma chave de transação for comprometida, ela não poderá ser usada para atacar transações anteriores daquele terminal ou transações em qualquer outro terminal. Além disso, o gerenciamento de chaves no ambiente DUKPT é simplificado por ter uma única chave mestra que pode gerenciar todo um estado de terminais.

A técnica DUKPT se utiliza do "número de série da chave, key serial number/KSN" não secreto e uma chave do tipo BDK, Based Derivation Key. Em cada transação, o PIN pad utiliza uma chave exclusiva com base na chave anterior e no número de série da chave, que contém um contador de transações. Ele criptografa o PIN com essa chave e, em seguida, retorna o PIN criptografado e o número de série da chave ao adquirente. Em um HSM, a chave gerada pelo PIN pad é derivada dinamicamente e independentemente do PIN pad, usando o BDK original junto com o número de série, KSN, da chave fornecido pelo PIN pad.

A mesma chave BDK pode ser usado por milhares de PIN pads, porque cada PIN pad possui um número de série exclusivo. Portanto, cada PIN pad produz uma chave exclusiva para cada transação, e um ataque criptográfico bem-sucedido em um PIN pad não terá efeito em nenhum outro. O adquirente precisa gerenciar apenas um número relativamente pequeno de BDKs, e o algoritmo para derivar uma determinada chave de transação é projetado de maneira a exigir muito pouca sobrecarga em um HSM.

O Host tem a responsabilidade de manter as chaves BDKs. Para cada transação, o Host verifica se o número de série fornecido pelo PIN pad é válido e extrai do armazenamento interno a chave BDK criptografado apropriada e identificada pela parte mais à esquerda do número de série. O host controla a geração da chave BDK.

## DES, 3DES e AES e DUKPT

O payShield 9000 HSM suporta os algoritmos de criptografia do tipo DES, 3DES e AES para as chaves DUKPT.

## DUKPT Keys

Três níveis de chaves são empregados no DUKPT:

- Base Derivation Key (BDK) - uma chave mestra 3DES de propriedade do adquirente. A chave BDK é utilizada em um grande número de terminais - talvez todos os terminais que o fornecedor envia, ou para um modelo de terminais ou para um intervalo de números de série.

- Initial Key (IKEY, IK ou IPEK) - uma chave 3DES exclusiva de um terminal. O IKEY é usado para iniciar a sequência de chaves de transação e é descartado pelo terminal.

- Transaction Key - gerada dentro do terminal. As chaves para criptografar o PIN, dados e MACing são derivadas da Transaction Key. Cada transação é fornecida com uma chave exclusiva para proteger seus dados. Quando os dados criptografados são recebidos pelo Adquirente, o Adquirente derivará a mesma chave de transação usando o mesmo processo que o terminal utilizado para derivar a chave de criptografia.

Eles são descritos nas seções a seguir.

Pode-se observar neste processo que não há exigência de que o Adquirente e o terminal troquem chaves - exceto no caso improvável de um terminal gerar um milhão de chaves de transação e, portanto, exigir um novo IKEY.

Espera-se que o DUKPT possa gerenciar as chaves AES no futuro.

## Base Derivation Key (BDK)

A chave BDK é uma chave 3DES de comprimento duplo que geralmente é gerada e é de propriedade do adquirente e é utilizada em muitos terminais. (Se a chave BDK pertence a uma organização que não seja o Adquirente, ele precisará ser distribuído ao Adquirente para permitir que ele processe transações. A chave BDK também precisará ser distribuída a qualquer outra organização envolvida na geração de IKEYs.) Geralmente, várias chaves BDKs são mantidos para permitir diferentes famílias ou grupos de terminais. A distribuição BDK pode ser feita:

- de forma eletrônica, com o BDK protegido por uma Zone Master Key (ZMK), ou

- na forma de componentes impressos, com suportes de componentes separados se unindo para permitir que o BDK seja formado a partir de seus componentes.

Uma chave BDK armazenada de forma correta, deve ser protegido por criptografia usando uma KEK (key Encryption Key) apropriada sempre que existir fora de um HSM. Nos casos em que os HSMs de pagamento da Thales são usados, as BDKs são protegidas pela LMK (Local Master Key) do HSM.

**BDK suportadas pelo payShield 9000**

Os seguintes comandos de Host são utilizados para gerenciar as chaves BDKs.

|  |  |
| --- | --- |
| **Comando** | **Descrição** |
| A0 | Gera uma chave BDK de forma randômica e retorna a mesma criptografada pela LMK. |
| A6 | Aceita a BDK criptografada sobre a chave Zone Master Key (ZMK) e traduz ela para a chave LMK |
| A8 | Traduz a chave a BDK criptografada pela LMK para a chave ZMK. |

A revisão mais recente do padrão DUKPT (X9.24-1:2009) define dois métodos diferentes para derivar chaves de autenticação e criptografia de dados:

- O método **bidirecional** usa uma única chave para proteção dos dados de terminal para host e dados de host para terminal. Os tipos 1 e 3 da chave BDK suportam o método bidirecional. (Observe que uma BDK do tipo 3 usa a variante 'criptografia de PIN/PIN encryption' para derivar a chave de criptografia de dados e, portanto, só pode ser usado para executar operações de criptografia de dados. Um BDK-3 não pode ser usado para executar operações relacionadas a PIN).

- O método **unidirecional** usa duas chaves: uma chave para proteger os dados do terminal até o host e outra chave para proteger os dados do host até o terminal. Os tipos 2 e 4 do BDK suportam o método unidirecional.

Mais informações sobre o uso dos vários tipos de BDK são fornecidas mais adiante neste capítulo na seção Tipos de chave Thales para DUKPT.

**Suporte para ZMK (Zone Master Key)**

O payShield 9000 HSM suporta chaves do tipo Zone Master Keys (ZMK) de comprimento único, 16 caracteres hexadecimais (64 bits); e chaves Zone Master Key (ZMK) de comprimento duplo, 32 caracteres hexadecimais (128 bits). O conjunto de comandos DUKPT ignora o parâmetro S/D (single/double) de comprimento definido nas configurações de segurança do payShield 9000.

.

## Chave Initial Key (IKEY ou IK)

A chave IKEY (originalmente referido como IPEK - Initial PIN Encryption Key) é exclusivo do seu terminal. O IKEY é calculado a partir de:

- BDK

- o número de série da chave (KSN), exclusivo do terminal - veja abaixo.

O comando do host payShield 9000 **A0** pode gerar um IKEY e exportá-lo sob uma TMK (Terminal Master Key) ou ZMK (Zone Master Key).

Uma vez criado, o IKEY é instalado no terminal. (O IKEY também é recriado transitoriamente pelo Adquirente ao processar transações do terminal para obter a mesma chave de transação que o terminal usou para criptografar seus dados.)

## Key Serial Number

A chave KSN possui um tamanho máximo de 80 bits e possui a seguinte estrutura:

|  |  |  |
| --- | --- | --- |
| **Elemento** | **Tamanho** | **Descrição** |
| Key Set Identifier | 5-9 Hex caracteres (20-36 bits) | Identifica a BDK para ser utilizado pelo terminal. |
| Sub-key Identifier | 1 Hex caracteres (4 bits) | Atualmente definido com o valor 0. |
| Device Identifier | 2-5 Hex caracteres (8-20 bits) | Identificação única, como por exemplo, um número de série para o terminal. |
| Transaction Counter | 1 bit + 5 Hex caracteres (21 bits) | Contador de número de PIN que foram criptografados desde quando o terminal foi inicializado. |

Os três primeiros elementos da tabela acima formam o número de série inicial da chave, Initial Key Serial Number(IKEI), e não são alterados durante a vida útil do terminal (a menos que um novo IKEY seja carregado por qualquer motivo).

Geralmente, apenas 64 bits do KSN são usados, com o KSN preenchido com caracteres hexadecimais "F" à esquerda. Nesse esquema, o KSN teria a seguinte estrutura:

- **Preenchimento** - 4 caracteres hexadecimais "F", 16 bits.

- **Identificador do conjunto de chaves** - 6 caracteres hexadecimais, 24 bits. Isso permite cerca de 16 milhões de BDKs diferentes.

- **Identificador do dispositivo** - 5 caracteres hexadecimais, 20 bits. Isso inclui um bit do contador de transações, deixando 19 bits para o identificador de dispositivo. Isso significa que cerca de meio milhão de dispositivos diferentes podem ser gerenciados pelo Identificador de dispositivo. Dois terminais com a mesma chave de derivação de base e identificadores de sub-chave podem receber o mesmo identificador de dispositivo. Como o terminal empacota o bit mais à esquerda do contador de transações como o bit mais à direita do identificador de dispositivo, esse campo é sempre uniforme (o bit mais à direita é definido como zero).

- **Contador de transações** - 5 caracteres hexadecimais, 20 bits (mais o bit incluído no identificador do dispositivo). O contador de transações é fornecido pelo terminal para identificar uma transação específica. É usado por um HSM para calcular a chave PIN real. O bit mais à esquerda é fornecido como o bit mais à direita do identificador do dispositivo, portanto, o comprimento desse campo é de 20 bits (5 dígitos hexadecimais). Isso permite cerca de 1 milhão de transações antes que um novo IKEY seja necessário - um limite que dificilmente será atingido.

O terminal não pode aceitar um número de série com mais de 20 caracteres; portanto, o host garante que o comprimento total dos três primeiros campos não exceda 15 caracteres.

O host também fornece a um HSM um KSN descriptor de três caracteres, que define o comprimento (em caracteres) de cada um dos três primeiros campos. Ele está incluído no KSN no Host storage e é usado pelo Host para identificar a chave de derivação básica. O descritor KSN consiste em:

- esquerda – tamanho que identifica a BDK

- meio – tamanho que identifica a sub-chave

- direito – tamanho que identifica o dispositivo

## Transaction Keys

Quando o IKEY é instalado no terminal, ele calcula até 21 "Future Keys". Essas chaves de transação são as chaves que serão usadas na criptografia de futuras transações. O cálculo dessas chaves envolve o valor do contador de transações, que é incrementado para cada transação.

Quando o lote inicial de Future Keys tiver sido derivado, o IKEY não será mais necessário e será excluído pelo terminal.

Quando uma transação está sendo processada, a próxima Transaction Key disponível é usada. As chaves usadas para criptografia de PIN block, MAC e criptografia de dados são derivadas dessa chave de transação.

O KSN também é modificado incrementando o contador de transações.

O terminal DUKPT envia seus dados criptografados e o KSN, juntamente com outros dados da transação, ao Adquirente.

À medida que cada Transaction Key é usada, ela é excluída pelo terminal e substituída por uma nova Future Transaction Key. Isso significa que, mesmo que a segurança do terminal seja comprometida de alguma forma e suas chaves sejam extraídas, elas não poderão ser usadas para atacar uma transação anterior deste ou de qualquer outro terminal, porque a chave para essa transação já foi excluída e cada terminal gera chaves diferentes.

## Processamento pelo Adquirente

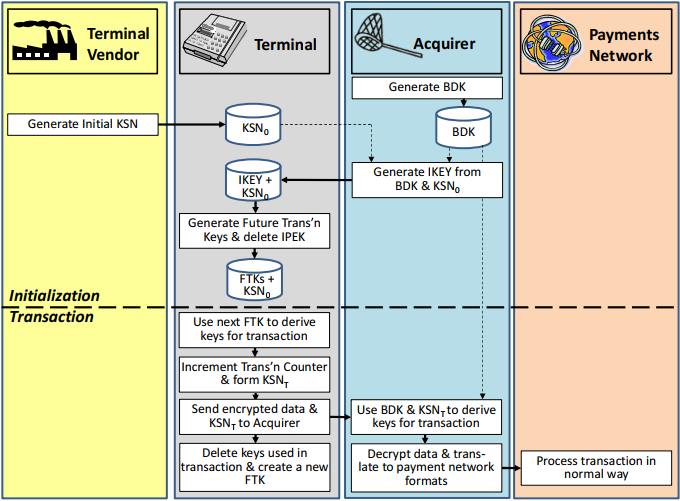
Quando os dados criptografados são enviados pelo terminal ao Adquirente, o KSN (incluindo o transaction counter) também é enviado. O Adquirente pode reconstruir a Transaction Key usada pelo terminal do KSN e o BDK apropriado (conforme identificado no KSN's Key Set Identifier).

O Adquirente precisa reembalar os dados recebidos do terminal nos formatos padrão usados pela rede de pagamentos. Isso incluirá ações como:

- Converter o DUKPT PIN Block criptografado com a chave de transação DUKPT em um dos formatos padrão de PIN block que a rede de pagamentos usa criptografados com uma chave Zone PIN Key (ZPK).

- Verificando e traduzindo MACs.

## Resumo das Operações DUKPT



Notas:

1. KSN0 = Initial Key Serial Number (com Transaction Counter = 0). Pode ser modificado pelo Adquirente antes de gerar IKEY.

2. KSNT = Key Serial Number da chave da transação, com o contador de transações incrementado)

3. As chaves BDKs mantidas pelo adquirente serão protegidos usando um HSM.

4. As operações do Adquirente mostradas aqui envolverão o uso de um HSM para várias funções criptográficas.

## AES DUKPT

O ANSI X9.24-3:2017 introduz o suporte para derivar uma chave de criptografia de chave (conhecida como DUKPT Update Key) de um BDK especificamente para transportar com segurança uma nova Initial Key (IKEY) para o terminal. Para dar suporte a isso, os comandos gerais de gerenciamento de chaves serão estendidos para dar suporte a esta nova KEK.

Em suporte ao 3DES DUKPT, o payShield 9000 suporta quatro tipos diferentes de DUKPT Base Derivation Key (BDK):

A BDK-1 implementa o ANSI X9.24-1:2009 e deriva dados bidirecionais e chaves MAC:

• Quando o HSM criptografa/descriptografa dados ou gera / verifica um MAC usando um BDK-1, ele deriva a chave de dados "nos dois sentidos" ou a chave MAC, respectivamente.

A BDK-2 implementa o ANSI X9.24-1:2009 e deriva dados unidirecionais e chaves MAC:

• Quando o HSM criptografa dados ou gera um MAC usando um BDK-2, deriva a chave de dados de "resposta" ou a chave MAC, respectivamente.

• Quando o HSM descriptografa dados ou verifica um MAC usando um BDK-2, deriva a chave de dados "request" ou a chave MAC, respectivamente.

A BDK-3 implementa o ANSI X9.24-1:2009 e deriva uma única chave de dados, mas usa as mesmas técnicas que outros tipos de BDK usam para derivar a chave PIN. O BDK-3 é suportado apenas pelos comandos de criptografia de mensagens (M0, M2 e M4) e, por uma questão de clareza, deve ser ignorado no restante desta especificação!

A BDK-4 implementa o ANSI X9.24-1:2009 e deriva dados unidirecionais e chaves MAC:

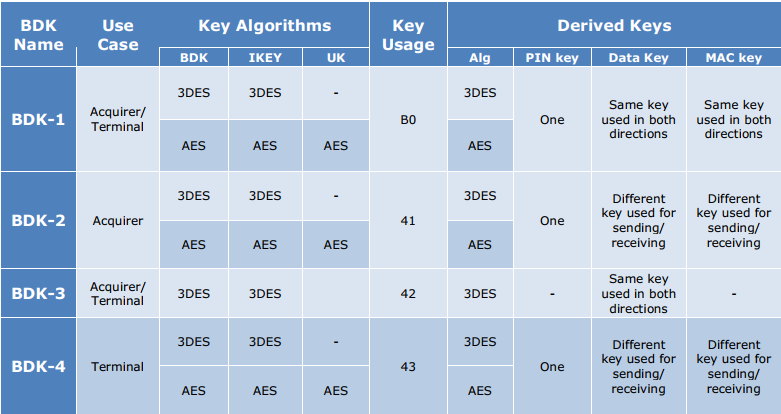
• Quando o HSM criptografa dados ou gera um MAC usando um BDK-4, deriva a chave de dados "requisição" ou a chave MAC, respectivamente.

• Quando o HSM descriptografa dados ou verifica um MAC usando um BDK-4, deriva a chave de dados de "resposta" ou a chave MAC, respectivamente.

Em resumo, são necessários três tipos diferentes de BDK (BDK-1, BDK-2 e BDK-4) para oferecer suporte total à funcionalidade padrão do 3DES DUKPT.

A partir do software v3.4a, cada um desses tipos diferentes de BDKs é estendido para permitir o uso de AES BDKs (além de continuar a suportar 3DES BDKs).

A tabela abaixo descreve todos os diferentes tipos de BDKs a serem suportados pelo HSM:



Os "casos de uso" são definidos da seguinte maneira:

"Adquirente" significa que o HSM está se comportando como um adquirente tradicional, recebendo mensagens de "solicitação" do terminal (que precisam ser descriptografadas e validadas) e gerando mensagens de "resposta" para o terminal (que precisam ser criptografadas e com MAC)

"Terminal" significa que o HSM está se comportando como um terminal, gerando mensagens de "solicitação" para um adquirente (que precisam ser criptografadas e com MAC) e recebendo mensagens de "resposta" de um adquirente (que precisam ser descriptografadas e validadas).

**Notas de implementação para AES DUKPT**

- A chave BDK (como todas as outras chaves AES) deve ser protegida por uma LMK AES.

- A chave BDK sempre será uma chave AES de 128/192/256 bits de tamanho.

· A chave Initial Key (IKEY) - derivada da BDK - sempre será uma chave AES de 128/192/256 bits.

- A chave Update Key (UK) - derivada do AES BDK - sempre será uma chave AES de 128/192/256 bits de tamanho.

- As chaves de trabalho derivadas (PIN, Dados, chaves MAC) sempre serão chaves do tipo AES e possuem o mesmo tamanho da AES BDK).

- Observe que o padrão X9.24-3 também permite a derivação de chaves 3DES e HMAC, mas isso não é suportado no momento.

- Todos os blocos de PIN protegidos por uma chave PIN AES usarão o formato ISO 4 (formato HSM 48)

- O tamanho do KSN é de 24 bytes.

## Variant DUKPT

**Encriptação de dados utilizando uma chave do tipo PIN Encryption**

O método tradicional de implementação do DUKPT X9.24-1:2004 é consiste em permitir que o terminal seja carregado inicialmente com um IKEY e daí em diante seja deriva uma nova chave de criptografia de PIN (e, opcionalmente, uma nova chave de MAC) para cada transação realizada. Como esse método não suporta a derivação de chaves de criptografia de dados, alguns fornecedores de terminais implementam um método híbrido, utilizando duas chaves do tipo BDKs: uma delas é usada para derivar as chaves de criptografia PIN e MAC e uma segunda chave BDK é utilizada para derivar as chaves de criptografia de dados.

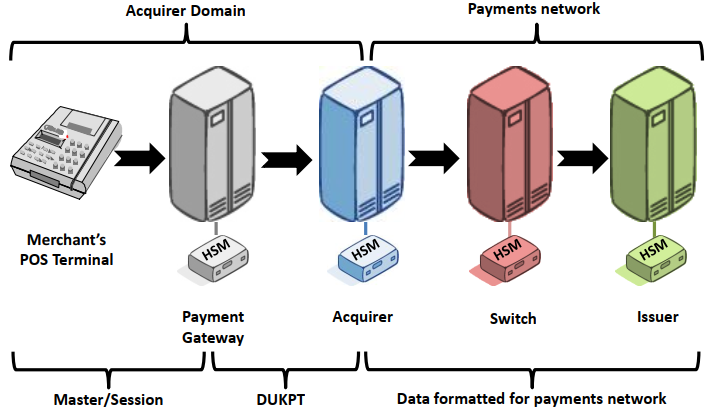
Nesse caso, o terminal é carregado inicialmente com IKEYa e IKEYb (que foram derivados de BDKa e BDKb, respectivamente). Durante cada transação, o terminal deriva duas chaves, TransactionKeya e TransactionKeyb. A chave de criptografia de PIN (e opcionalmente a chave MAC) é derivada do TransactionKeya usando métodos compatíveis com X9.24-1:2004. A chave de criptografia de dados é derivada do TransactionKeyb usando o método de derivação da chave de criptografia de PIN descrito em X9.24-1:2004.

BDKa é um tipo de BDK 1 e BDKb é um tipo de BDK.

**Tradução de dados para DUKPT**

No diagrama acima, o fluxo de dados é do terminal POS para o Adquirente e depois para a rede de pagamentos padrão. Como já foi mencionado, isso envolverá a tradução de dados DUKPT para os formatos usados pela rede de pagamentos.

No entanto, considere o seguinte ambiente:



No esquema acima, nós temos um elemento novo no esquema tradicional, o Payment Gateway. Este é um intermediário entre o comerciante e o adquirente. O Payment Gateway fornece vários serviços ao Comerciante e pode ser necessário porque o comerciante é muito pequeno para ter um relacionamento direto com um Adquirente. Para a presente discussão, assumiremos que o Payment Gateway é necessário porque o Adquirente fornece suporte apenas para terminais DUKPT, mas o comerciante não suporta DUKPT.

Nesse ambiente, o Payment Gateway implementará o gerenciamento de chaves master/sessão com os terminais POS do comerciante, usando os mesmos dados e formatos de bloqueio de PIN que a rede de pagamentos. No entanto, como o Adquirente suporta apenas o gerenciamento de chaves DUKPT e os formatos de dados no lado do terminal, o Payment Gateway deve ter a capacidade de converter dados para os formatos DUKPT.

## DUKPT para Point-to-Point Encryption e Mobile Acceptance

**Point-to-Point Encryption (P2PE)**

Tradicionalmente, no processamento de pagamento com cartão, apenas o PIN é criptografado (na forma de um PIN Block) - outros dados, como o PAN, geralmente não são criptografados: o PCI DSS cobre o domínio do adquirente e requer que os dados do titular do cartão, como o PAN, quando os dados sejam criptografados quando passam por uma rede pública. Já em uma rede privada segura, as informações não precisam ser criptografadas.

A criptografia dos dados do titular do cartão capturados tanto do lado do comerciante quanto do lado do Adquirente e na rede que os conecta - seja uma rede privada ou não, não é coberto pela conformidade e pela auditoria do PCI DSS. Isso é relevante para os comerciantes e adquirentes, devido ao custo da conformidade e auditorias do PCI DSS, e, portanto, há um interesse crescente em criptografia dos dados do titular do cartão no domínio do adquirente - isso é chamado de P2PE.

Existe uma grande variedade de implementações de P2PE e está disponível, com projetos proprietários de seus fornecedores. Essas implementações podem envolver um Payment Gateway.

O DUKPT é frequentemente usado como o método de gerenciamento de chaves para uma solução P2PE, pois fornece uma maneira padronizada de troca de dados criptografados (não apenas PINs) e um altos níveis de segurança.

**Mobile Acceptance ou Mobile Point of Sale - mPOS**

Os terminais tradicionais de pagamento com cartão, POS, estão fora do alcance de pequenos comerciantes ou comerciantes móveis devido ao custo dos terminais e à infraestrutura de comunicações de linha fixa necessária para apoiá-los. A aceitação móvel é uma tecnologia que soluciona esse problema e desfruta de uma rápida taxa de adoção.

Com a tecnologia Mobile Acceptance, o terminal POS consiste em um leitor de cartão seguros e um dispositivo de entrada de PIN conectados ao dispositivo de comunicação móveis inteligente. Isso coloca a tecnologia ao alcance financeiro de qualquer comerciante ou provedor de serviços e elimina a dependência de uma infraestrutura de comunicações fixas.

O P2PE deve ser empregado entre o terminal móvel e o Adquirente ou um Payment Gateway que forneça um serviço de transações móveis aos comerciantes e repasse as transações a um Adquirente.

Novamente, o DUKPT é frequentemente empregado como o esquema de gerenciamento de chaves nesse ambiente.

## payShield 9000 na DUKPT

O payShield 9000 fornece uma plataforma segura para a execução de funções criptográficas, protegendo os dados secretos contra invasores, dentro ou fora da organização. Pode realizar tarefas como:

- Gerando chaves;

- Protegendo chaves criptografando com uma LMK, tecnologia KEK;

- Criptografar/descriptografar dados;

- Importar/exportar chaves;

- Converter o valor do PINs que foi criptografado por uma chave de derivação básica para uma chave de intercâmbio apropriada e que foi compartilhada entre o adquirente e o emissor ou comutador;

- Verificar os valores dos PINs recebidos de um terminal usando as chaves de derivação básica. Métodos de verificação utilizados pelo HSM (IBM, Diebold, Visa PVV e PIN criptografado) são suportados.

- Cálculo e verificação de vários tipos de valores de verificação.

- Verificação do valor do MACs nas mensagens recebidas de um terminal POS e gerar valores MACs nas mensagens a serem enviadas para um terminal POS.

Os esquemas de cartões exigem que as organizações envolvidas no processo (Adquirentes, Comutadores e Emissores) e que estão processando as transações de seus cartões usem HSMs, como o payShield 9000, para determinadas funções criptográficas. Mesmo para funções não cobertas por esses mandatos, o uso de HSMs é uma prática recomendada de segurança para essas organizações.

Os sistemas e produtos usados ​​por essas organizações devem obedecer aos padrões desenvolvidos pela PCI. Os sistemas dos adquirentes são auditados de acordo com o padrão PCI DSS. Os produtos de pagamento devem obedecer a vários padrões e certificados por vários padrões PCI. Por exemplo, os terminais de pagamento, incluindo dispositivos compatíveis com o processo DUKPT, devem ser certificados de acordo com o padrão PCI PED (Dispositivo de entrada de PIN). No caso dos HSMs, o PCI introduziu o padrão PCI HSM recentemente. Embora esse padrão ainda não seja obrigatório pelos esquemas de cartões, o PCI em seu padrão para P2PE especifica que os HSMs usados ​​pelo adquirente são certificados como FIPS 140-2 ou PCI HSM

# Release Notes

|  |  |  |
| --- | --- | --- |
| **Autor** | **Descrição** | **Data** |
| Caio Ferreira | Primeira versão | 02/04/2020 |
|  |  |  |