FUJ00098154

A&TC
Enterprise
Solutions

ICL Pathway Horizon Project

Key Management High Level Design

Author: Rob Arthan
(see also section 0.7)

Reference: RS/DES/010

Issue: 3.0

Date: 10 March, 1999

Comments by

Abstract: This document is the top of the design documentation tree for the Pathway Key Management System for NR2+.

Approver: Dave Johns

Signature & Date:

This is a controlled document. This issue is definitive if it is the latest which has gained the approver's signature. Check with the document controller (below) that this is the latest issue. **An out-of-date issue or a non-approved issue is not definitive**.
This issue (3.0) is an internal draft.

Controlled by: Pauline Grice Location: BRA01
Phone: GRO
Electronic repository: Visual Source Safe \\nt025\Agent_dev\vss
$/Pathway/Cryptography/Documents/ Key Mgt Service (R2+ - KMS) /HLDUpdate/DES010 KMHLD.doc

Distribution: (Internal drafts are distributed to the starred names only.)

BRA01	Belinda Fairthorne	MAN27	Simon Fawkes
FEL01	Alan D'Alvarez*	BRA01	Chris Sundt
FEL01	Alan Ward	BRA01	Gareth Jenkins
FEL01	Anne Cooper	BRA01	Glyn Thomas*
FEL01	Barry Procter	BRA01	Pathway KMS Team*
FEL01	Bill Curtis	BRA01	Pete Drewett
FEL01	Chris Wannell	BRA01	John Lyon
FEL01	Gill Jackson		
FEL01	Glen Stephens		
FEL01	Harvey Potts		
FEL01	Jerry Boyce		
FEL01	Mark Fisk		
FEL01	Mark Jarosz		
FEL01	Pathway Library		
FEL01	Peter Jeram		
FEL01	Peter Wiles		
FEL01	Richard Long		
FEL01	Stephen Doyle*		
FEL01	Tom Parker*		

FUJ00098154

0. DOCUMENT CONTROL

0.1 Document history

Issue	Date	Reason
0.1		Informal and very sketchy draft, distributed to gather early comments.
0.2	10/3/98	Informal complete draft for more detailed comment. Major changes: (i) details of key client processes added; (ii) "Migration" section completely rewritten following discussions.
0.3	17/5/98	Total rewrite. Comments received on issue 0.2 led to the conclusion that the design was largely sound but ambiguous in some areas and not well organised. In particular: (a) there was confusion between the management of principal keys (i.e. the signature and data encryption keys) and the key protection mechanisms (red keys, black key sets); (b) there was repetition but also inconsistency in the text, which did not clearly identify the opportunities for commonality in implementation; (c) there was ambiguity in the concept of "key domain" which became clear when trying to specify the data relationships for the KMA; (d) there was ambiguity in the concept of "distribution channel". To address these problems it was necessary to revise the "System Design" diagram. Since that is the core element of the document, extensive text revision was unavoidable. For guidance, here is a synopsis of the changes with respect to issue 0.2. 1. "System Design" diagram completely revised. a) Extended to emphasise key protection as a separate concern and accommodate different protection schemes (to unscramble the "red key"/"black key" discussion from the fundamental business of managing the data keys). b) Different key flows distinguished from one another. c) Distribution routes and key clients redrawn to represent key flows more accurately. 2. In line with the above, the text of the "System Design" section now discusses the design principles and techniques to be used in the Key Management system, without the obscuring details which are a consequence of technology choice. Those details are moved to a later section, "Detailed Design Units". This decoupling will allow for changes in choice of technology (e.g. VPN vs. CHAP) without impact on the design framework. 3. Distribution channels are explained in greater detail, both in the System Design and the Detailed Design sections. 4. The Detailed Design section identifies common process units.

FUJ00098154

5. Physical architecture of the Key Management Centre will be dealt with in a separate document (possibly the KMA design), rather than a future issue of this document.
6. The term "key domain" is replaced by the term "application domain" because it better describes the space being labelled, and does not lead to the conceptual error that only one key relationship exists in a domain (the AP application domain contains up to 20,000 public key relationships).
7. The "Key distribution matrix" which was in the "Key Domains" section of 0.2 is separated into several key routing tables and moved to the "Key Management Application" subsection of the Detailed Design section.
8. Most of the architectural appendix is moved to a new early section: "System Context". The remainder of that appendix was redundant paraphrasing of text that was already in the main body of the document, and so has been discarded.
9. The System Context section includes a discussion of revocation and latency.
10. The "Workpackages" section has been removed because the "Detailed Design Units" section better serves the purpose.

0.31 18/5/98 Minor updates to sections 2.1, 4.4 (causing some renumbering) and 4.6

(Alex Robinson)

0.4 08/07/98 Substantial revisions following inspection.

Again, the document has been re-structured to separate the general design of the Pathway Key Management system from the specifics of the keys that are currently envisaged for R2+.

Many readers of the HLD are looking for details of specific keys. Yet the design must look beyond those specifics for two reasons:

(i) efficient implementation comes from designing around common factors, not specifics;

(ii) future-proofing is not achieved by concentrating on today's specifics.

Section 1 includes a summary profile of the specifics (keys, domains, etc.) for R2+.

Sections 2, 3 and 5 remain general, and some of the R2+ detail has been removed from §3. References to specific R2+ items are for illustration only. Section 5.6 "Potential for Change" is important because it describes how the general design of the system will accommodate extensions beyond the currently envisaged R2+.

Section 4 contains a detailed profile of R2+ specifics.

Numerous details have been added, such as the contents of a public key certificate, at the request of the inspectors.

Numerous other details have been deferred as "Design Issues", to be resolved in later issues.

Terminology has again been adjusted.

"Protection domain" replaces the term "Application domain", because some cryptographic protections apply to data links which cross the

FUJ00098154

		boundary between things which might be considered separate applications. "Campus" replaces "Data Centre" in the names of channels. The term "directed" is introduced to describe channels which can target specific recipients, as opposed to "broadcast" channels.
1.11	28/8/98	Revisions after comment from Pathway and further work on design approach amounting to a concentrated attack on section 3 and the early parts of section 4 and addition of sections 2.7 and 2.8. The section on the scope of this document now emphasises the role of "Requirements for Key Management". A brief description of the cryptosystems being used has been added. Distribution channels now modelled around Riposte. System design and component breakdown made more explicit.
1.12	3/9/98	The planning process highlighted the following errors and omissions which have now been addressed. The component summary in section 3.12 did not reflect the component breakdown of section 3.8. Section 3.8 (and so section 3.12) did not give a component breakdown. The diagrams of section 4 have been revised to bring them into line with the TED and to correct various minor errors. Section 4 now includes a list of keys. The beginnings of the descriptions of interfaces and protocols in section 3 have been added.
1.13	15/9/98	Addressed comments received up to and including section 4.4 Accommodated recent design decisions (in particular those made at the KM Agent workshop) Added more detail in section 3
1.21	5/10/98	Addressed residual comments from Tom Parker (ref PA/TEM/0020) System diagrams of section 3 redrawn and corrected in the light of comments and workshop discussions. Sections 1 to 4 inclusive have been updated in the light of comments and ongoing design work. A start has been made on section 5.
1.3	26/10/98	Responses to comments passim. Corrections to protection domain delivery diagrams. Platform definition diagram added. More diagrams redrawn in light of detailed design work Structure diagrams for generic client and KMA added. Overview of VPN technology added. Registration of nodes in PO outlets and PO synchronisation treated at greater length including state transition diagrams for synchronisation protocol

FUJ00098154

		State transition diagrams for all the other protocols added.
		Document references are now by mnemonic rather than number.
		Issues section renamed "Risks and Assumptions" and brought in line with Crypto Team standards.
		Restructure section 3.2 to reflect current position on key packaging.
		High-level picture of PMMC agent simplified.
1.4	11/11/98	Changes in response to inspection of 3/11/98. All points listed in the Quality Review notes have been addressed as have many other comments submitted to the inspection.
2.0	16/11/98	Changes in response to comments on version 1.4 and residual comments on version 1.3.
		Circumvented bugs that made V1.4 hang MS word.
2.1	8/3/98	Actioned changes from [KMHLDUPD].
		Cosmetic changes (layout of cross references section, typos).
		Added acknowledgments section and truncated list of authors to avoid confusion.

3

FUJ00098154

0.2 Changes Forecast

This is the second approved baseline of this document and accompanies the first baseline of the detailed design documentation which underpins the development work. In parallel with the development work, the design process will enter a consolidation phase, in which outstanding issues and any problems thrown up during development will be addressed. A list of errata and addenda will be prepared to record subsequent design decisions and, under Pathway change control processes, these will be incorporated in the next issue of this document. The following sources of likely change can be identified at the time of writing:

- There is a requirement to recover a PO outlet from broken or lost PMMC or PIN when the comms are not available. A proposal for this has been made but is conditional on security requirements for the L&G key material which have not yet been finalised. This proposal or some suitable alternative will be added to this document when the details are known.
- A requirements trace section will be prepared when a version of [KMREQ] is baselined including requirements tags.
- It is intended to move the material on standards in this document into [POKM] at some future date.
- During the design phase, studies have been carried out on Threat Analysis, Performance and Resilience. The results of the performance study have been included in this document as have some of the results of the resilience study. The report on the Threat Analysis is still being assessed in detail and there may be minor changes to the design as a result.
- The impact on KMS of the ECCO migration process will be assessed.
- The impact, if any, of LFS on KMS will be assessed.
- The assumptions in section 10.1 of this document will be validated against other design efforts, and any necessary accomodations to the KMS design will be made.

0.3 Cross References

	Title.	Reference	Version (date)	Source/author
[ACP]	Access Control Policy.	RS/POL/003	2.0 (24/02/98).	Belinda Fairthorne
[CAPSINTSPEC]	Interface Specification for CAPS Link Crypto Services.	PWY/SEC/D/10	2.0 25/07/97	TSC Crypto Team library
[CRYPARCH]	Cryptographic Architecture.		0.1 31/10/97	Tom Parker
[INTUTIMACO]	Integrating Utimaco Code and Crypto Code..	SD/DES/082	TBA	Alex Robinson

FUJ00098154

[ISO11770-1]	Information technology – Security techniques – Key management – Part 1: Framework.	ISO/IEC 11770-1:1996		ISO
[KEYGENDES]	KM Key Generation Detailed Design.	TSC/CRY/044	0.6 29/1/99	Tony Dolton
[KMACDES]	KM Automatic Channel Detailed Design.	TSC/CRY/060 RS/DES/033	0.5 29/1/99	Mike Garrett
[KMAPDES]	KM Application Detailed Design.	RS/DES/018	0.8 29/1/99	Alex Robinson
[KMCAGDES]	KM Client Agent Detailed Design.	TSC/CRY/061 RS/DES/035	0.5 29/1/99	Peter Haydon
[KMCAWDES]	KM Certification Authority Detailed Design.	TSC/CRY/042 RS/DES/029	0.6 29/1/99	Adrian Barclay
[KMHLDUPD]	KM HLD Update Proposals.	TSC/CRY/070 RS/DES/026	0.5 29/01/99	Rob Arthan
[KMICDES]	KM Interactive Channel Detailed Design	TSC/CRY/058 RS/DES/032	0.4 29/1/99	Tony Dolton
[KMMCDES]	KM Manual Channel Detailed Design	TSC/CRY/065 RS/DES/031	0.4 29/1/99	Andy Williams
[KMMIG]	Key Management Migration (NR2 to NR2+)	TSC/CRY/068 RS/DES/025	0.4 29/1/99	Peter Robinson
[KMPLATFORMS]	Key Management Platforms.	RS/DES/20	0.5 29/1/99	Andy Williams
[KMREQ]	Requirements for Key Management.	RS/REQ/009	Issue 1.0 [1] 8/5/98 Issue 5[2] 22/2/99	Tom Parker
[KMTERM]	Key Management Terminology.	TSC/CRY/057	TBA	Peter Haydon
[LANDGCRYPTO]	Cryptographic Support for L&G Smart Token Detailed Design.	TSC/CRY/049	0.7 29/1/99	Graham Rogers
[LOGREQ]	Logging Requirements for Crypto Code.	RS/REQ/007	TBA	Rob Arthan
[PMMCADES]	KM PMMC Agent Detailed Design.	TSC/CRY/059 RS/DES/036	0.7 29/1/99	Keith Simons and Richard Glanville
[POKM]	Post Office Key Management High Level Design.	RS/DES/021	4	Tom Parker

[1] Latest Baseline version

[2] Latest available draft at time of issue

FUJ00098154

[SCHNEIER]	Applied Cryptography. Bruce Schneier.	ISBN 0-471-11709-9	2nd edition.	John Wiley & Sons Inc.
[SFS]	Security Functional Specification..	RS/FSP/0001	3.0.	Pathway library
[SMH]	(Secure Material Handling)			
[TED]	Technical Environment Description.	TD/ARC/0001	4.0 16/6/98	Alan Ward/Peter Wiles

0.4 Abbreviations

See also [KMTERM].

AP	Automated Payment
API	Application Programmer's Interface
ASN.1	Abstract Syntax Notation One
BES	Benefit Encashment System
BKS	Black Key Set
BPS	Benefit Payment Service
CA	Certification Authority
CAPS	Benefits Agency Customer Accounting and Payments Strategy
CAPU	CA Public Key
CAS	CAPS Access Service
CAW	CA Workstation
CESG	Communications - Electronics Security Group
CHAP	Challenge Handshake Authentication Protocol
CM	Configuration Management
CMS	Card Management System
Comscire	Third party provider of Random Number Generator hardware
CRL	Certificate Revocation List
DEK	Data Encryption Key
DLL	Dynamic Link Library
DSA	Digital Signature Algorithm
Dynix	Proprietary Unix operating system
ECB	Electronic Code Book
EDS	Company name: the managed service provider operating the CAPS system for the Benefits Agency
Escher	Provider of Riposte software
FAD	Financial Accounts Division (one way of identifying a Post Office, see also OUC)
FEK	Filestore Encryption Key
FTMS	File Transfer Management System
GUI	Graphical User Interface
ISDN	Integrated Services Digital Network
ISO	International Organisation for Standardisation
KEK	Key Encryption Key

FUJ00098154

KES	Key Encryption Seed
KM	Key Management
KMA	Key Management Application
KMC	Key Management Controller
KMS	Key Management System
L&G	Landis & Gyr
LAN	Local Area Network
Layer 7	Cryptography library provided by Sapher Servers
LFS	Logistics Feeder System
OUC	Organisational Unit Code (another way of identifying a Post Office, see also FAD)
NR2	Pathway New Release 2
NR2+	Pathway New Release 2+
NT	New Technology: the Microsoft operating system widely used in Pathway
PA	Payment Authorisation
PAPR	PA Private Key
PIN	Personal Identity Number
PKC	Public Key Certificate
PMMC	Post Master's Memory Card
PO	Post Office
POCL	Post Office Counters Ltd
PoLo	Post Office Logon
POM	Post Office Manager
SMC	Systems Management Centre
R1c	Pathway Release 1c
Rambutan	A symmetric encryption algorithm implemented in Zergo communication hardware.
RD	POCL Reference Data
Red Pike	A symmetric encryption algorithm
Riposte	A resilient messaging system
Sequent	Hardware box running Dynix
SI	Software Issue
SIPR	SI Private Key
TBKMA	Thames Bridge Key Management Algorithm
TCP/IP	Transmission Control Protocol / Internet Protocol
TIP	Transaction Information Processing
Tivoli	A distributed system management system
VME	Virtual Machine Environment
VPN	Virtual Private Network
X.509	collective heading for a number of standards and draft standards which define an infrastructure for managing the public components of asymmetric (private/public) key pairs

32

FUJ00098154

0.5 Contents

FUJ00098154

FUJ00098154

0.6 Figures

FUJ00098154

154 0.7 Acknowledgments

155 Versions 0.1 to 0.4 of this document were written by Charles Lambert.
156 The key distribution diagrams of section 4.1 were originally prepared and maintained by Alex Robinson.
157 James Stinchcombe provided much of the new material added in section 5.1 at version 3.0.

FUJ00098154

1. INTRODUCTION

1.1 Scope

This design responds to "Requirements for Key Management" [KMREQ], which in its turn responds to the Pathway documents "Cryptographic Architecture" [CRYPARCH], "Security Functional Specification" [SFS]. The Pathway document "Access Control Policy" [ACP] provides additional context and requirements for this document as well as for "Requirements for Key Management" [KMREQ]. Where relevant, UK & International standards have been consulted for design guidelines; however, the above-mentioned Pathway documents provide the definitive statement of the requirements that this document addresses.

"Post Office Key Management" [POKM] provides a useful alternative view and additional detail on important aspects of the KM service.

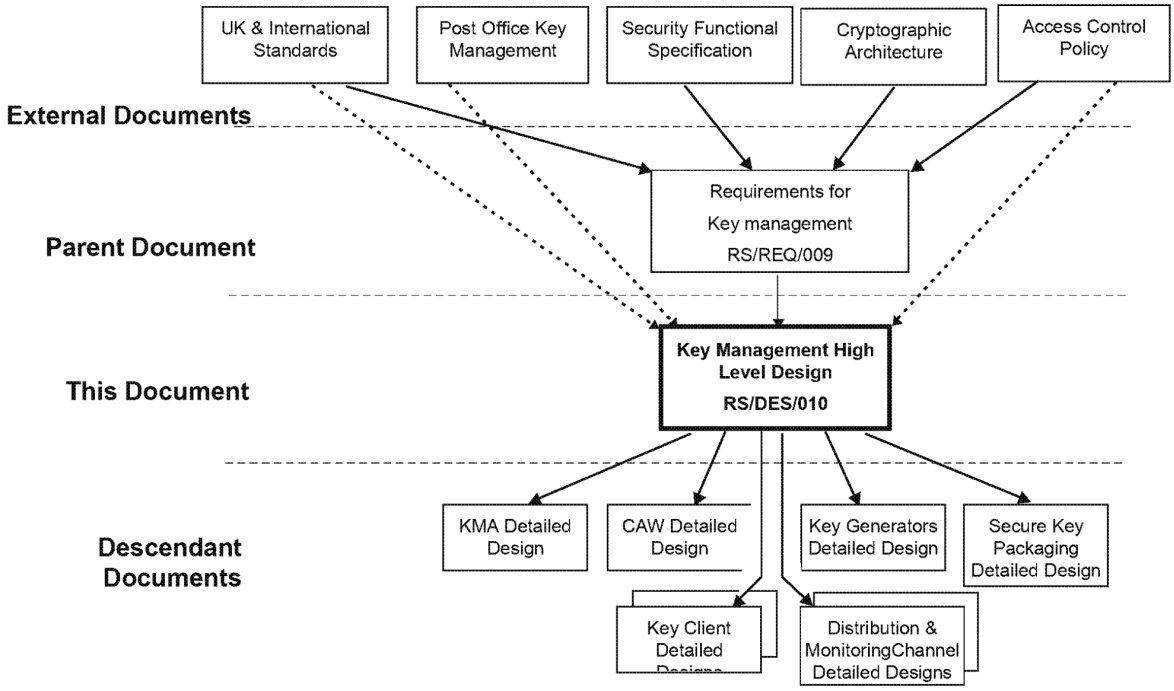


Figure 1. Document relationships

This design addresses the management of cryptographic keys throughout the Pathway system at Release 2+ and beyond. It is a high level design: as such it describes the processing structure of key management as a whole, specifying the principal system modules and the interactions between them.

The detailed design of the modules will be described in documents which descend from this design, as shown above.

FUJ00098154

Recognising the need for the Pathway Horizon system to migrate from the existing (Release 2) key management procedures to the systems and procedures described here, a later section of this document discusses migration.

It is not assumed that the NR2+ software defined in this design will be deployed across the entire Pathway estate in a single exercise. For business or operational reasons a phased deployment may well provide benefits. Rather than predict the business decisions, the main body of the KM design describes the eventual steady state. The design documents include observations about the migration considerations for the individual components; however, it is for the business to dictate how the KM system is to be introduced and, in particular, to define which protection domains are to be supported at each phase of deployment.

1.2 Background

1.2.1 Cryptography in Pathway

The Security Functional Specification [SFS] identifies a number of uses for cryptography in securing the Pathway business services. Subsequent agreements have identified further requirements for cryptography to protect third-party software. With one exception, the complete list of cryptographic protections at the time of writing (with abbreviations) is

- Benefit Encashment System (BES) Payment Authorisations (PA)
- Software Issue (SI)
- Client services Automated Payment service (AP)
- post office data Filestore Encryption Key (FEK)
- Benefits Agency Customer Accounting and Payments Strategy (CAPS)
- benefit claimant Card Management System (CMS)
- Post Office Counters Ltd (POCL) Transaction Information Processing (TIP)
- POCL Reference Data (RD)
- Automated Payment service bulk Client transaction records (AP Client)
- Landis & Gyr 3rd party code and data protection (L&G Code)
- Landis & Gyr transaction-enabling functions (L&G Enabling)
- Utimaco Virtual Private Network (VPN)
- Rambutan encryption of data links (Rambutan)

The item excluded from the above list is Escher Riposte application software authentication. Keys for this cryptographic function will not be managed within the Pathway run-time system and so are excluded from the scope of this document.

FUJ00098154

1.2.2 Organising key management

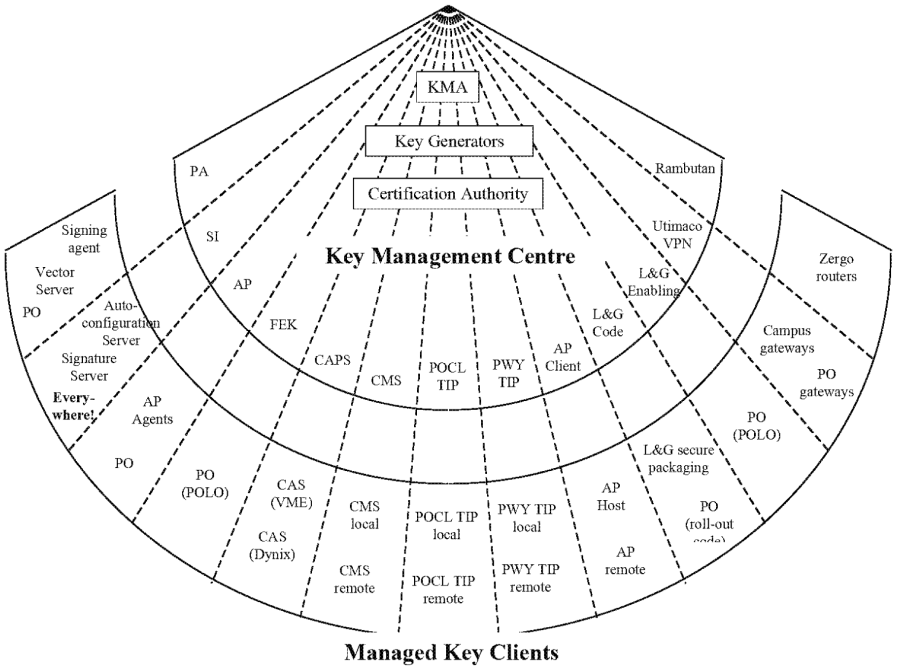


Figure 2. Key management "fan diagram"

All the cryptographic functions in the above list require keys. These keys must be securely created, distributed and installed in the cryptographic functions, and each key must be changed periodically. Hence, there are a number of common key management activities to be performed across a diverse spectrum of keys. All of this activity is to be managed by a single officer of Pathway, defined as the Cryptographic Key Manager [ACP].

To help to visualise this problem space, and to begin to organise it, the "fan diagram" of Figure 2 was evolved. It represents key management emanating from a single point of control and fanning out along segments which correspond to the various uses of cryptography (as listed above) to the many points at which the keys are used. Note that the TIP and RD cryptographic applications are considered under the protection domains POCL TIP and PWY TIP, one corresponding to authentication of POCL to Pathway and the other corresponding to authentication of Pathway to POCL.

Some key management actions will be manual. Representation in the fan diagram does not necessarily imply automation. For example, Rambutan keys, which are supplied by an external agency and installed in special hardware, will be managed entirely by manual procedures. However, the Key Management system will provide the Key Manager with facilities to record and track manual procedures.

The diagram identifies several functional blocks, such as "Key Generators" and a "Certification Authority" which would form a central facility to support the Key Manager. These functions will be explained later in this document. However, the "KMA" merits particular note here.

1.2.3 Some Important Terms and Concepts

The Key Management Controller (KMC) is the software providing the control centre for the key management system. It comprises the Key Management Application (KMA), which is in fact a suite of

FUJ00098154

programs built around a management information database, together with supporting software and hardware for key generation and certification. The database contains a model of the rest of the system and all the managed objects (keys, clients, etc.) within it. The KMA uses this model to give the Key Manager a view of the system status, and to assist the Key Manager in performing management actions, guarding the integrity and coherence of the system as a whole.

A Key Management client comprises a platform and associated software requiring the services of the Key Management Controller. The client population is numerically dominated by the PCs on PO counters but there are many other client types (see the diagrams of section 4.1). On many types of client, a Key Management Client Agent is installed; this is the software primarily responsible for mediating between the Key Management system and the cryptographic support software running on the client during normal operation.

The KMC and its clients communicate by means of distribution and monitoring channels. There are several types of channel depending on the transport mechanisms that are appropriate for a given purpose (see the diagrams at the beginning of section 3).

1.3 Document structure

The remaining sections of this document are organised as follows:

Section 2 lays the groundwork for the design of the key management system. It introduces an entity relationship model for the management of keys, discusses key protection and the need for additional keys to achieve this, surveys the industry standards which apply to key management, and outlines the process model for Pathway key management.

Section 3 is the system design for Pathway key management. It defines the structure of the management system in terms of processes, data structures and data flows. This is the general design, intended to be applicable at Release 2+ and also beyond. This section is independent of R2+ specifics.

Section 4 defines what will be implemented for Release 2+, conformant with the framework of Section 3.

Section 5 defines the non-functional design qualities of the system, e.g. security, performance, etc.

Section 6 is a brief discussion of the way in which Release 2 platforms will migrate to Release 2+ functionality.

Section 7 discusses systems management of the main components of the KM system.

Section 8 identifies testing requirements.

Section 9 lists major dependencies on other developments and systems.

Section 10 lists risks and assumptions.

FUJ00098154

2. DESIGN PRINCIPLES

2.1 Key management entities

The known list of cryptographic functions (see 1.2), for which keys are to be managed, is diverse. As the Pathway system develops commercially, and more third-party client services are added, the diversity can be expected to increase. In order to design a key management system which is efficient to implement and also flexible enough for the future, it is necessary to organise this diversity of keys in some way which

(a) identifies common characteristics for common processing, and

(b) gives the Key Manager a manageable and comprehensible view of the material under his (or her) control.

The entity relationship diagram (Figure 3) will form the basis of this organisation.

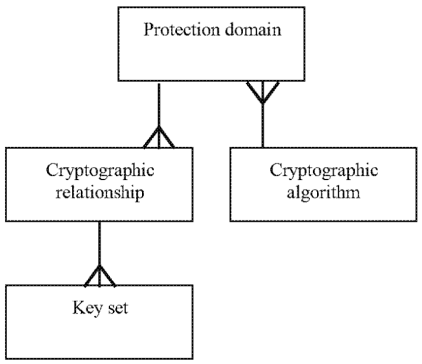


Figure 3. Entity relationships in key management

2.1.1 Protection domain

The Key Manager will view the task of key management from an understanding of the Pathway technical environment and the business functions it supports. He will therefore think in terms of keys for "AP", "PA", "CMS", etc. Each of these divisions is a "Protection domain".

2.1.2 Cryptographic algorithm

Within one protection domain, the cryptographic functions implement a particular "Cryptographic algorithm" (e.g. DSA, Red Pike). Conversely, one algorithm may be employed in several domains (both PA and AP employ DSA).

2.1.3 Cryptographic relationship

Within one protection domain there may be many separate "Cryptographic relationships". For example, every individual post office is accountable for the AP transactions which it conducts. Therefore, the AP transaction harvester must be able to distinguish the digital signature of one post office from another. That is to say that each post office has a separate relationship with the harvester within the AP protection domain.

FUJ00098154

2.1.4 Key set

A cryptographic relationship is distinguished by the fact that the participants share a unique "Key set". At first sight, one might therefore expect a 1:1 relationship between "Cryptographic Relationship" and "Key Set". However, the entity relationship diagram takes into account the fact that the keys in a particular relationship will be changed at routine intervals, or in case of compromise. So over time one cryptographic relationship will use a series of key sets. A cryptographic relationship may also use more than one key set at the same time.

2.2 Key protection

The primary purpose of the Key Management system is to manage the keys required by the Pathway business systems. In handling those keys, the management system must protect them against corruption (an attacker might attempt to compromise Pathway security by perverting Pathway key material for his or her own ends). Confidential keys, that is to say symmetric encryption keys and private signing keys must also be protected against malicious or accidental disclosure to unauthorised parties.

Public keys are protected against corruption by digital signatures. Confidential keys are protected against disclosure by encryption under another key (a key encryption key or KEK). For simplicity, this design generally uses just one KEK for each client that holds confidential keys; this KEK is generally held on a removable token (a memory card or diskette). For historical reasons, the per-client KEK is referred to as TK (traffic key).

Hence the Key Management system, in order to manage the primary keys, will introduce and employ keys of its own. The system will manage these keys according to the same entity relation model as described in section 2.1. That is to say, the design of key management will define "KM protection domains" with associated algorithms, relationships and key sets. These KM protection domains are identified in detail in section 4.1.3.

The most prominent key protection domain is the "CA" domain, in which a Certification Authority will sign public keys to protect their integrity and all users of the signed keys will verify the signature. This is further explained in section 2.3.2.

2.3 Applicable standards

References to standards in this document do not imply commitment to implementing those standards. In particular sections 2.3, 2.4, 2.6.1 and 2.6.2 simply provide informative background material. These sections will be moved out of this document into [POKM] in a future issue.

2.3.1 ISO Key Management Framework

Draft standard ISO/IEC DIS 11770-1 [ISO11770-1] defines the stages in the life of a key and the transitions between them.

FUJ00098154

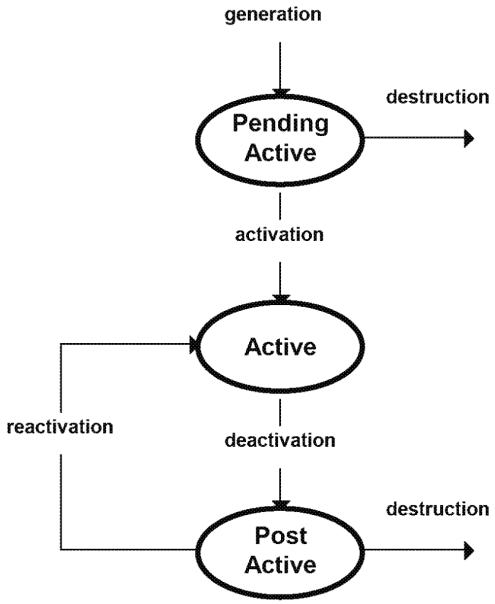


Figure 4. ISO Key Life Cycle

Within each transition (generation, activation, etc.) the standard identifies several "services" - i.e. processes - such as "generate-key", "create-key-certificate", "revoke-key". It also says

> "other life cycle models may have additional details that may be sub-states of the three states presented."

This design defines two sub-states of the "Active" state:

"Loaded", meaning that a copy of the active key is available to executing cryptographic processes in processor memory;

"Not Loaded", meaning the opposite of the above.

To manage the transition between these two sub-states, the design defines two services (key processes): "load-key", and "unload-key". These sub-states and services are illustrated in a later subsection.

2.3.2 X.509

There are a number of standards and draft standards under the collective heading "X.509" which define an infrastructure for managing the public components of asymmetric (private/public) key pairs. Although ISO 11770-1 embraces asymmetric keys, it does not address the considerable difficulties of making the public components widely available whilst assuring their integrity, currency and attribution. The X.509 standards concentrate on this subject.

Most usefully, X.509 defines a data structure called a "public key certificate" (PKC), which carries a public key together with such management information as the identity owner of the corresponding private key. The certificate data is digitally signed by an authority – the Certification Authority (CA) – which underwrites the information to some extent.

The standards also define a structure called the "certificate revocation list" (CRL). This is also signed by the CA and carries information about certificates in circulation which are no longer to be trusted.

FUJ00098154

2.4 Implementation of standards

2.4.1 ISO 11770-1 key processes

Where appropriate, the Key Management System will implement a complete key life-cycle according to the ISO 11770 framework (section 2.2). The draft standard identifies mandatory and optional "services" (key processes) in each transitional phase. The key management system will follow the ISO 11770 processes as defined in the following diagram wherever that is deemed to be appropriate and cost-effective for ICL and its customers and collaborators.

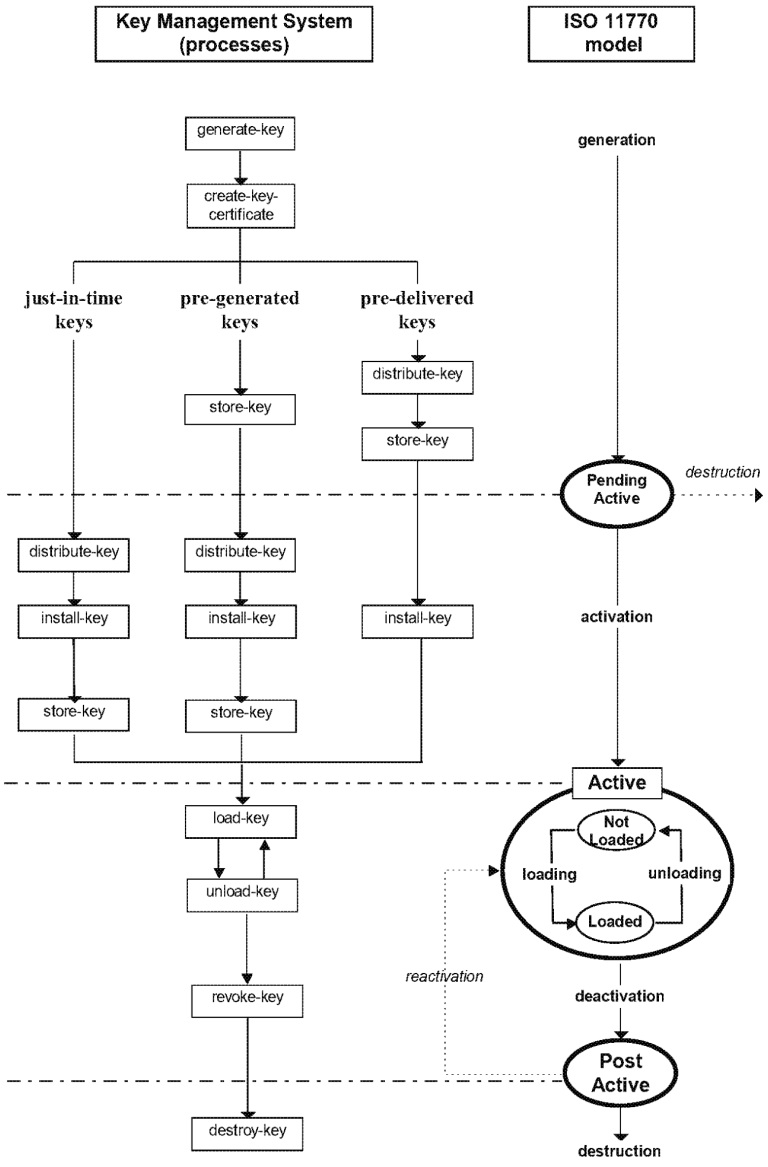


Figure 5. Pathway profile of the ISO model

FUJ00098154

Figure 5 is a chronological diagram, not a topological one. There are several instances of the transition process "store-key"; the diagram indicates only the instant in the key life-cycle at which they occur, not their physical location in the system. Hence, for example, the "store-key" process in the "pre-delivered keys" path will be located at the key client, not in the key management centre, despite its occurrence early in the life-cycle; whereas the "store-key" process in the "pre-generated keys" path will be located at the management centre.

The ISO 11770 model does not reflect the rather important differences in the lifecycles of a key according to its role in a cryptographic algorithm: signing and encryption keys, decryption keys and public keys are not all handled in the same way.

Figure 5 shows three parallel life-cycles for "just-in-time keys", "pre-generated keys" and "pre-delivered keys". These distinctions are explained as follows.

2.4.1.1 Just-in-time keys

A **just-in-time key** is one that is generated immediately prior to activation. For example, new CMS encryption keys will be generated only when a key change is due, and delivered into the active state as soon as possible. (Note the anomaly that, since the CMS service has begun operation before the introduction of this Key Management Service, a CMS key will already be active at introduction. This will be treated as a "pre-delivered key"; see below.)

2.4.1.2 Pre-generated keys

A **pre-generated key** is one that is generated well in advance of the need to use it. The key is held in the "Pending Active" state at a central location. It will not be distributed to the point(s) of use until it is due to be installed in the "Active" state. This technique is used for the Certification Authority private key only (CAPR). A stock of these keys is generated prior to first operation of the KM system and the corresponding public keys (CAPU) are pre-delivered (see below). All other private keys are generated just-in-time and delivered to their point-of-use authenticated by CAPR.

2.4.1.3 Pre-delivered keys

A **pre-delivered key** is one that is generated and delivered to the point(s) of use well in advance of the need to use it. It will be held in the "Pending Active" state at the point of use. This technique applies to the Certification Authority public keys only (CAPU). A stock of these keys is generated and these are installed into the relevant platforms at manufacture.

2.4.2 X.509 public key infrastructure

2.4.2.1 Public key certificates

The standard defines a PKC as a large, information-rich structure in ASN.1. This is impractical and unnecessary for the purposes of a closed community such as Pathway. This design therefore uses a subset of the X.509 semantics and implement them in a data structure optimised for the chosen programming language; this is defined in section 3.1.1

FUJ00098154

Note: the fit of the X.509 fields to the Pathway requirement is not ideal. The key identifiers that have been proposed for Pathway KM have to be modelled as X.509 V3 extension fields.

2.4.2.2 Certificate revocation lists

The same comments apply as to PKC (above). See section 3.1.2 for details.

2.4.3 Key management process model

In the Introduction to this document, the key management problem space was presented in the "fan diagram" (Figure 2). The principal key processes of the ISO model are mapped onto the fan as shown in Figure 6 (storage and sub-states excluded).

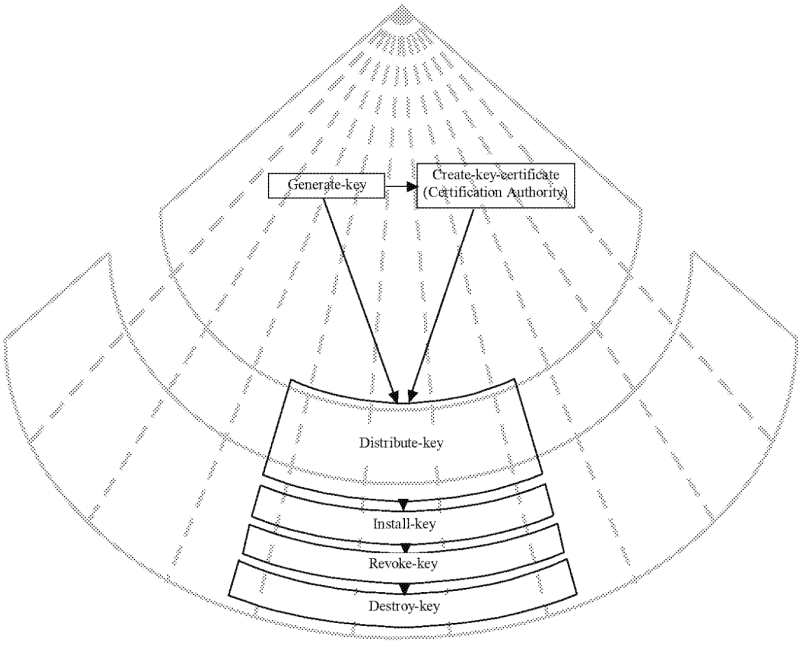


Figure 6. Mapping the standards to Pathway key management

2.5 Key changes

A key change is the co-ordinated procedure of moving the currently active key into the post-active state while moving another key from the pre-active to the active state. The old key might optionally be destroyed. The new key might be a just-in-time key, which implies that it must be generated during the procedure of key-change, since the pre-active state is only transitory for these keys.

When a symmetric key changes to the active state in a client using the key for encryption, the key must be made available to all clients that use the key for decryption in order to sustain the cryptographic relationships between users of the key. This may be achieved either by using a key-ring containing old and new keys in the decrypting clients or by ensuring simultaneous changes amongst all users of the key.

When an asymmetric private key changes, the corresponding public key must be available to the recipients of material protected by that key. For asymmetric public keys, new and old keys overlap in the active state for some time until it is known that the old keys are no longer required (i.e., until it is believed that all data protected under the corresponding private key has been processed).

FUJ00098154

2.6 Revocation and latency

In outline, latency is the period between a message being (validly) signed, and the signature being verified by a recipient. It is possible that a message can be validly signed, but the key becomes invalid before the verification takes place. Thus the verification will 'fail' in some fashion.

In the case of benefit payments for example, a benefit payment can be signed at the centre, and delivered to a Post Office, where it can lie for up to 3 months before being collected, at which point the signature is verified. At this point the verification key might have timed out, or might have been explicitly revoked.

Several million messages could have reached this state.

A similar problem arises with SI messages which can be held in depots for long periods (months).

Business requirements may dictate that some applications accept signatures which verify against an expired or revoked PKC. However, while the KM data structures allow for more information to be passed to the application, at NR2+, the signature verification functions only report success or failure.

Key management standards address this subject as follows.

2.6.1 ISO

The main relevant feature of the ISO standard in this area is the separation of deactivation from destruction. Thus a key can be deactivated (e.g., timed out, revoked), but is not destroyed. It goes into a post-active state, and can in some circumstances be re-activated. Destruction is an action that can be taken on a post-active key.

> "A public key may remain in the Active or Inactive state for an indefinite time after its related private key has been deactivated or destroyed."

> "After a key is revoked it may only be used for decipherment and verification."

> "Whether public key certificates expire or are revoked, copies of old public key certificates shall be retained by the issuing CA for the time required by prudent business practice, law and regulations."

In the following two quotations, the interpretation of the phrase "keying material sent and protected by that public key certificate" is not entirely clear. The safest reading is to presume that the public key contained in the certificate and any key material that has ever been delivered protected by that key (and so on recursively in general).

> "When a public key certificate is revoked because of suspected or actual compromise of a private key, all keying material ever sent and protected by that public key certificate … should be discontinued immediately."

> "When a public key certificate is revoked for reasons other than actual or suspected compromise, all keying material sent and protected by that public key certificate … should be replaced as soon as is operationally convenient."

2.6.2 X.509

Some points relevant to this area: (paraphrased)

- The verification code should check the CRL (Certificate Revocation List) and warn if the CRL is not available or is out of date.

FUJ00098154

- Local policy should be in place to decide format of warning, whether the date of the last CRL is returned, and whether to accept validation with no CRL check.
- If revoked certificates are encountered the user should be warned (stringently). Can include date of revocation. Local policy to decide if accept.
- If no human interaction involved, the verification interface must include parameters to tell the verification code what to do in these circumstances.
- Expired certificates will normally be removed (but down to local policy)
- The private key corresponding to a certified public key is typically used over a different period from the public key.

2.6.3 Pathway working policy

2.6.3.1 Symmetric Keys

Built-in expiry dates and certificate revocation lists do not apply to symmetric keys in the Pathway KM design. Instead the following policies apply to symmetric encryption of data streams:

- symmetric keys managed by the KM Controller are reissued according to the routine maintenance cycle of approximately 2 years or when a compromise has been detected.
- Transient symmetric keys as generated in a Diffie-Hellman exchange are discarded immediately after use.
- Symmetric keys used as PINs that are managed locally by a KM client are changed when the key material they protect is changed or recovered. The local user (a POM in the case of the PMMC PIN) may also elect to change a PIN. (Note: VPN PINs are centrally managed, so this does not apply).
- When a symmetric encryption key is issued to a client that uses it for encryption, the client will transfer to using the new key as soon as the business functionality and availability of any key encryption keys for the new key permit.
- When a symmetric encryption key is issued to a client that uses it for decryption, the client will generally add the key to a key ring of keys available for decryption. The key ring will hold a small fixed number of keys and so adding the new key will generally cause an old key to be removed from the key ring. Use of a key ring is not mandatory if co-ordination can be achieved by other means.

2.6.3.2 Asymmetric Keys

Certification by the Pathway Certification Authority (CA), certificate revocation lists and built-in expiry dates are used to manage asymmetric keys in the Pathway KM design. The following policies apply to the DSA keys supported by the bespoke crypto functions for Pathway.

- For the purposes of calculating expiry dates, the life of a private key that is distributed over a network begins when it first goes on-line. The life of a private key that is either generated for immediate use on the platform that uses it or that is loaded from magnetic media begins when it is first generated or loaded.
- When a client checks for expiry it should use as the effective date the later of its system clock and the timestamp on its certificate revocation list. This prevents an expired key being reinstated by winding the system clock back.

FUJ00098154

- Data signed under a certificate certified under a revoked CA key must fail to verify regardless of the date of the certificate and of the date of compromise of the CA key.
- Certificate expiry dates are absolute. The expected policy is that 768-bit DSA keys expire after 2 years and 1024-bit DSA keys expire after 8 years.
- A private key expires when the certificate holding the public counterpart expires.
- Every public/private key pair is allocated a unique identifier (its "key tag"). Key tags are never re-used.
- A private key is revoked by including its key tag in a certificate revocation list signed using the CA key. This should normally only done when the appropriate client has received and begun to use the new private key (since otherwise existing business will be disrupted). Revocation is not undoable.
- A private key is routinely changed some time before it expires, to allow time for all data signed under the key to be processed and verified before the expiry date. The period between withdrawal and expiry is the maximum expected latency period. This period will vary from application to application.
- Keys are never revoked automatically, since it always requires business judgment to assess the risks of revoking a key. Latency periods are defined for each asymmetric key for the purpose of calculating the dates of routine key changes and for informational purposes only.
- If a private key is suspected of being compromised, the actions to be taken are replacing the private key and revoking the compromised key. These actions are taken at the discretion of the Pathway Key manager depending on the perceived commercial risks. Typically the private key will be replaced as soon as it is expedient to do so, while revocation may be deferred to reduce the cost of spurious rejections.
- In PO outlets (and potentially other clients where there may be a significant delay in delivering new public key certificates), public key certificates are provided with spares. When a key is revoked, the corresponding spare becomes the current certificate and a new spare is provided by KM. (This process only applies to keys held in certificates, and not to the CA key).
- At NR2+, the policy is that a verification of a signature using a revoked key will fail. In subsequent releases, a facility may be provided allowing the date and reason for revocation to be taken into consideration.

The policies for asymmetric keys supported by third-party products such as Utimaco VPN will be to some extent dictated by the product. Where possible, policies similar to the above will be applied.

2.6.3.3 Certification Authority Key

The certification signs public key certificates using a public/private key pair CAPU/CAPR. A life-time stock of CAPU values is made available to every client that uses PKCs. CAPU values may be revoked by the usual CRL mechanism. In addition to the requirements identified in section 2.9 of [KMREQ], the following policy applies:

- The CAPU values are used in a fixed order, say $CAPU_1$, $CAPU_2$, … Revocation of $CAPU_i$ is only permitted when in a CRL certified with $CAPU_j$ where $j > i$.

FUJ00098154

2.7 Cryptosystem

The cryptosystem of choice both in the Key Management System itself and in its clients comprises algorithms approved by HMG and supported by the Layer 7 software supplied by Sapher Servers Ltd. In some clients, notably the CAPS link, Layer 7 is not available for the target platforms and bespoke implementations of the appropriate HMG algorithms are used. In other clients, notably the Utimaco VPN, the cryptosystem is defined by a product vendor rather than by Pathway.

The HMG algorithms used are identified in sections 2.7.1 to 2.7.5 below.

2.7.1 Symmetric Key Encryption

The algorithm for encryption using a symmetric (i.e., shared secret) key is Red Pike. See CESG documentation for details of the algorithm.

Red Pike uses a 64-bit key and encrypts data in 64-bit blocks. A block cipher like Red Pike may be used in several modes, see "Applied Cryptography" [SCHNEIER] for details. For fixed-size messages that will fit in a single 64-bit block (typically such data is a key), Pathway crypto applications generally use Red Pike in Electronic Code Book (ECB) mode (i.e. they just use the cipher directly to encrypt a single block under a given key). Variable length data or data exceeding 64 bits in length is generally encrypted using Cipher Block Chaining (CBC) mode. In addition to the key, CBC encryption requires a random, or at least time-varying, public 64-bit initialisation vector (IV) to be transmitted with the data. In both ECB and CBC modes, the cipher text is a multiple of 64-bits in length, and, with CBC, it is the application's responsibility to transmit (or know) the length of the plain text.

Symmetric encryption may also be used for authentication, in the sense that if party A and party B have a shared secret symmetric key *AK* say, then A can authenticate itself to B by sending B a message including a public value (e.g., a name for A) encrypted under *AK*. (For adequate security, the message should also include a nonce to hinder replay and birthday-book attacks).

Since 64 bit keys may become amenable to brute force attacks of moderate cost within the next 5 to 10 years, it is a design goal of the Pathway KM service to facilitate a future upgrade to use a symmetric algorithm with a longer key

Pathway Crypto design documentation commonly uses the abbreviation *(X)K* to mean data item *X* encrypted using Red Pike under key *K*. The abbreviation *[X]K* is used to mean a data item *X* sealed using Red Pike key *K*: this comprises *X* in clear together with a cryptographic checksum derived from *X* and *K*. Where Layer 7 is used this checksum is the X9.9 compliant message authentication code defined by Layer 7 using Red Pike as the block cipher.

2.7.2 Session Key Exchange

Two parties may agree on a shared secret over a potentially insecure communications path using the Diffie-Hellman algorithm. The mathematics and potential applications of this algorithm are described in "Applied Cryptography" [SCHNEIER]. A brief summary of the algorithm is as follows:

- Parties *A* and *B* wish to share a secret; a prime number *N* and a base number *g* have been agreed in advance (as public values that may be used for many exchanges). All arithmetic in the exchange is carried out modulo *N*
- In parallel, or in sequence (either order):
  - *A* generates at random a private secret value *x* and transmits a public value g^x to *B;*
  - *B* generates at random private secret value *y* and transmits a public value g^y to *A;*

FUJ00098154

- Using the secret value x and the transmitted value g^y, A can now compute $g^{xy} = (g^y)^x$; In the same way, B can compute $g^{xy} = (g^x)^y$; this common value, g^{xy}, now known by both A and B is the shared secret.

For Pathway, N and g are 1024-bit numbers (i.e., they lie in the range 2^{1023} to 2^{1024}) and the private secrets are 160-bit numbers (i.e., they lie in the range 2^{159} to 2^{160}). The private secrets x and y are sometimes (correctly) referred to as "exponents", as also (incorrectly) are the public values g^x and g^y.

The 1024-bit shared secret may be used either directly (via an XOR) to encrypt up to 1024 bits of data or indirectly to communicate a RED PIKE key with which bulk data may be encrypted.

The algorithm as above stated is vulnerable to a man in the middle attack. To defend against this attack each party must provide proof of origin of its public values (g^x must come from A and g^y from B). A digital signature using an asymmetric public/private key pair or a seal derived from a shared secret symmetric key may be used to provide this proof.

Either A or B or both may defend against the man in the middle by signing the public value they send to the other party.

2.7.3 Asymmetric Key Encryption

Asymmetric key encryption is carried out using the Thames Bridge Key Management Algorithm (TBKMA). Mathematically this involves the same computations as the Diffie-Hellman algorithm describe in section 2.7.2 above. Operationally, party A, say, generates the value g^x and publishes it as a permanent public key. Other parties then proceed as party B in the description in section 2.7.2, generating transient public key values g^y with which they can communicate "for-your-eyes-only" information to A.

Note that unlike cryptosystems based on RSA, the keys used for asymmetric key encryption are mathematically different from those used for digital signature.

This technique is not used at NR2+.

2.7.4 Digital Signature

Digital signing is done using the US National Institute of Standards and Technology's Digital Signature Algorithm, DSA. The mathematics of this is discussed in chapter 20 of "Applied Cryptography" [SCHNEIER]. In summary, given certain public parameters, p, q and g, a private key, x, and a public key, y, DSA allows a party A to compute from a message M a signature $S = (r, s)$, such that other parties can efficiently check that it is computationally highly improbable that any party not privy to x could have generated the same signature for that message. (In fact, the Layer 7 implementation packages S in a three part structure, (d, r, s), where $d = SHA(M)$.)

DSA allows a choice of a modulus that influences the security of the algorithm; for Pathway, 768-bit and 1024-bit moduli are used. See the Layer 7 documentation for the resulting sizes of the public and private keys and of the digital signatures. As specified by NIST, SHA (see section 2.7.5 below) is used to compute the one-way hash value of the data being signed.

Note that unlike cryptosystems based on RSA, the keys used for digital signature are mathematically different from those used for asymmetric key encryption.

Pathway Crypto design documentation commonly uses the abbreviation $\{X\}K$ for data item X accompanied by the digital signature obtained from X using private key K.

FUJ00098154

2.7.5 One-way Hash Algorithm

The one-way hash algorithm used is the US National Institute of Standards and Technology's Secure Hash Algorithm, SHA. The mathematics of this algorithm is discussed in chapter 18 of "Applied Cryptography" [SCHNEIER]. In summary, given a message, *M*, say, of length less than 2^{64} bits, SHA computes a 160-bit hash value, $h = SHA(M)$, such that it is computationally difficult to deduce *M* from *h* or to find an alternative message M' such that $SHA(M') = h$.

2.7.6 Key Naming

Layer 7 provides a systematic method for naming private keys. Keys are known by a key tag comprising 4 16-bit numbers. This is used by the KM system to ensure that keys throughout the system are unique. To facilitate migration and future extension of the system, knowledge of the mapping of protection domains and client names onto key tags should not be exploited by client KM software.

2.8 Riposte

The Key Management Service makes use of the Riposte Message Server for communication of keys and other information. An overview of Riposte is given in section 7.3 of "Technical Environment Description" [TED].

The main Riposte mechanism used in KMS is its persistent objects which logically provides a resilient store of named shared objects. See section 7.3.5 of "Technical Environment Description" [TED] for more information.

In this document we use the terms "harvester" and "loader" to refer to particular kinds of Riposte agent in the same sense as these terms are used in section 5.2.3 of "Technical Environment Description" [TED]: a harvester transfers data out of the Riposte Message Server; a loader transfers data into it.

2.9 Virtual Private Networks

At NR2+ and later, the Pathway system uses a Virtual Private Network (VPN) product to provide confidentiality and authentication over an IP network. The VPN product is provided by Utimaco and is supported by a public key infrastructure system also supplied by Utimaco. This PKI system is integrated into the Pathway KMS, which manages the Utimaco key generation and certification processes and provides the route whereby Utimaco key material is distributed to KM clients that need it.

The Utimaco certification process depends on an RSA public/private key pair (Utimaco CA) for which the private key is subject to strong physical protection. The public part of this key pair is communicated to all parties wishing to use VPN.

Parties communicate within a VPN via encrypted IP packets passed as the data payload of an in-clear IP packet. Encryption and decryption is via session keys established when one party first attempts to send a packet to another. The Utimaco VPN system provides authentication and confidentiality using RSA cryptography to establish these session keys. Each party using VPN is provided with a VPN key containing a public/private RSA key pair with the public key certified by the Utimaco CA. When two parties wish to communicate they exchange their public key certificates, validate them against the Utimaco CA public key and use them to generate a shared secret session key. VPN does not therefore require separate distribution of public keys.

[Note: IP is not session-oriented; in practice, the lifetime of a VPN session key is a time interval determined by configuration parameters.]

FUJ00098154

2.10 Client Names

The KM system needs a uniform scheme for identifying its clients. Many of its clients are themselves multi-node systems (e.g., a PO outlet comprises 1 or more counter PCs) and for some purposes, the KM system needs to identify individual nodes within one logical client. All clients will have one or more names defined by the Pathway system (e.g., at NR2, FAD codes and Riposte GroupIds are used to identify PO outlets) The KM system also generates its own unique identifiers for the clients. The term "name" in this design means an identifier derived from an external source (e.g., FAD codes are used as the names of PO outlets); the term "id" means a KM-generated identifier (e.g., the numeric owner-id of a private key).

FUJ00098154

3. SYSTEM DESIGN

The purpose of this high level design is to define the subsystems which implement key management for Pathway at NR2+. This section is concerned with identifying and scoping those subsystems and the interfaces between them.

An abstract view of the subsystems and main data flows of the Key Management System as seen by a particular client in a particular protection domain is given in Figure 7. Figure 7 does not show the physical architecture of the key management controller. This is discussed in section 3.2.1. The figure does show (for purposes of assessing security threats) the distribution of the mechanisms amongst the Campus, the communications mechanisms (LAN, WAN, magnetic disk, paper) and the client.

For simplicity in the KM design, it is appropriate to consider all electronic networks and links used for the distribution of key material as insecure and unreliable. Thus non-public key material must be encrypted before transmission over a network and public key material should include an adequate integrity check.

The abstraction of Figure 7 is realised in several different ways according as key material is distributed (i) fully automatically, (ii) to a token manufactured at a remote site, (iii) to a token (diskette) manufactured at the Pathway campus. At NR2+, case (ii) comprises only the case of the keys that are stored on a PMMC for use in PO outlets.The realisation of Figure 7 also varies according as monitoring is done automatically or by a human procedure. The various instantiations of the data flow model used at NR2+ are shown in Figure 8, Figure 9, Figure 10 and Figure 11.

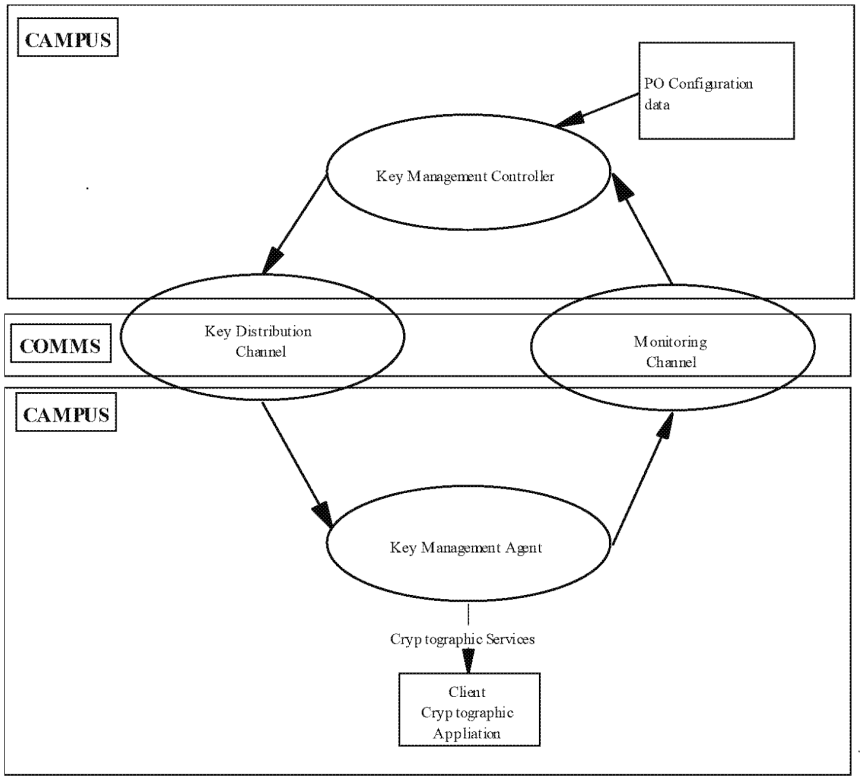


Figure 7. KM Data Flow - Abstract View

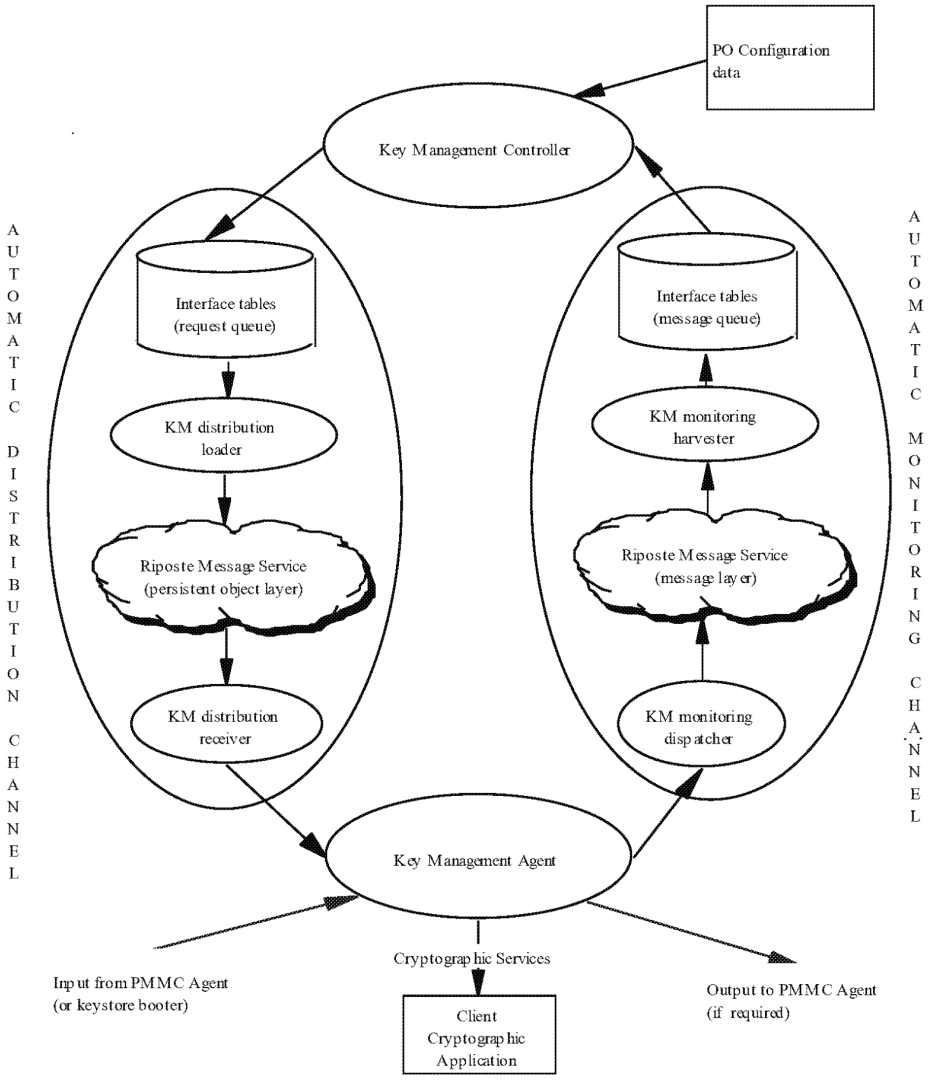
FUJ00098154

686 *Note: The KM Controller was formerly called KM Centre; the intention is that the KM Controller is*
687 *the software and hardware that implement the central KM functionality, not the Cryptographic Key*
688 *Manager's centre of operations.*

689

FUJ00098154

690 For many clients, in particular the Post Office outlets, the public and some secret keys can be managed
691 fully automatically. Such clients have the Riposte infrastructure available, and Riposte provides a
692 convenient model for implementing the required communications between KMC and these clients. The
693 Riposte service is not available during system start-up; consequently, the KM client agent software that
694 runs on the clients that use this distribution model must communicate with the PMMC agent or Keystore
695 Booter software that handles key management on the client before the Riposte service starts (see Figure 9
696 and Figure 11). The resulting architecture is shown in Figure 8 below. The distribution and monitoring
697 channels in this figure are defined in more detail in sections 3.3 and 3.6 below.



698

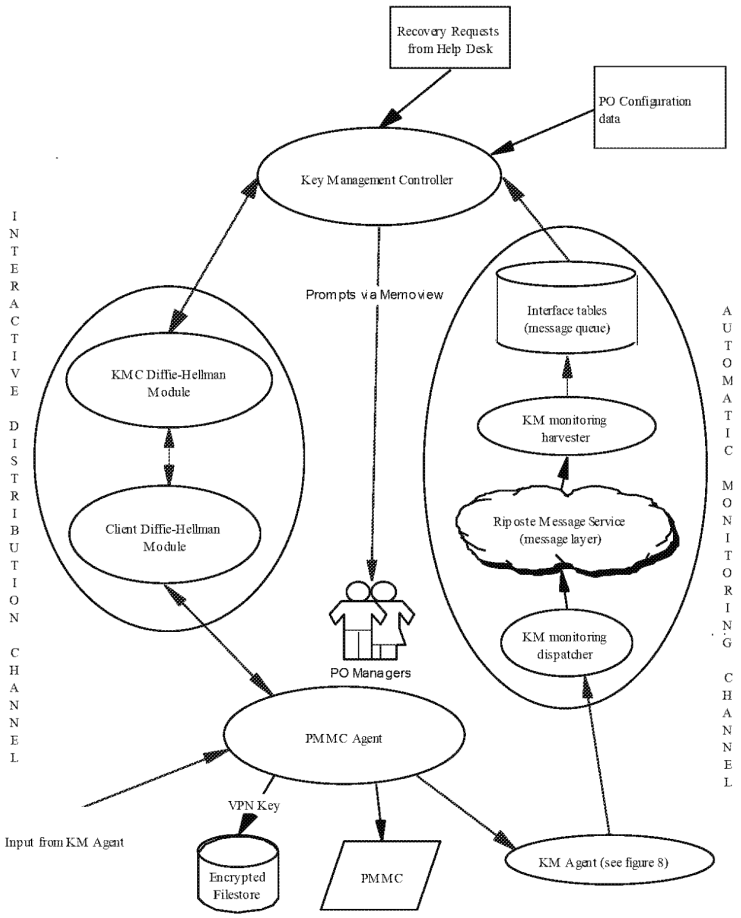
699 Figure 8. KM Data Flow: automatic distribution and monitoring

FUJ00098154

700 During personalisation of a PO counter PC at roll-out and under certain other circumstances, the Riposte
701 service and hence the automatic distribution channel is not available. In these circumstances a connection
702 running over TCP/IP is used to transfer key material to the client. This distribution mechanism is referred
703 to as the "interactive channel" and is used for delivering keys that are stored on the PMMC. A software
704 component called the PMMC Agent at the client controls the manufacture of the new PMMC; this
705 includes a GUI part of the Post Office Logon system (PoLo) to guide the POM through this process. At
706 roll-out or when a PC is replaced, the PMMC Agent may also have to store other key material (notably
707 VPN keys) in encrypted filestore. Since this channel requires the cooperation of the Post Office Manager,
708 the MemoView interface is used to send prompts to the POM asking him to do the reboot that inititates
709 the transfer.

710 When using this means of distribution, the keys being delivered may be "master keys" that are not under
711 any other form of protection. The interactive distribution channel must generate a transient session key to
712 encrypt these keys in transit.

713 In this case Riposte will still be available for monitoring purposes. The resulting architecture is shown in
714 Figure 9 below. "Monitoring" via the NT event log is not shown here; we only show the Riposte
715 mechanism.

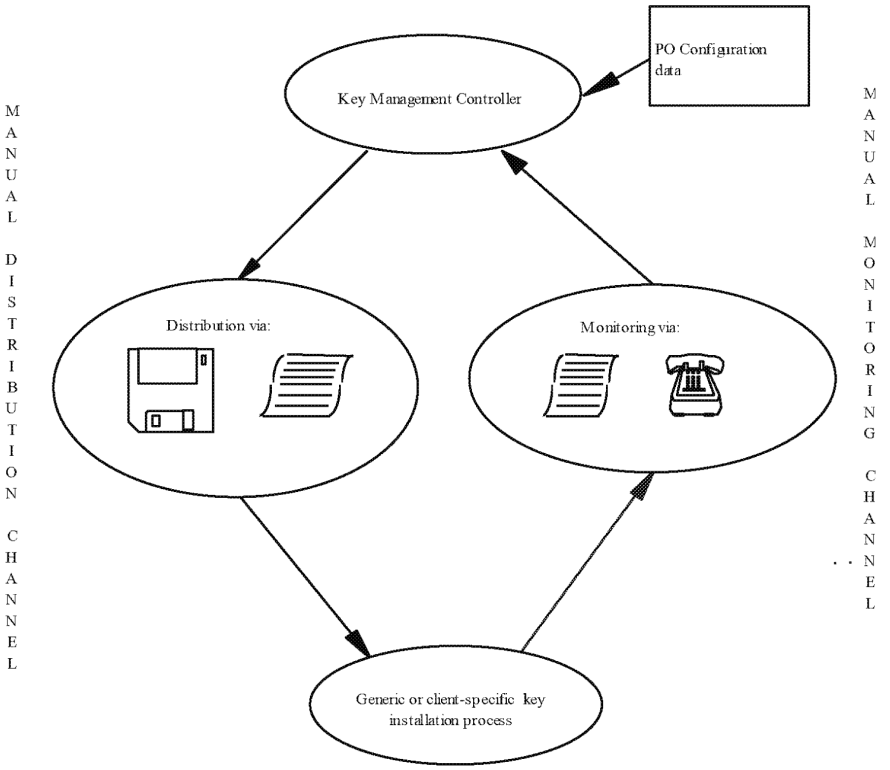


716

717 Figure 9. KM Data Flow: distribution via interactive channel with automatic monitoring

FUJ00098154

718 For specific operational or security considerations in certain protection domains, the KM system supports
719 delivery of key material via non-electronic means. In these cases, key material is generated at the Key
720 Management Controller and issued on paper or a removable disk. The key material is then shipped to the
721 clients by a (human) key custodian. The case where monitoring of key changes is also carried out by a
722 manual procedure is shown in Figure 10.



723

724 Figure 10. KM Data Flow: manual distribution and monitoring

725

FUJ00098154

In some protection domains, it is operationally convenient to support key distribution of some key material via a manual channel but with monitoring (and management of other key material) done automatically via Riposte. In these cases, the architecture is a variant of the fully automatic distribution mechanisms shown in Figure 8; this variant is shown in Figure 11 and uses a software component called the Key Store Booter to substitute for the PMMC agent of Figure 8. The Key Store Booter reads key material from the distribution medium used in the manual channel and arranges for the KM client agent to operate in much the same environment as in the fully automatic case. No communication from the KM client agent to the Key Store Booter is required.

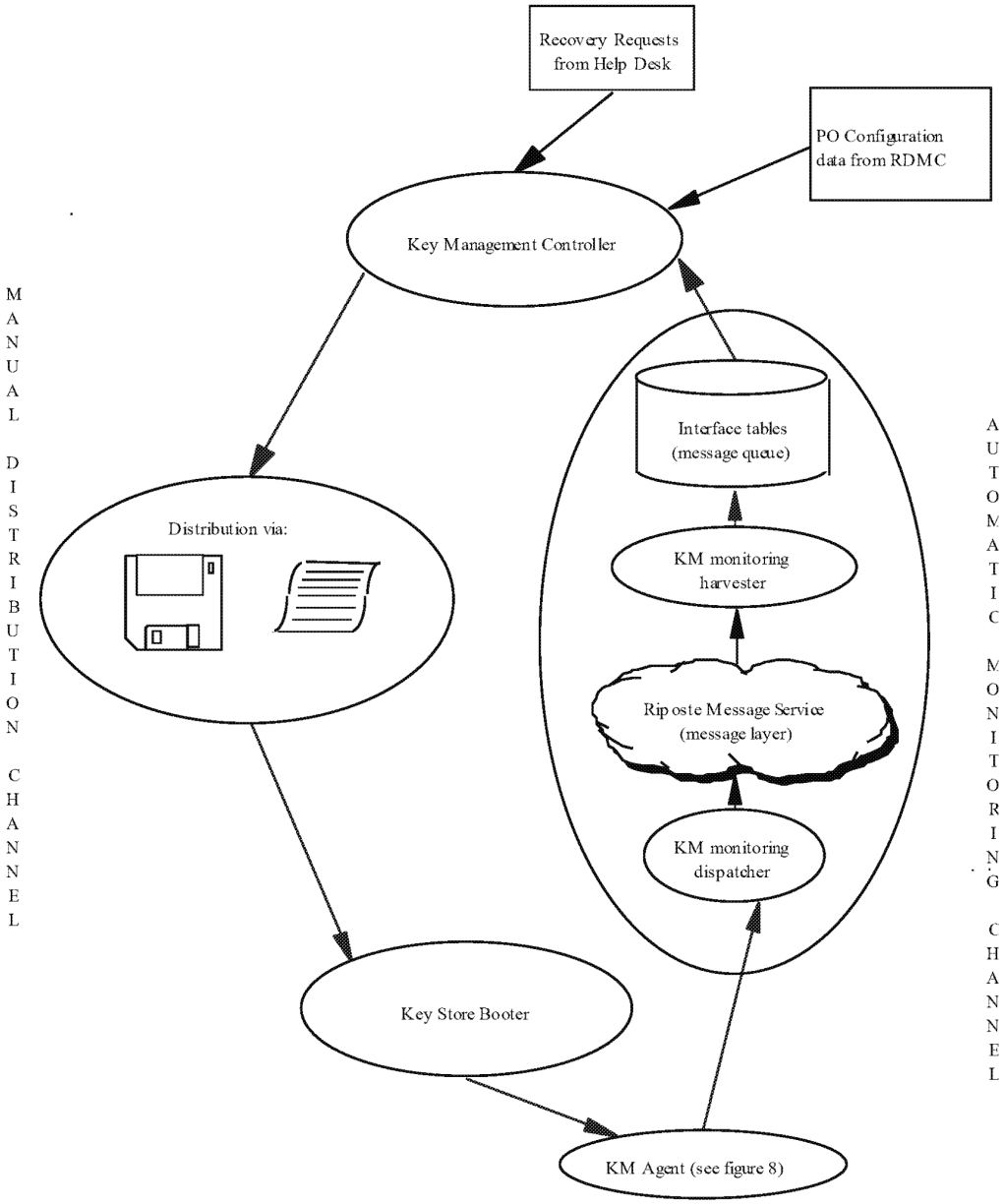


Figure 11. KM Data Flow: distribution via manual channel with automatic monitoring

FUJ00098154

736

737 In Figure 8, Figure 9, Figure 10 and Figure 11, we have identified the software subsystems shown in the
738 following table. Refer to the indicated sections of this document for more information about each
739 subsystem. No specific software support is currently envisaged for the monitoring route in Figure 10
740 although the KMC will provide a means for tracking management information obtained via this route see
741 section 3.7 below.

742

Automatic distribution channel	3.3
Automatic monitoring channel	3.6
Automatic monitoring channel	3.6
Interactive distribution channel	3.8
Key Management Client Agent	3.5
Key management controller	3.2
Key management controller	3.2
Key management controller	3.2
Key Store Booter	3.4
PMMC Agent	3.9

743

744 3.1 Principal data structures

745 3.1.1 Public key certificate

746 The data in a PKC is signed using the CA private key. Integrity and authenticity of a PKC are guaranteed
747 by this signature. A PKC can therefore be distributed openly without further protection, even by lodging
748 it in a public repository, since any user of the certificate may use the CA public key (CAPU) to verify it.

749 The data in a PKC includes all the fields shown in the following table:

Cert-id	is an identifier for the certificate
CAKey-Tag	is the identifier of the CAPU that should be used to validate the signature on the certificate
Owner Name	identifies the owner of the private key corresponding to the Public Key in the certificate (see below)
Protection Domain	the protection domain in which this PKC is to be used
Owner Key-tag	is the identifier of the key pair of which the key in the certificate is the public component.
Owner's PK	is the public key of the key pair owned by Owner-Id
Valid-From Date	is the date and time this certificate is to become valid.

FUJ00098154

Expiry Date	is the date and time after which use of the certificate will trigger expiry errors.

The detailed representation of these fields is defined in "Detailed Design of Certification" [KMCAWDES]. Each private signing key is associated with an "owner" - the party that signs using the key. The "Owner Name" is the name of the party in question. Several platforms that sign data will typically share one owner name (e.g., all the counter PCs in a PO outlet or all the POCL TIP gateways at the Pathway Campus). The following classes of owners have been identified for DSA signatures generated and verified by Pathway-supplied code at NR2+.

- A Post Office outlet (Owner-Id = FAD code at NR2+)
- the KMA
- the PA signing agents
- the Software Issue signing agent
- ICL Pathway (identifying itself to POCL)
- ICL Pathway (identifying itself to AP clients)
- POCL (identifying itself to ICL Pathway)

Since during the lifetime of the KM system, it is intended that Pathway applications migrate from using FAD codes to OUCs as the means of identifying a PO outlet. To accommodate this, PKCs will support inclusion of both forms of name (and the KMC will generate PKCs with both forms). It is then the responsibility of the business applications invoking cryptographic functions to supply the appropriate name: the verification of a signature against a PKC will allow either the FAD or the OUC form.

For signing platforms other than PO outlets, the "owner id" is determined by the protection domain in which the signing is carried out.

3.1.2 Certificate revocation list Capsule

A CRL contains the following information, where *n* is the number of keys revoked by the CRL.

Timestamp	Date and time when this list was signed	
× *n*	Key-tag	identity of the key
	Date of Compromise	date before which the key is still believed OK
	Reason	not used at NR2+; reserved for future expansion
Signature	Digital signature using CAPR	

At NR2+, the "Date of Compromise" and "Reason" fields do not affect revocation (see section 3.12.4). X.509 defines possible values for the reason field for potential use in later releases.

Since key tags are never re-used, the key tag is sufficient to identify the key for all purposes. For convenience of implementation, information such as the owner id and the key type is in fact encoded in the key tag, but this is not a requirement of this design.

FUJ00098154

3.1.3 Confidential Key Capsule

A confidential key capsule contains either a DSA private key or a Red Pike key or third-party key material protected under a key encryption key (see section 3.11). The protocols of section 3.12 require the following information to be available in the key capsule.

Protection Domain	The protection domain in which this key is intended to be used
Owner Name	The name of the party which is intended to use this key for signing or encryption.
Key Tag	Layer 7 key tag for this key
TK Tag	Layer 7 key tag for the key encryption key (TK)
Serial number	Serial number for this capsule
Layer 7 Key Data Payload	Either the Layer 7 key transport data or the encrypted third-party key material with check bytes encrypted under TK.

Some of this information may overlap in the physical representation; e.g., the serial number may be extracted from the Key Tag. The details of the representation are to be defined in "KMA Design" [KMAPDES].

Inside the Layer 7 Key Data, the raw bit pattern of the key is encrypted under the TK.

The "check bytes" mentioned above are described in 3.11.

3.1.4 CA Public Key Capsule

CA public keys are packaged as NT files using the Layer 7 key transport format. From the point of view of this high level design, the structure is as follows:

Key Tag	Layer 7 key tag for this CA public key.
Serial number	Serial number for this CA public key.
Layer 7 Key Data	The Layer 7 key transport data for use in carrying out verification with this CA public key.

In fact, the first two fields here are physically represented within the Layer 7 key data and they are only made visible because of the role they play in implementing the policies of section 2.6.3.3.

Note that CA keys themselves are not managed by the automatic key management mechanisms described in this document. CAPU/CAPR pairs are manufactured by a Managed Key Service operated in the List-X secure environment at ICL BRA01. The CAPR (private) keys are delivered into secure storage under the control of the Pathway Key Manager at ICL FEL01. A life-time stock of CAPU (public) keys are delivered into Celestica for inclusion in the software build of all platforms that need them. In a disaster recovery situation where the stock of CA keys needs to be extended because of compromise, this may be done via Pathway's Tivoli software distribution mechanisms to add new CA key files. Automatic oordination of such a disaster recovery process is outside the scope of this document.

3.1.5 CAPU Check Capsule

A CAPU check capsule (see section 3.12.5) contains the following information:

FUJ00098154

SHA(CAPUS)	The 160 bit SHA value for the files comprising the stock of CA public keys.
Signature	Digital signature using KIPR.
KICERT	KIPU certificate.

805

806 3.1.6 Protocol Requests and Acknowledgments

807 The various protocols of sections 3.10.2 and 3.12 involve requests sent from the KM controller to clients
808 and acknowledgments from the clients to the KM controller containing information as follows:

809

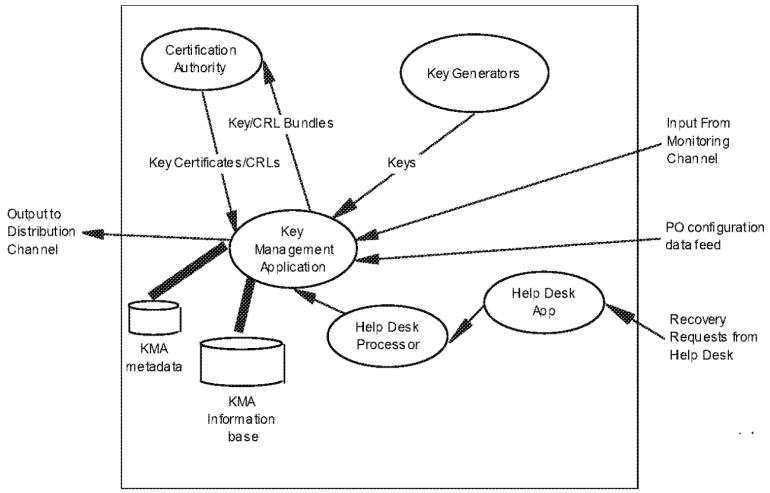
Name	Contents
PMMCKeyChangeReq	Client Name Key tags for new PMMC keys
PMMCKeyChangeAck	Client Name
IntExchAck	Client Name
NewPMMCAck	Client Name
ConfK	Confidential key capsule
Ack.Installed.ConfK	Client Name Key tag of installed key
Ack.Received.ConfK	Client Name Key tag of received key
PKC	Public key certificate
Ack.PKC	Client Name Key tag of received key

810 3.2 Key management controller

811 The components and main data flows of the Key Management Controller for a particular protection
812 domain are shown in Figure 12 below. The Key Generator and Secure Key Packaging components may
813 vary from domain to domain, the Certification Authority and Key Management Application do not. The
814 GUI for the Key Management Controller is considered to be an internal part of the Key Management
815 Application and is not shown separately here.

816

FUJ00098154



817

818 Figure 12. Key management controller data flows

819 The targets for the outputs from the KMA to the distribution channels and the security considerations
820 depend on the channels as shown in the following table:

Automatic distribution channel	The target is an interface table accessible to the KM distribution loader (see section 3.3). Any confidential data must be protected under a key encryption key (since the channel will place the data in the Riposte Message Store, which is stored and archived in clear in the campuses). In all current cases, the key encryption key is the Traffic Key TK associated with the recipient of the key
Interactive distribution channel	The target is a buffer in the memory of the KMC Diffie-Hellman module (see section 3.8); the data includes KEKs and is held in clear (the Diffie-Hellman exchange will protect it).
Manual distribution channel	The target is a printer or a diskette drive; the data is all or part of a confidential key; the printed output or diskette will be handled with good physical security.

821 During migration and roll-out, the KMC will have to manage keys in a situation where hundreds of
822 NR2+ PO outlets are coming online every week. Preparation of keys for a new outlet takes time and
823 involves some manual intervention (e.g., to process any key material that needs certification).
824 Throughout the NR2+ lifecycle, some outlets will be closing permanently or temporarily. The KMC
825 therefore requires a feed of PO configuration data to notify it in advance of the appearance of a new and
826 migrating outlets and of the closure of existing outlets. The KMC design supports feeds from multiple
827 sources to allow flexibility in the design of the systems management servers that provide the feed. The
828 design also caters for changes to the data, e.g., to handle an operational delay in the roll-out. Further
829 information about migration and roll-out may be found in [KMMIG].

830 The four components of the Key Management Controller are described in more detail in sections 3.2.2,
831 3.2.3 and 3.2.4 below (the metadata and information base are covered in section 3.2.2).

832 ### 3.2.1 Physical architecture and platforms

833 The requirements document [KMREQ] states that "the KMA functionality must be available to the key
834 manager located in FEL01". This will be achieved via a client-server architecture with the client

FUJ00098154

workstation at FEL01 and a server at each of the Pathway Campuses. One server will act as a standby for the other. The disks containing the KM information base on the standby server will mirror those of the active server via a high speed link. The disks are actually attached to an EMC server which manages the replication. For simplicity in this design we consider the EMC server to be part of the KM server. A spare client workstation is available in one of the campuses.

For simplicity in the KM design, it is appropriate to consider all networks and links used for the distribution of key material as insecure and unreliable. Thus non-public key material must be encrypted before transmission over a network and public key material should include an adequate integrity check. In particular, as the KM information base is replicated via a link between the two campuses, all non-public key material held in the KM information base must be encrypted under a KEK (the KMA key) that is not held in memory on-line.

A Comscire hardware random number generator is fitted on all the KMA platforms that carry out key generation and is used to provide high-quality entropy for those key generators that can use it.

The physical architecture of the KM client platforms is derived from the requirements of the Pathway business and is documented in "Technical Environment Description" [TED]. See also "KMA Design" [KMAPDES].

3.2.2 Key management application

3.2.2.1 Overview

The KMA provides overall control and monitoring of the key management processes. It has the following major features.

1. A database of information about the status of keys: their locations, stages of production and distribution, expiry times, etc. This includes tracking information for keys in the manual distribution channels
2. Functions to instigate the generation of new keys and route them to the CA, online or off-line storage or distribution channels using the key transfer protocols of section 3.12 below.
3. Scheduling and load-balancing of routine key changes so as to manage the key transfer protocols cost-effectively (by smoothing out the load to lie within the bounds identified in section 5.1 below).
4. Functions to instigate processes at clients where the KMA can have direct control.
5. The user interface which gives the Pathway Key Manager access to the above features.
6. Metadata which describes the protection domains and clients that the KMA controls.
7. Provision of prompts and reminders to POMs when counter PCs need to be rebooted to support a key change.

The KMA does not attempt to change keys automatically when potential compromises have occurred. It is up to the Pathway Key Manager to decide whether an event like recovery of a PO outlet after lost PMMC or PIN constitutes a compromise. The KMA reports on such events and allows the Pathway Key Manager to respond to them according to the needs of the business.

The detailed design of the KMA is documented in "KMA Design" [KMAPDES].

Prompts and reminders are to be sent to POMs using the MemoView product.

FUJ00098154

873 *3.2.2.2 Key Protection and Packaging*

874 The private component of each asymmetric key pair and any symmetric data encryption key must be
875 protected both in the KMA's database and during distribution. In the KMA database, any field containing
876 confidential data must be encrypted using the KMA key (see section 4.1.3).. Depending on the security
877 requirements and on the distribution channel the protection during distribution is achieved as follows:

Automatic distribution channel	the key is symmetrically encrypted under a key encryption key.
Interactive distribution channel	the key is encrypted under a transient session key generated as part of the protocol described in 3.12.1
Manual distribution channel	the paper or diskette holding the key is subject to secure manual procedures.

878 Thus in the case of a key delivered via the automatic distribution channel, the KM controller must
879 provide secure packaging for the key. Analogously, the authenticity and integrity of all public keys must
880 also be protected by packaging them in PKCs (this is done by the Certification Authority described in
881 section 3.2.4).

882 The key encryption key, or ("Traffic Key", TK), will be generated by a suitable key generator and must
883 also be delivered to the key client so that the client is able to decrypt the package. The TK must be
884 delivered using cryptographic or physical/procedural protection.

885 Where the TK is delivered electronically, it will be protected under a transient session key shared with
886 the receiving client using a (Protected) Diffie-Hellman exchange. This session key is only held in RAM
887 and is discarded once the TK has been delivered (and needs no packaging or further protection).

888 Where the TK is delivered manually, good physical security must surround the delivery (and subsequent
889 storage, if necessary) of the TK. Wherever possible a key delivered on diskette should not contain a
890 complete data encryption key or private signing key. In fact, in the NR2+ design what is delivered on
891 diskette is usually the TK that protects a Layer 7 black key file held on the clients disks and delivered
892 automatically.

893 The KMA, rather than the key generator component, is responsible for the encryption, decryption and
894 periodic re-encryption of the keys stored in its database. The KMA must ensure that every confidential
895 key in the database is encrypted under the current value of the KMA key. It must also ensure that every
896 confidential key it passes to the automatic distribution channel (see section 3.3) is encrypted under a
897 suitable key encryption key. The KMA delivers keys in clear to the interactive channel (see section 3.8)
898 which must therefore pass the key across the communications layer encrypted under a transient session
899 key. The formats and storage used for keys at the clients is discussed in section 3.11.

900 The KMA key must itself be managed. The KMA carries this out internally and so the KMA does not
901 need to run the usual KM client agent software. The mechanisms for routine and emergency change of
902 the KMA key are specified in [KMAPDES].

903 *3.2.2.3 Module/Process Structure*

904 The KMA is best understood as maintaining a model, in the KMA information base, of the state of
905 progress of key material through the Pathway system. This model is built up on the basis of initial
906 configuration, the KMA's records of requests it has sent to its clients and the acknowledgments received

FUJ00098154

from those clients. Together with the KMA metadata this model is used to control the flow of requests to clients and prompts to human operators. The overall dataflow is shown in Figure 12. A structure diagram giving more detail on the internal structure of the KMA and the people and KM software components it interfaces with are shown in Figure 13. In this diagram: the cells are software modules, each potentially comprising many source files; the arrows indicate calling structure, each arrow pointing from caller to called. Calling arrows labelled "SQL" or "Networked SQL" correspond to communication of control information or data via updates to shared tables. The key generators are shown outside the KMA from the point of view of software structure only - the key generators actually execute on the KMA platforms.

As can be seen from the diagram the KMA software comprises four layers all of which use a DBMS product to manage the database containing the KMA metadata and information base. A design goal for the KMA is to structure the database design so that the division of ownership of information amongst the layers (and their subcomponents) is carefully controlled and defined. This is further discussed in [KMAPDES].

The top layer of the KMA runs on the KMA workstation and communicates with the KMA server via RPC and Networked SQL. This layer provides direct support for the functions that must be carried out by the Key Manager and other workstation users.

The Key Operations Layer provides the server side support for managing the main operations on keys: creation, change, revocation etc.

The Logistics layer is responsible for scheduling and monitoring the KMA's routine tasks and for maintaining the model of the state of key distribution in the information base.

The Primitives layer supports the actions of the Logistics layer by providing basic services such as calling the key generator, re-encrypting a key, or dispatching key material to the automatic channel.

Only the Primitives layer and the Key Manager Functions layer deal with unencrypted key material. No unencrypted key material is passed along any network link.

FUJ00098154

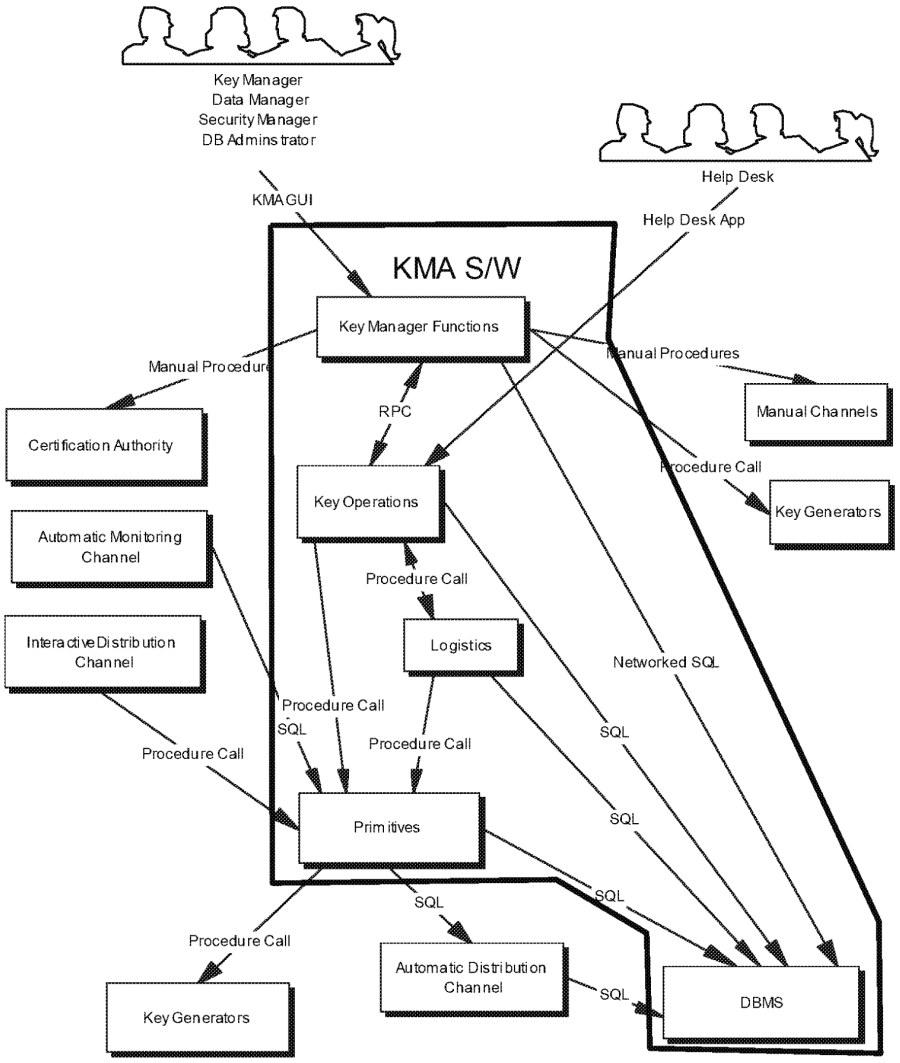


Figure 13. KMA Structure Diagram

3.2.3 Key generators

Each key generator is technology-specific. That is to say, it will produce keys in a particular format compatible with the technology of the target cryptographic process. For example: the *Layer 7 Red Pike* key generator produces keys in a key transport file that can be imported by the FTMS cryptographic functions (and others), which have been implemented with the Layer 7 cryptographic tool kit.

To simplify the interface between the KMA and the key generators, the key generators software component provides a simple interface to the key generation facilities of Layer 7 and other products.

3.2.4 Certification Authority

The Certification Authority (CA) is an application which takes public keys as input and packages them in public key certificates (PKC). The certificates are signed with the CA private key. The CA also signs

FUJ00098154

CRLs. The CA is implemented on a dedicated off-line platform, the CA workstation (CAW). Data is transferred between the CAW and the KMA workstation using removable disks. These disks are held in secure storage when not in use.

Because of the widespread trust placed in the CA's signature, it is essential that the CA workstation be designed and implemented to strongly protect the CA private key. The CA workstation will be a secure off-line facility.

Random numbers required by the CA application will be supplied by a Comscire hardware random number generator.

Bundles of keys and CRLs passed to the CA for signing are signed under the key KIPR by the KM Application; in its turn, the CA signs the bundles of signed PKCs and CRLs under the key CAPR. This provides an integrity check mainly to defend against operator errors.

The CA handles both DSA keys (using Layer 7) and RSA keys (using the Utimaco product). Its design is described in detail in [KMCAWDES].

The CA platform also provides both generation and certification of VPN keys using the Utimaco product. VPN keys are produced in response to requests included by the KMA in the bundles it generates for transfer to the CAW.

3.2.5 Help Desk

The SMC provides a Help Desk giving the second line of support for the majority of the Pathway system's user community, including the POMs. SMC needs access to the KMA to enable exceptional deliveries of key material needed as a result of hardware or software failures, operator errors or other operational problems. The dominant cases involve recovery of a PO outlet after: failure of a gateway PC; loss of or damage to the PMMC; loss of or damage to the printed PIN.

A simple client-server application is provided to support these requests from the help desk. It comprises a client offering a user interface ("help desk GUI") for the SMC staff and a server component ("help desk processor") which communicates the requests to the KMA. The design of this system is further discussed in [KMAPDES] and its descendant documents. The main functional role of the help desk in this high level design is in defining some of the circumstances under which the KMA is prepared to engage in a certain key transfer protocol described in section 3.12.1.

3.3 Automatic distribution channel

3.3.1 General description

The automatic distribution channel provides to the KMA the service of delivering key material and key management requests to the key clients. It is implemented as a service layer over the Riposte Message Service. The channel provides for delivery of write-once key capsules as named objects in the client's logical data store and for notification to the client software of the arrival of a new object. The API at the client allows interrogation of the available key capsules sufficient to implement the various protocols of section 3.12.

Some key material, e.g., PKCs destined for the PO outlets, has to be delivered to a community comprising many clients. The Pathway implementation does provide a mechanism for automatic delivery to all PO counters, but this does not cover all of the KM requirement for delivery of keys to sets of clients. Since the KM Controller needs to have all the relevant client identification information, it has been decided that it will be responsible for managing its own distribution lists. The detailed design of the automatic distribution channel therefore shows the interface tables as being organised into two tables: a

FUJ00098154

985 dynamically changing list of requests and a more slowly changing distribution list which associates
986 logical destinations with sets of client identifiers. This prevents the request queue needing to contain a
987 large amount of repeated data.

988 In addition to requests to deliver key material and key management requests, the channel also supports
989 requests by the KMA to delete public keys and other shared material when they are no longer needed.

990 Some important attributes of the automatic distribution channel are summarised in the following table:

As-soon-as-possible delivery:	any object the dispatcher sends will be delivered to the client platform as soon as the infrastructure allows. If, for example, the receiving platform is switched off at the time of dispatch, the channel will store the message until the platform is restarted and will deliver the message as soon as the necessary platform resources are available.
Guaranteed delivery	any object sent to a client through the channel will eventually arrive at its destination, if the destination exists.
Exclusive delivery:	an object will only be delivered to the client for which it is intended (although intermediates at the campus and archive traces of the object may remain).
Ownership/deletion policy	1. **Key management requests and private keys**: these are considered to be owned by the client: once the object has arrived at the client it is the client's responsibility to delete it when it is no longer needed (this is a question of garbage collection rather than secure destruction). 2. **Public keys and other shared material:** these are considered to be owned by the KM Controller: it is the KM Controller's responsibility to delete the object when it is no longer needed.

991 The component breakdown and main dataflows of the automatic distribution channel are shown in
992 Figure 14. The automatic distribution channel interfaces are specified in detail in "KM Automatic
993 Channel Detailed Design" [KMACDES].

FUJ00098154

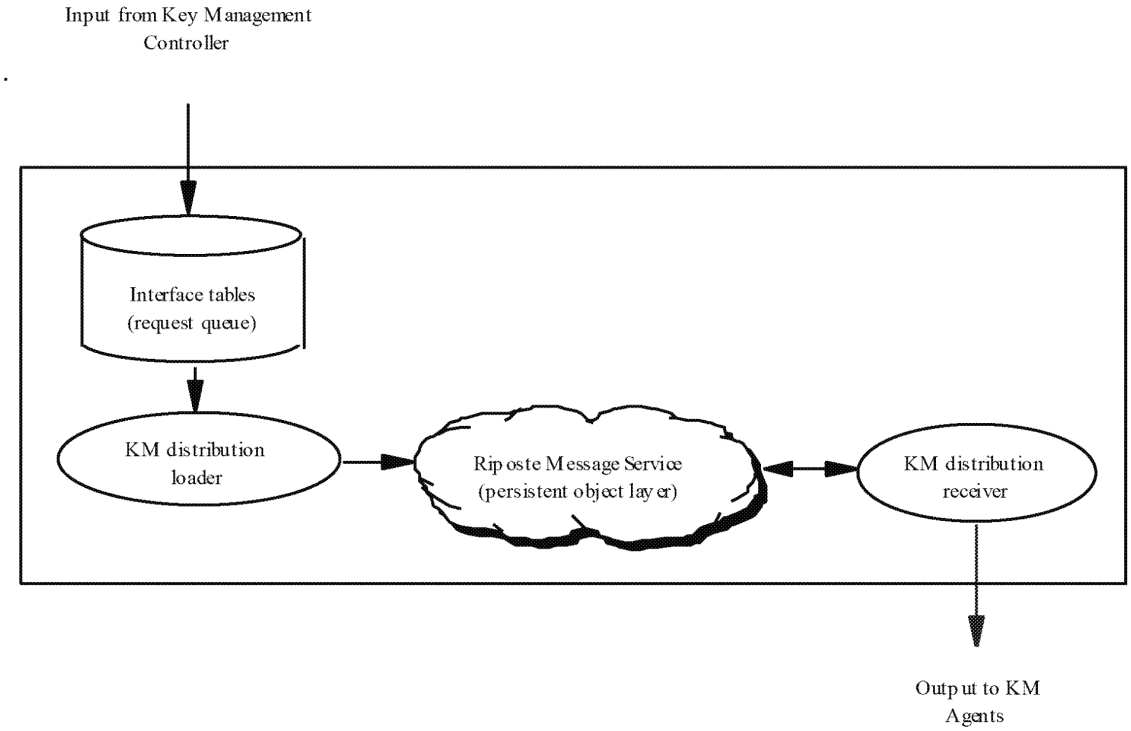


Figure 14. Automatic distribution channel data flows

3.3.2 Dispatch interface

The KMA writes task entries into an interface table, from which the KM distribution loader reads them and takes the necessary action to update the Riposte Message Store. The campus end of the channel comprises the KM distribution loader which reads entries from an interface table produced by the KMC and writes corresponding key data and control information to the Riposte persistent object store for access by the client software.

The interface tables implement a queue of requests from the KMC. Logically there is a single queue of requests going out from the KMC; this may be physically divided amongst several tables to avoid excessive duplication of public key material. The precise organisation is described on [KMACDES]. Note that the access rights granted to the KM distribution loader should be the minimum compatible with it doing its defined job with adequate performance.

Each entry in the queue comprises a package of key data or a key management request for distribution to a client. The approach using the interface tables mediates between the KMC and the Riposte Message Service and protects the KMC from the details of managing the Riposte traffic in a resilient and reliable fashion. The interface tables are also used for the channel to report to the KMC on the status of outgoing requests so that the KMC can delete completed requests or take remedial action for failed requests.

FUJ00098154

3.3.3 Delivery interface

When processed each outgoing request causes a named persistent object to be lodged in the Riposte Message Store accessible to the target client. The KM distribution receiver mediates between the KM Client Agent and the Riposte Message Service. It comprises a C interface which offers the following services: alerting the KM client agent when a request is received (by calling a C function that provides the entry point to the KM client agent); delivering the payload of a request to the KM client agent (by a call back from the client agent to the distribution receiver); housekeeping functions such as deletion of a named object. The distribution receiver does *not* perform automatic garbage collection; that is the responsibility of the KM client agent.

The distribution receiver offers the following resilience features:

- Notification of arrival of any request is guaranteed; however, duplicate notifications are possible in rare circumstances.
- When a failing PC is swapped-out, on start-up of the replacement PC, the KM client agent software will be alerted for all requests that have arrived since the failure (and under exceptional circumstances, for arrivals immediately preceding the failure).

On roll-out of a PC (including addition of a new node to an existing PO outlet or other multi-node client), the KM client agent will be alerted for all persistent objects corresponding to KM requests that are extant in the Riposte Message Store for the client.

3.4 Manual distribution mechanisms

Manual distribution channels are means of delivering keys in physical packages by human intervention. They are described in detail in [KMMCDES].

The KMA cannot control the manual channels directly, but will provide the key manager with information management facilities to track the progress of key material in the channels. The KMA will also alert the key manager when new key material is ready for output onto physical media and give delivery details for the media.

The uses of manual distribution channels are

(i) to deliver a confidential key to a client that does not run the KM client agent software described in section 3.5. (The architecture for this case is shown in Figure 10.)

(ii) to deliver the key encryption key TK to a client that does run the KM client agent software but is not a PO counter PC. (The architecture for this case is shown in Figure 11.)

In both the above cases, the key material requires protection that cannot be provided automatically, and so manual distribution channels must employ strong physical security. The operation of manual key channels will be defined in procedural documents.

In case (i) above, the key is in some cases handled directly by the client crypto software. In other instances of case (i) and in case (ii), software is required to load the manually delivered material into memory on the client at boot time as shown in Figure 15. The purpose of the key store booter is to read the key material from diskette and make it accessible to the KM client agent software of section 3.5. Thus, logically, the key store booter offers a subset of the functionality of the PMMC Agent described in section 3.9; physically, it reads the initial key material from diskette rather than a memory card.

FUJ00098154

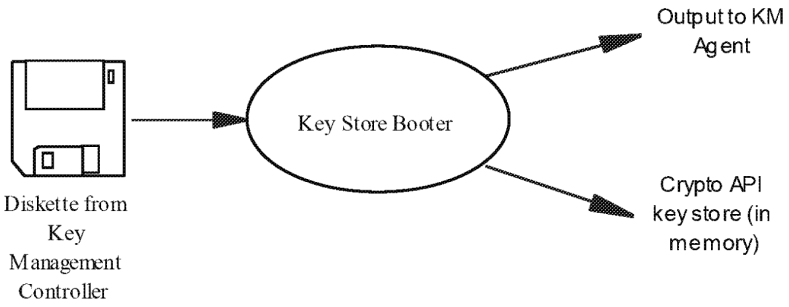


Figure 15. Manual distribution: key store booter

3.5 Key Management Client Agent

The KM client agent software is provided on each client that uses automatic distribution or monitoring. This provides the interface between the key distribution channels and the cryptographic applications that run on that client. It supports the following operations as appropriate for the cryptographic algorithms and keys deployed on the client.

Install key: (initiated indirectly by the KMC via the distribution channel): receive a new key or control request from the distribution channel and process it accordingly. Actual installation of the key may not be possible at the time when it is first delivered. Note that installing a key does not load it (see below).

Install CRL: (initiated indirectly by the KMC via the distribution channel): receive a new CRL. The authenticity of the CRL must be verified using the CA public key; if the CRL is authentic it should quickly be used to replace the CRL data structure held in the client's memory.

Load key: (initiated by the client Crypto application calling an initialisation function): place an installed key value into process memory where it can be used by cryptographic processes.

Unload key: (initiated by the client Crypto application and indirectly by the KMC via the distribution channel, see section 3.12.2): remove the key value from process memory. This may happen implicitly when the cryptographic processes using the key are unloaded; it will of course happen when the platform is shut down.

Revoke key: (initiated indirectly by the KMC via the distribution channel) removes a key from the active configuration (the reverse of installation). At NR2+ only one policy is supported: namely signatures using a revoked key do not verify (see section 3.12.4).

Destroy key: delete all persistent copies of a key from the system. Archival copies may be kept, and short-lived copies in the system swap file may persist but these are inaccessible to the "load key" process. Since the Riposte Message Store does not support a cryptographically reliable destruction of the bit pattern for the key, this operation just reclaims resources and is **not** a security feature.

The component breakdown and main data flow of KM client agent are shown in Figure 16. The handlers support the stages in the life-cycle of a key and feed into the load/unload module which produces keys for the KM client applications. The handlers read keys and CRLs from the Riposte persistent object store and may write monitoring messages to it if the audit via the NT event mechanism does not provide sufficient information for the protection domain in question.

FUJ00098154

3.5.1 Key Dispatch Agent

The components to the left of the key store in Figure 16 are event-driven. The events that drive them are: boot-up (when the PMMC agent may have reported a PMMC change) and arrival of key material or key management request via the automatic channel during normal operation.

The key dispatch agent receives the incoming events and passes them on to a handler function for subsequent processing according to the type of the event. It is not responsible for providing the various acknowledgments specified in the protocols of section 3.12. It can conveniently assist in logging arrival of key material and key management requests via the NT event log as required by section 3.6 of [KMREQ].

3.5.2 Key Load/Unload Module

The Key load/unload module provides the interface presented by KM to the crypto functions that applications call to carry out cryptographic work (encrypt/decrypt, etc.). This code uses a Key Store data structure including the Layer 7 "KeySTOR" and a data structure representing the CRL and other information. When key capsules arrive in the client and a key is to be loaded or revoked, these data structures are updated so that the applications will use up-to-date keys and an up-to-date revocation list. In addition to supporting the encrypt/decrypt/sign/verify functionality for the standard cryptosystem of section 2.7, the Key load/unload module also enables the crypto functions to service the needs of third-party applications for which KM provides distribution services (see section 3.12.6 for a discussion of the delivery protocols supported).

The Key load/unload module is responsible for implementing the confidential key selection policy defined in section 3.12.2. To this end, on a counter PC, it requires the PMMC agent to make available in memory the tags of the available TK keys (see section 3.9); on clients that use diskettes rather than PMMCs it requires the same information to be presented in the same way by the key store booter (see section 3.4).

The Key load/unload module must be able to provide a limited service when the Riposte service is not available. In particular, checking of digital signatures that contain an in-line PKC must be possible in the absence of Riposte to permit software packages to be verified and to allow the interactive channel to check messages signed with the KI key.

3.5.3 Key Install Handler

The key install handler is a component which uses the table of metadata shown in Figure 16 to identify the protection domains and other details of the incoming keys it is intended to process on any given client. In addition to handling new keys, the key installer also actions key management requests from the KMC. It is responsible for implementing the client side of protocols specified in sections 3.12.2, 3.12.3 and 3.12.6. This is a joint responsibility with the PMMC agent in the case of the protocol of section 3.12.2.

When it is time for a change to the keys on the PMMC, the key install handler receives the key management request for such a change and must notify the PMMC agent that on the next reboot new PMMC keys should be fetched. There is thus a dataflow from the key install handler to the PMMC agent. There is a dataflow in the opposite direction through which the PMMC communicates information about PMMC updates to the key install handler (effectively causing a key arrival event to be notified at boot-up for the KM client agent). See section 3.9 for more information.

The metadata shown in Figure 16 is part of the static configuration of the KM system. It is delivered as part of the KM client agent software build. It may be updated by Tivoli as part of a software upgrade

FUJ00098154

when new protection domains are introduced. The metadata does not need to be updated during normal operation or administration; it is not confidential and does not need cryptographic protection.

3.5.4 CRL Handler

The CRL handler manages deliveries of CRLs. It implements the client side of the protocol defined in section 3.12.4.

3.5.5 CAPU Check Handler

The CAPU check handler manages the periodic checking of the CA public key. It implements the client side of the protocol defined in section 3.12.5.

3.5.6 Key Destroy Handler

The Key destroy handler is a garbage collection process that is invoked by the other components of the KM client agent whenever they have taken actions that may render some key material obsolete. The Key destroy handler determines which key material may be deleted and deletes it.

3.5.7 KM Client Agent software structure

3.5.7.1 Data Flow

A data flow diagram for the KM client agent is given in Figure 16. The various handlers and the key load/unload module shown in in this diagram are concurrent processes requiring read/write access to a shared data store (the Key Store). The process structure is shown in Figure 17. Semaphores are used to protect critical sections of code requiring access to the Key Store. The shared store is locked and unlocked as a single shared resource. This simplifies identification of potential deadlocks. Finer granularity of control of access to the shared store is not required (and could introduce deadlocks).

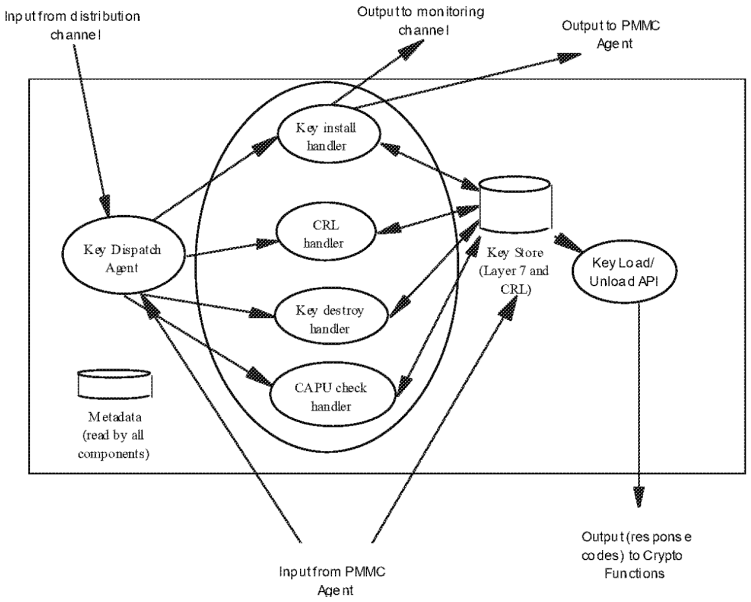
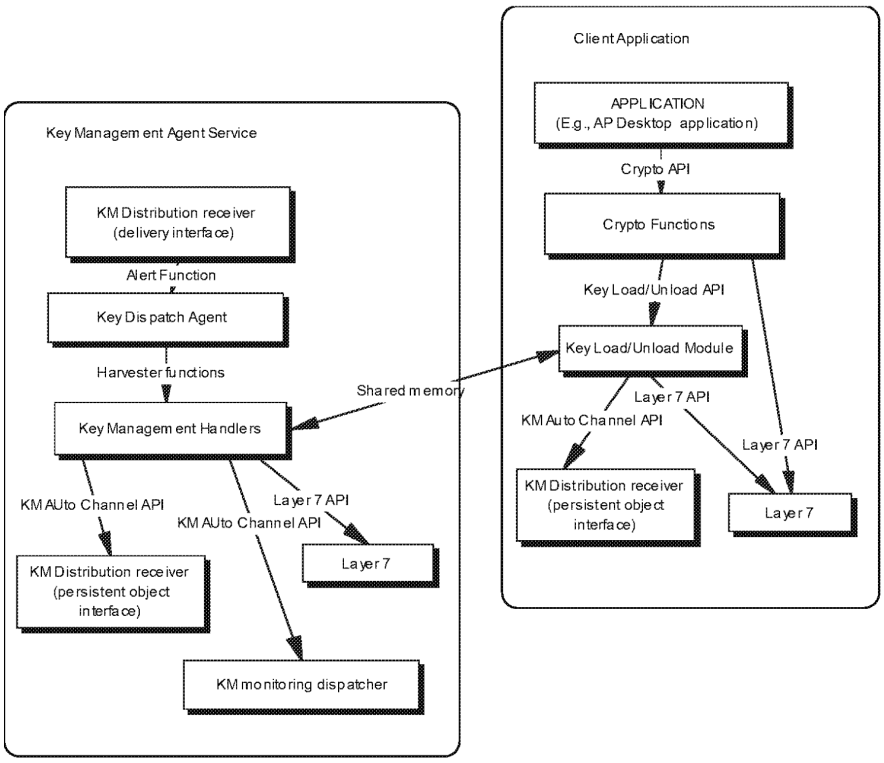


Figure 16. Key Management Client Agent data flow

FUJ00098154

1151 *3.5.7.2 Module/Process Structure*

1152 A structure diagram for the KM client agents together with the crypto functions that they support and a
1153 typical application using those functions is given in Figure 17. The modules shown in this diagram
1154 implement the data flow diagram given in Figure 16. The single-headed arrows in this table indicate
1155 invocation via implementation language procedure or function call. The double headed arrow represents
1156 communication (of new key material) via shared memory. Note that the KM Client Agent Service has no
1157 user interface - it is an NT background service that exists solely to mediate between the key delivery
1158 channel and the key store used by the crypto functions.



1159

1160 Figure 17. Key Management Client Agent Structure Diagram

1161

FUJ00098154

3.6 Automatic monitoring channel

The component breakdown and main data flows of the automatic monitoring channels are shown in Figure 18.

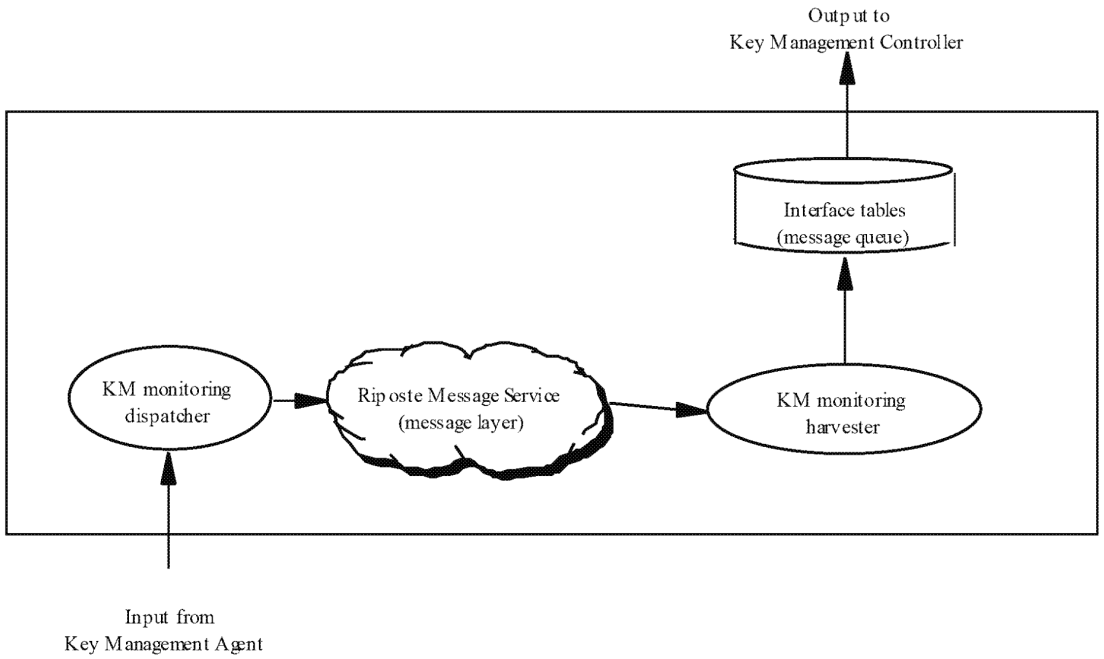


Figure 18. Automatic monitoring channel (via Riposte)

3.6.1 Dispatch Interface

The KM monitoring dispatcher is the client software component that provides the KM client agent an API for sending acknowledgments and other reports to the KMC. It forwards reports to the KMC via the message layer of the Riposte Message Service.

3.6.2 Harvesting Interface

The KM monitoring harvester is the campus component that reads messages from the Riposte distributed object store and inserts records into an interface table in the KMC. This interface table provides a queueing mechanism for incoming messages.

3.7 Manual monitoring channel

Where keys are changed manually, the manual monitoring channel comprises the procedures whereby a key custodian reports on the status of a key change to the Pathway key manager who then arranges for completion of an operation to be registered in the KMA.

FUJ00098154

3.8 Interactive Distribution Channel

The component breakdown and main data flows of the interactive distribution channel are shown in Figure 19. It comprises modules that support an (enhanced) Diffie-Hellman exchange over a TCP/IP link between the KMC and the client.

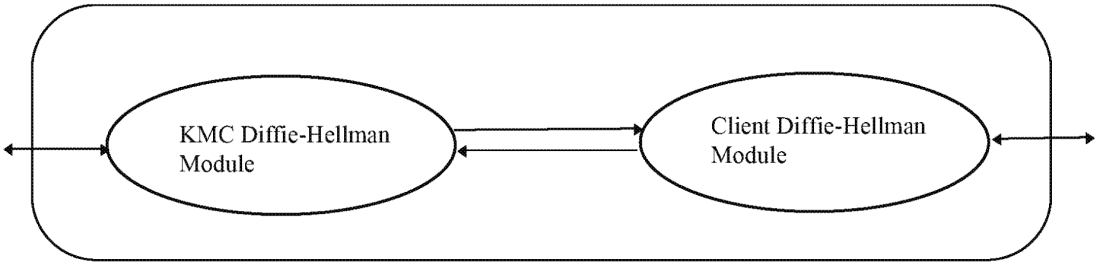


Figure 19. Interactive Distribution Channel data flows

3.9 PMMC Agent

This subsystem is to be constructed as an extension and adaptation of the release 1c/2 Post Office Logon system (PoLo).

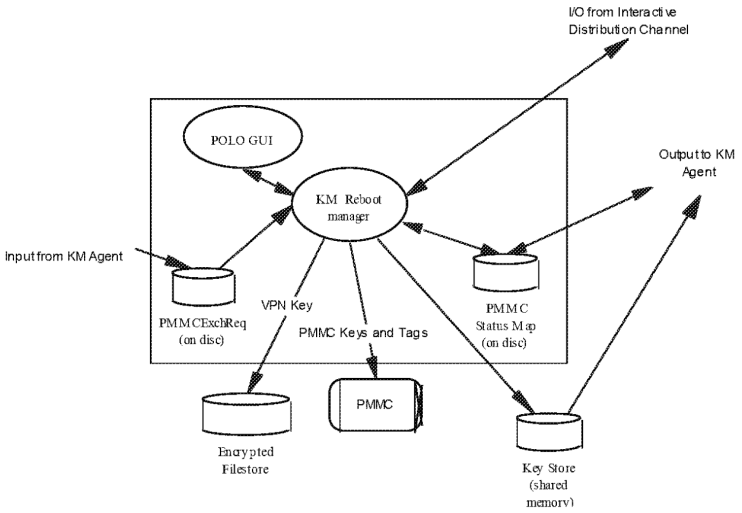


Figure 20. PMMC Agent

3.9.1 Steady State Operation

The function of the KM reboot manager is to implement the client side of the protocols defined in sections 3.10.2 and 3.12 of this document. These protocols enable the delivery of the key material that is held on the PMMC and of the VPN key that is required for the main communication route with the Pathway campuses (and so for the Riposte message service).

From the KM point of view, the overall duty cycle of a counter PC in normal operation (not roll-out or replacement) after reboot is as follows.

FUJ00098154

1. The PoLo GUI is invoked giving the POM various options depending on the state of the PC as managed by the KM reboot manager (in collaboration with the KM client agent). Once its actions are done the KM reboot manager has no further work to do other than ensure that the in-memory data structures it has set up for later use do not disappear from memory.

2. The following processes are started (in some order):
   Riposte Message Service;
   KM client agent service;
   VPN;
   Desk-top application.

3. PC does normal business (until item 4 or item 5 occurs).

4. Systems management actions take place (e.g., overnight) under Tivoli control; at this point the Riposte Message Service is stopped but verification of software packages using inline SI PKCs can take place during this process. PC reverts to normal business operation (item 3).

5. PC is rebooted; this kills the KM client agent service and the PC starts again at 1.

KM software is responsible for item 1 and provides the KM client agent service mentioned in item 2; the other items are outside the control of the KM system. Thus the KM reboot manager and the KM client agent, in effect, take turns in owning responsibility for key management activities. They communicate via NT filestore.

The KM reboot manager is invoked via the PoLo GUI when a PO counter PC is booted. It communicates with the KM client agent of section 3.5 via NT filestore. When it receives a request to carry out the interactive exchange to get new PMMC keys, the KM client agent writes to filestore information about the pending change. When the PC is next rebooted, the KM reboot manager detects the presence of this information and attempts to carry out the interactive exchange protocol described in section 3.10.2 below. This information is not confidential.

The last round of the interactive exchange requires the client to send an acknowledgment over the automatic monitoring channel. The KM agent also needs various non-confidential information about the PMMC, e.g., the tags for each key on the card (e.g., current and spare TK) so it can implement its part of the protocols defined 3.12. The KM reboot manager writes to NT filestore the information about any pending acknowledgment and about the PMMC state.

It is a design goal for the PMMC agent and the KM client agent to pass sufficient information between themselves so that the PoLo GUI can offer a convenient and helpful interface to the POM detecting and supporting resolution of problems situations such as insertion of the wrong PMMC or a blank PMMC.

The PMMC status map in Figure 20 represents the information held on disc to support this. In [PMMCADES] it is actually implemented in several files including a "trigger file" for communication with the KM client agent and an "image object identifier file" that uniquely identifies the PMMC.

3.9.2 Roll-out

At roll-out of a gateway PC, the KM reboot manager is responsible for bringing the key material on the PC into a state where the KM client agent is able to start processing keys delivered on the automatic distribution channel. The initial boot process involves the following steps:

1. The gateway contacts a boot server which delivers a small volume of initial identification data, including an initial POK and its tag (from a list supplied to the boot server under human control). The boot server is outside the VPN curtain and has its own private ISDN number with an IP address

FUJ00098154

known to the PO. Security is gained by a dial-back and by keeping this step in the process very brief (which also helps with availability).

2. The gateway then contacts the KM Controller via a special VPN recovery server (which works with a non-secret VPN recovery key). The KM Controller accepts the POK as authentication and delivers initial key material. The key material is delivered in two stages (a) the VPN key an the PMMC package including the FEK needed to protect the VPN key in filestore enabling normal campus comms for the autoconfig process and (b) a second delivery of the PMMC key package, which will have been discarded prior to the reboot that occurs when autoconfig is complete. (The PMMC package includes a new POK so that the lifetime of the POKs in the boot server's list is typically very short). After stage (b), the PMMC keys are stored on the PMMC encrypted under a PIN as described in section 3.11.

3. The gateway can now dial in and begin its autoconfig activities, which will ultimately put it in a position to start Riposte and then the KM client agent service so that the initial delivery of keys via the automatic distribution channel can begin.

The KM reboot manager is responsible for step 2 in this process, using the protocol of section 3.10.2 below for authentication and protection of the initial key material.

To prevent the supply of initial POK values in the boot server running out, the boot server remembers which initial POK values have been delivered to which PO outlets. In the event of a gateway PC being replaced, the request for an initial POK value from an outlet will receive the same value as was given at initial roll-out.

3.9.3 Recovery

If the PMMC or the PIN that encrypts the data on it is lost, the KM reboot manager uses the protocol of section 3.12.1 to ask for the lost key material to be redelivered by the KMC. The VPN key only needs to be restored in this case if the gateway PC has also failed and been replaced.

If a gateway PC fails and a replacement is swapped in but a good PMMC and PIN are available, the KM reboot manager recovers the PMMC status map from the PMMC. If the PMMC or PIN is not available, then these must be recovered as described in the previous paragraph. In either case, the VPN key must be redelivered as must information about any outstanding PMMC update request (since the files that hold these will not be available). As at roll-out, the initial contact with the KM Controller is via the VPN recovery server using the non-secret VPN recovery key.

The gateway failure may occur after the PMMC has been updated but before the acknowledgment of that update via the automatic monitoring channel has been received. If there is an outstanding PMMC update request the PMMC agent should check the PMMC and if it has been updated then it should indicate that in the PMMC status map. If the PMMC has not been updated, then the POM should decide whether or not to update the PMMC on this reboot or on a subsequent one.

Just as in normal operation it is a design goal for the PMMC agent and the KM client agent to cooperate so as to offer a convenient and helpful interface to the POM detecting and supporting resolution of problems situations such as insertion of the wrong PMMC.

3.9.4 PoLo GUI

The PoLo GUI is to be implemented by adapting the NR2 PoLo GUI. In addition to reducing development costs, this will help to ensure that the NR2+ interface as seen by the POM is uniform with the NR2 interface.

FUJ00098154

At NR2+, the question arises as to whether or not the POM should be given the option to defer updates to the PMMC or to defer re-encryption of the counter filestore after the FEK has changed. For operational reasons, it is important to allow the POM to defer the re-encryption of filestore (since this may take some time). Both for operational reasons, and to simplify the design of the KMA, the chosen approach is to allow the POM one option: whether or not to embark on a PMMC update at all. When presenting this option to the POM, the GUI should make clear what the consequences and pre-requisites are (i.e., if FEK is changing then the whole process may take several hours, if the PMMC is to be updated at all, then the receipt printer needs to be working and the POM needs to be in a position to handle the PMMC and PIN according to POCL's procedures (e.g., time-locked safe storage may need to be available). If the POM continues not to co-operate, then the Pathway Key Manager will eventually be notified and will follow an appropriate procedure to ask for co-operation. Pathway's SLA's for key changes are not affected, since the fault lies with POCL rather than Pathway.

3.10 Multinode Clients

Some clients, e.g., the PA signing servers, are themselves distributed systems. Management of keys on such clients will typically require some form of synchronisation within the client or between the client and the KM Controller, e.g., so that the KM Controller can know when all nodes have completed a key change. The requirement will generally depend on the architecture and business function of the client in question. Two coordination/synchronisation issues have been identified in the present design; these relate to the PA protection domain and to PO outlets. These issues are resolved in sections 3.10.1 and 3.10.2 below.

3.10.1 PAPR Synchronisation

It is necessary for each Agent Server to use the same DSA PQG values as are in use at the Vector Server that supports it. Experimentation and discussion with Sapher Servers reveals that while Layer 7 uses the key transport file containing the DSA private key to provide the PQG values to the vector generation, the actual key value in the key file is irrelevant - only the PQG values affect the computation (as one expects from the maths behind DSA). As PQG values do not need to be changed, the PQG values can be delivered to the vector servers in a key transport file containing a DSA private key that is not used elsewhere. The template used to carry the PQG values will be delivered to the KMA for use in subsequent public/private key pair generation. Thus the "synchronisation" problem that the release 2 procedures catered for can be solved as part of the static configuration in NR2+ and later.

3.10.2 PO Synchronisation

This synchronisation problem is to coordinate the KMAs model of the state of the platforms and of the keys held on the PMMC with the reality in a PO outlet.

To solve this problem the KM must have some model of the platform inventory at each PO. Obtaining an accurate picture from external sources is believed to be problematic, and so the KM will maintain its own records. After the first dispatch of key material to a gateway PC being rolled-out, the KMC will expect acknowledgments of receipt of the keys from any number of nodes within the PO outlet containing that gateway. Once the KMC has received acknowledgment from a node it will register that node as belonging to the outlet and will expect in future to have to manage that node. Subsequent key changes will not be considered to be complete until all registered nodes have been acknowledged. When a node is taken out of service for a prolonged period, its registration can be cancelled by manual intervention at the KMA. If a definitive automatic feed of information about changes in the PO population comes available, the KMA could be enhanced to reduce the administrative burden.

FUJ00098154

The method described above allows KMA to determine the platform inventory in each PO outlet. When a PMMC key change occurs, it is the responsibility of the POM to install and distribute the new keys by rebooting the gateway PC to update the PMMC and then rebooting the non-gateways PCs using the updated PMMC. KMA must track this process so that the Pathway Key Manager can be notified if it is not carried out in a timely fashion. The KMA's model of the process is necessarily an approximation, but the KM system as a whole must ensure that this model is eventually synchronised with reality at the PO outlets. The protocol used to achieve this is shown in the state transition diagrams given in Figure 21.

Figure 21 depicts a view of the KMA model and of the reality of the situation at a single outlet. There are several parties involved in the protocol that coordinates the model and the reality: the KMA, the gateway PC, N registered non-gateway PCs and the POM. The KMA's model for each outlet can be viewed logically as compromising a register of the N non-gateway PCs at the outlet together with 2+N separate state machines representing the state of: the overall outlet, the gateway PC and the N non-gateways. At the outlet, there are 1+N state machines: the gateway PC and the N non-gateways together with the POM who is modelled in the figure as another state transition system.

Each of the state transitions in Figure 21 is associated with an event corresponding to an input or an output of the party undergoing the state transition; the events names are tagged with "In" or "Out" accordingly. The events are also tagged with "PO" representing the FAD code or OUC of the outlet, and where relevant, with a node number: "1" for the gateway and "i" for a non-gateway ($2 \leq i \leq N+1$). The events identified are described in the following table.

Name	Description
PMMCKeyChangeReq	The message sent by the KMA to instigate the PMMC key change via the key install handler (not the reminder to the POM); when this is sent out the three state machines in the KMA model move out of their steady state (in lock-step, as part of the same transaction that sends the message).
PMMCKeyChangeAck	The acknowledgment of receipt by a counter PC of the message corresponding to PMMCKeyChangeReq.
PMMCKeyChangePrompt	The prompt message sent by the KMA to the POM to ask him to reboot the gateway and then the other PCs
TimeOut	A time-out event causing a prompt message to be resent if the POM has not cooperated. (After a while, the Pathway Key Manager will also be alerted but this is not shown here.)
IntExchAck	The acknowledgment sent out by the gateway PC at the end of the interactive exchange described in section 3.12.1. (The detailed state changes of the exchange itself are not shown here.) This acknowledgment is sent both by TCP/IP and via Riposte for resilience.
NewPMMCAck	The acknowledgment sent out by all counter PCs when they have processed the updated PMMC. This is not sent out until the filestore has been re-encrypted if the FEK has changed.
Reboot	The POM reboots a PC (logically, an output by the POM and an input to the PC).
ManualDeletion	Removal of a counter PC from the inventory of PCs registered at an outlet.

The protocol as defined by the diagrams therefore operates as follows. Note that the ordering of these steps is not necessarily as given. Any ordering compatible with the concurrent operation of the state machines in the diagram is possible and the steps may even be interleaved.

FUJ00098154

1. All state machines are in the steady state (bootstrap is not considered here).
2. When a PMMC key change is necessary, the KMA causes the event PO.PMMCKeyChangeReq to occur for the PO outlet in question (and its state machines move out of the steady state).
3. The gateway acknowledges receipt of the PMMCKeyChangeReq message.
4. The KMA sends out its PMMCKeyChangePrompt to the POM
5. The POM reboots the gateway causing the interactive exchange to take place and the IntExchAck event to occur.
6. The gateway re-encrypts its filestore if necessary and causes the 1.NewPMMCAck event to occur.
7. The KMA model of the gateway goes back to the steady state
8. The POM reboots the non-gateways in some order causing N i.NewPMMCAck events to occur
9. If any of the i.NewPMMCAck events refers to a counter i that has not yet been registered the new counter is registered (added to the inventory).
10. The non-gateway models revert to the steady state as the i.NewPMMCAck events occur.
11. When all the i.NewPMMCAck events have occurred the overall PO model reverts to the steady state.

The above relies on the co-operation of the POM. If the steady state is not reached in a timely fashion additional prompts may need to be sent out. For simplicity these are not shown on the diagram. The wording of these prompts should be adjusted to suit the situation according to its model of the PO state. The POM may have rebooted the gateway for some other reason prior to reading the initial prompt; both the wording of the initial prompt and the relevant PoLo dialogues should be worded to cater for this possibility.

FUJ00098154

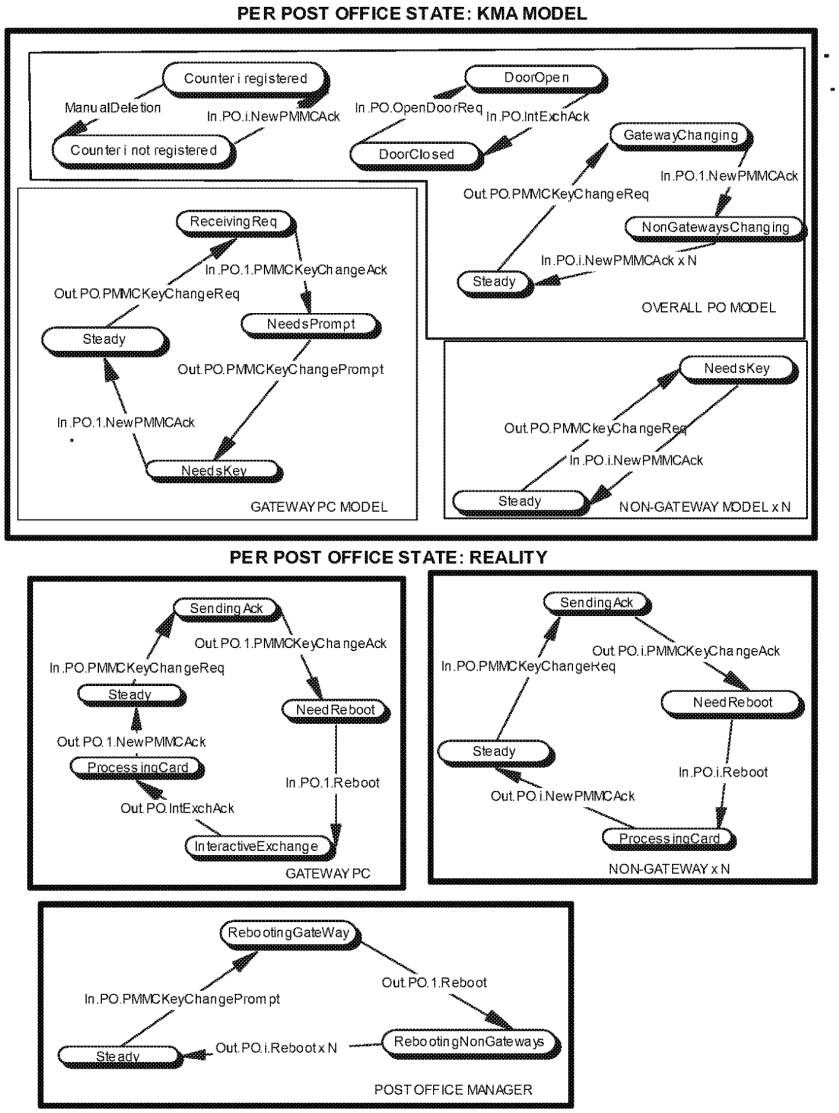


Figure 21. PO Synchronisation State Transitions

3.11 Key Storage

A complete list of the key material managed by the KM Service may be found in section 4. Keys that are distributed by electronic means fall into several general categories applicable to any client running NT and Riposte. The categories and the policy for storing the key material in each category are shown in the following table. Key material stored in Riposte is stored using the Riposte persistent object layer. To illustrate the categorisation, the table also shows examples of the keys in each category. In the table, "third-party crypto material" refers to key material required for out-sourced cryptographic products such as Utimaco VPN for which the KM Service provides key delivery services.

FUJ00098154

Where key material is stored encrypted, the storage format must permit an integrity check to allow early detection of accidental or deliberate corruption of the cipher-text. When Layer 7 formats are used, the relevant Layer 7 functions to be used to enable checking, and the receiving software should use the appropriate Layer 7 function to carry out the integrity check. When raw bit patterns are used, 4 check bytes comprising the first 4 bytes of SHA of the data must be appended to the data prior to encryption (at the KMA) and the receiving party should decrypt the data and check that the last 4 bytes are as expected. Red Pike encryption of any item longer than 8 bytes should be done using CBC mode.

Category	Examples	Algorithm	Storage	Format	Packaging
Key Encryption Key	TK	Red Pike	PMMC/diskette	Raw bit pattern + layer 7 key tag encrypted under PIN.	None
Filestore Encryption Key	FEK	Red Pike	PMMC	Raw bit pattern + layer 7 key tag encrypted under PIN.	None
Authentication key	POK	Red Pike	PMMC	Raw bit pattern + layer 7 key tag encrypted under PIN.	None
Confidential Key	APPR , GDK	DSA/Red Pike	Riposte (per client)	Layer 7 transport (using TK as KEK)	See section 3.1.3
Public Keys	APPU, FTPPU	DSA	Riposte (global)	Layer 7 transport (no KEK)	mini X.509 PKC See section 3.1.1
CRL	CRL	n/a	Riposte / NT Filestore	Raw bit pattern	See section 3.1.2
Certification Authority	CAPU	DSA	NT Filestore	Layer 7 transport (no KEK)	None
Third-party crypto material	VPN	application-defined	application-dependent	application-dependent	application-dependent
Transient	CK	Red Pike	RAM only	Layer 7 C data structure	none

Keys on non-NT platforms are stored using appropriate facilities of the operating system in question.

The PIN mentioned in the table is only used in PO counter PCs at NR2+. It is generated automatically by the PMMC agent using local entropy generated in software by Layer 7 facilities. It is stored in a printed

FUJ00098154

record held by the POM separately from the PMMC and is not managed by the KM system. A new PIN is generated each time the card is updated.

3.12 Key Transfer Protocols

For most of the key categories identified in section 3.11 the KM service defines protocols to be used to transfer the key material to a client. The protocols must preserve working cryptographic relationships while delivering keys in a timely fashion. The PO synchronisation protocol of section 3.10.2 is one protocol of this sort. In this section, we discuss the protocols that are used to manage the various categories of keys identified in section 3.11. The protocols are described in sections 3.12.1 to 3.12.6 below.

Both the KMC and the KM Client Agent software at a client know from their metadata the list of protection domains that are supported on the client and hence know the total inventory of keys that the client needs.

The design of the various distribution channels is such that some of the events that steer the protocols may occur at unexpected points in the protocol. Multiple notification of an event is expected to be rare, but is not prohibited by the design of the automatic distribution and monitoring channels. Except where otherwise stated, the functional requirement is that both the KMC (resp. a client) should ignore a small number of unsolicited events from a client (resp. the KMC). Unsolicited events should always be logged via the NT event log, and if a large number of unsolicited events occur, the recipient should raise a security alert.

A study into the overall resilience properties of the NR2+ KM system as defined in this document is planned prior to the second approved issue of this document. The protocols defined in section 3.10.2 above and in sections 3.12.1 to 3.12.6. below have deliberately been kept simple to facilitate this resilience analysis. Where that analysis suggests refinements to the protocols, they will be incorporated in this document. It is envisaged that these refinements will themselves be quite simple, typically manual intervention to restore a client to a known state followed by manual administration of the KMA database to make its model of the client state reflect that known state. The resilience analysis will include an indicator of the likely frequency (and hence of the lifecycle costs) of such recovery processes.

The state transition diagrams used to describe the protocols in section 3.10.2 above and in sections 3.12.1 to 3.12.6 below follow the conventions discussed in section 3.10.2. The diagrams show the KMA model and the client reality in terms of the inputs and outputs of the KMA and the client. Each diagram should be thought of as being instantiated for each delivery of a particular item of key material to a particular client (single or multi-node) via the protocol in question. Distinct "run"s of a protocol for a particular item to a particular client do not overlap.

For keys held on PO outlets, the mapping of protocols to the various categories of keys given in section 3.11 above is shown in the following table.

Category	Transfer Protocol	Document Section
Key Encryption Key	Interactive Exchange Protocol	3.12.1
Filestore Encryption Key	Interactive Exchange Protocol	3.12.1
Authentication key	Interactive Exchange Protocol (initial value delivered from boot server)	3.12.1 (see also 3.9 and 4.5.3)
Confidential Key	Confidential Key Protocol	3.12.2
Public Keys	Public Key Protocol	3.12.3

FUJ00098154

CRL	CRL Protocol	3.12.4
Certification Authority	CAPU Check Protocol	3.12.5
Third-party crypto material	application-dependent	3.12.6
Transient	N/A (used in interactive exchange protocol and then discarded)	3.12.1

3.12.1 Interactive Exchange Protocol

This key transfer protocol is required on clients using confidential keys protected under a per-client key encryption key and manufactured at the client site rather than centrally. At NR2+, this applies to PO outlets only, but the method could be used for other clients in future releases of the KM system. An authentication key *AK* is used to authenticate the client to the KM Controller. For a PO outlet, this authentication key is either POK (normal usage) or a 64-bit digest of SHA(POK) (recovery). The key transfer may be used to deliver a new KM Traffic Key (TK) to the client or other secret material for storage on the client's PMMC or other removable token. The protocol requires the cooperation of the local key handler for the client (the POM at a PO outlet) to reboot the client and to supply the PMMC or other token. For each client, the KM Controller is only prepared to accept the protocol in particular time intervals determined by the roll-out programme, the routine key change programme, and recovery or hardware replacement operations authorised by the Pathway help desk.

The protocol is initiated under several circumstances including roll-out of new or replacement gateway; recovery from lost PIN or PMMC; routine key update. For the first two cases, there is no need for the KM Controller to prompt the key handler; in the case of a routine key update, the KM Controller will notify the gateway PC to go through the exchange on its next reboot as described in section 3.10.2. In all cases, the KM Controller is aware that the client needs to action this protocol and has "opened a door" allowing the client to participate in this protocol.

The protocol proceeds as follows once the key handler has rebooted the client and gone through the GUI to select this key transfer:

1. Client software selects appropriate *AK* (POK or 64-bit digest of SHA(POK)) and sends the KM controller the following data:

   [Req-Type, X, DT, Client Name]*AK, POK-Tag*

   (where:
   X = $\mathbf{g}^x$ mod n is a Diffie-Hellman public value for a fresh random value x;
   DT is a date-and-time-stamp;
   Req-type identifies required key material and reason for request; recovery requests may be distinguished using this value Request types and all data formats associated with the interactive exchange are defined in in [PMMCADES].
   POK-Tag is the Layer 7 key tag for the POK.)

2. KMC authenticates data received from client using **Client Name** and POK (if not recovery request) or 64-bit digest of SHA(POK) (if recovery request); if authentication fails, this is an attack or an attempt by a POM to recover without following the proper procedures. KMC sends client:

   {Y, DT, Client Name}KIPR, KICERT, (*Keys*)CK

   (where:
   Y = $\mathbf{g}^y$ mod n is a Diffie-Hellman public value for a fresh random value y;
   DT is the DT value received from step 1;
   KIPR is the key issue private signing key;

FUJ00098154

KICERT is a PKC containing the KI public key signed by the CA private key;
CK is a 64-bit digest of the Diffie-Hellman shared secret $\mathbf{X^y (= (g^x)^y = g^{xy} = (g^y)^x = Y^x)}$;
Keys is the key material required by the client as determined by the Req-Type from step 1; ***Keys*** may include **TK** and its tag **TK-id**). KMC now erases CK from memory.

3. Client verifies KICERT using appropriate CAPU and then authenticates KMC using KICERT (if this fails, this is an attack). Client recovers ***Keys*** and writes them to their target storage (the PMMC or other token (where it also stores the old TK value for resilience purposes if a new TK has been delivered) and/or encrypted NT filestore). The client sends an acknowledgment along the interactive channel to the KMC, although little harm is done if this acknowledgment fails to get through. Once the automatic monitoring channel is available the client sends an acknowledgment via that means as well. Neither acknowledgment should be sent if the ***Keys*** have not been successfully transferred to the target storage. The client now erases CK from memory.

4. KMC receives either the acknowledgment sent through the interactive channel or the one sent through the automatic monitoring channel and "closes the door" on this client. If the interactive distribution channel fails to deliver the acknowledgment then the door will just be open for slightly longer than necessary.

There is no persistent client-side state associated with the interactive exchange; once it arrives in the NeedReboot state of Figure 21; the client will just attempt the exchange every time it is rebooted and the user elects to update the PMMC. The state transition diagram of Figure 22 shows the opening and closing of the KMA's doorways for the interactive channel. The event OpenDoorReq is generated by the help desk or at roll-out and its generation is not shown in the state transition diagrams in this document. The diagram involves the event PO.IntExchAck as discussed in section 3.10.2.

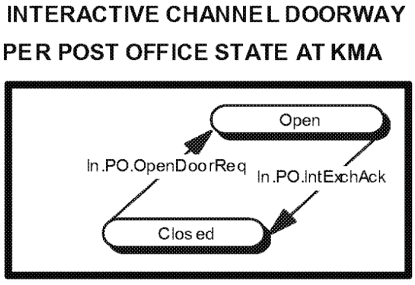


Figure 22. Interactive Channel Doorway

3.12.2 Confidential Key Protocol

This protocol is used to install a new DSA private key or Red Pike key on a client. In broad outline, the protocol operates by the KM Controller lodging a key capsule encrypted under the key protection key TK into the Riposte Message Store accessible to the client. The client acknowledges receipt of the key capsule and arranges to use it for future signing or encryption operations.

The protocol is steered by the serial numbers of the confidential keys and of the key encryption key TK. In the description below we write $ConfK_m$ for confidential key *ConfK* at serial number *m* and TK_n for TK at serial number *n*. We assume that the data store into which confidential key packages are delivered allows easy searching for all held $ConfK_m$ values and that the client loads and retains in memory available TK values from the PMMC or other token at boot-time - there will be at most two of these say TK_n and $TK_{(n-1)}$. Let us assume that the client is currently using $ConfK_m$ (i.e., the current serial number for this confidential key is *m*).

FUJ00098154

The protocol has two alternatives (determined at the client according to the availability of an appropriate value of the key encryption key TK):

Change Confidential key using existing TK:

1. KM Controller sends client $(ConfK_{(m+1)})TK_n$.
2. Client notes it can decrypt this and so acknowledges installation of the new key and triggers unloading of $ConfK_m$ (which will make a future load for *ConfK* pick up the new one). At this point it can invoke the key install handler (garbage collector) to remove unwanted key capsules.

Change Confidential key using new TK:

1. KM Controller sends client $(ConfK_{(m+1)})TK_{(n+1)}$.
2. Client notes it cannot decrypt this and does nothing except acknowledge receipt of the key. The key will be considered to be installed when $TK_{(n+1)}$ arrives.
3. KM Controller sends $TK_{(n+1)}$ using the protocol of section 3.12.1.

Note in the first alternative, only installation is acknowledged; in the second, only receipt is acknowledged; in the second case, the KMA can infer that installation has taken place when the new TK value is installed.

State transition diagrams showing the KMA model and client reality for this protocol are given in Figure 23. The diagrams involve the event PO.IntExchAck described in section 3.10.2. For assessing the resilience of this design, it should be noted that this event is passed from the PMMC agent to the KM client agent via NT filestore. The state transition diagrams also use the following events specific to this protocol:

Name	Description
ConfK	This event is associated with dispatch or receipt of the confidential key capsule. It has an attribute ConfK.TK(Y) giving the key tag of the TK used to encrypt it.
Ack.Installed.ConfK	The acknowledgment of installation of the confidential key.
Ack.Received.ConfK	The acknowledgment of receipt of the confidential key.

Confidential key selection policy: corresponding to this protocol, the key load/unload module must adopt the following policy: given a choice between several *ConfK* and TK serial numbers it should use the highest *ConfK* serial number that it is able to decrypt.

FUJ00098154

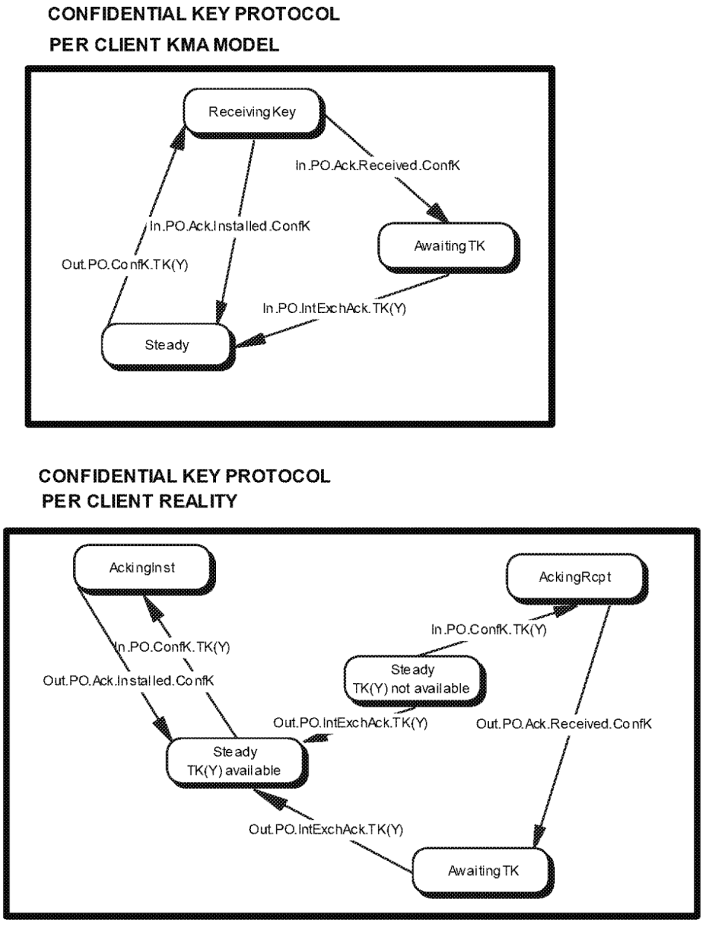


Figure 23. Confidential Key Protocol State Transitions

3.12.3 Public Key Protocol

This key transfer protocol is required on any client that does DSA verification. The protocol makes a new PKC available to the client. The delivery protocol is as follows:

1. KMC delivers PKC to client into Riposte Message Store via the automatic distribution channel
2. The conventions for loading a key from a PKC by key-tag ensure that the new PKC will be found when needed and so the installation is automatic and the client immediately acknowledges receipt and installation of the new PKC.

As a general principle, the KMA must ensure that the parties who sign with the private key do not begin to use it until all the parties who verify with the corresponding public key have received it. However, the Pathway Key Manager may wish to override this policy in some cases where the risk of not updating the key is considered to outweigh the potential lost business. Thus this protocol interacts with the confidential key protocol described above. The state transition diagrams are shown in Figure 24. In these diagrams, there are assumed to be N verifying parties, whose names are represented by "VerifierId" and M signing parties, whose names are represented by "SignerId". The events in the diagram are as follows:

FUJ00098154

Name	Description
PKC	This event is associated with dispatch or receipt of the PKC. It is qualified by the name "VerifierId" of the recipient.
Ack.PKC	The acknowledgment of installation of the PKC.
ManualOverride	This event occurs when the Pathway Key Manager elects to override the usual protocol and send out the private keys without waiting for all the acknowledgments.
ConfK	This event is associated with dispatch of the private key corresponding PKC. It is qualified by the name "SignerId" of the recipient. This event will initiate the confidential key protocol as defined in section 3.12.2.

The issuing of the private key in this protocol is not intended to be implemented as automatically occuring when all the associated PKCs have been installed. In the case of a spare, for example, the private key should not be issued until needed. The state transition diagrams are showing the minimum level of co-ordination that is needed and suppress controls that are internal to the KM Controller.

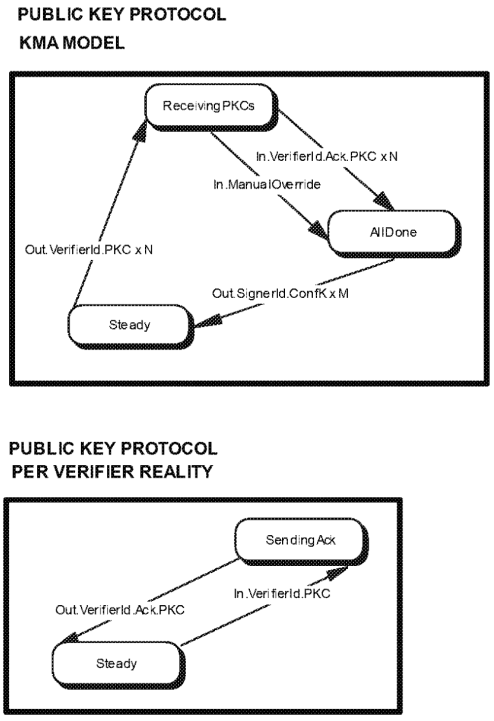


Figure 24. Public Key Protocol State Transitions

3.12.4 CRL Protocol

This key transfer protocol is required on any client that does DSA verification. The protocol makes the new CRL available to the client. The client copies the CRL to NT filestore and acknowledges receipt. For performance reasons in the event of a large-scale compromise, it is undesirable for the PO outlets to

FUJ00098154

receive a CRL potentially containing many thousands of entries that are not relevant to them. For uniformity in the implementation, optimised CRLs are constructed for various types of client containing entries that relate to protection domains with which those clients do signature verification.

1. KMC calculates CRL entries for client and delivers the resulting list to client into Riposte Message Store via the automatic distribution channel
2. The client writes CRL data into NT filestore for future use, loads CRL into memory and acknowledges installation.

Two modes of use of the received CRL are envisaged:

Hard revocation: in this usage, an attempt to verify an item signed with a key whose tag appears in the CRL is simply failed.

Soft revocation: in this usage, an attempt to verify an item signed with a key whose tag appears in the CRL results in a return value that the calling application may use to decide on the basis of the reason for and date of compromise of the key whether or not to accept the item.

Early releases of KM will only support hard revocation.

The state transition diagrams for this protocol are shown in Figure 25. For the purposes of assessing the resilience of this protocol, it should be noted that these diagrams do not highlight the use of NT filestore to hold the CRL. The events in the diagram are as follows:

Name	Description
CRL	This event is associated with dispatch or receipt of the CRL. It is qualified by the name "Client" of the recipient.
Ack.CRL	The acknowledgment of installation of the CRL.

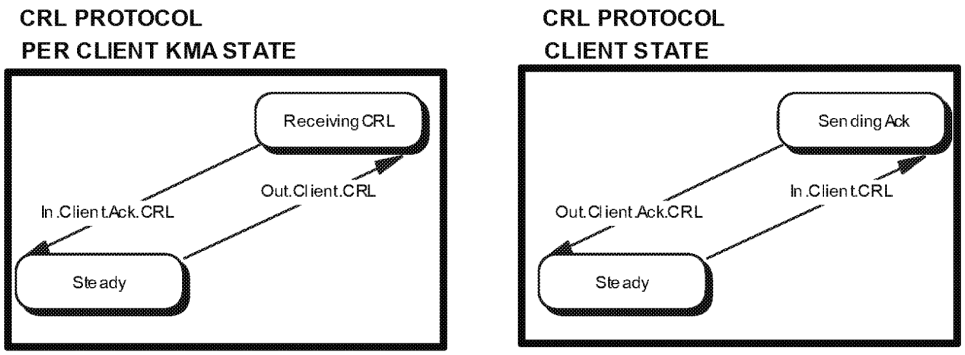


Figure 25. CRL Protocol State Transitions

3.12.5 CAPU Check Protocol

This key transfer protocol is required on any NT client that receives key material from the KM controller. The KM Controller delivers to the Pathway manufacturing unit a life-time supply of CA public keys and these are manufactured into the filestore of all relevant clients.

At configurable intervals CAPU check protocol is run for each client to increase confidence that its CAPU keys have not been tampered with. The protocol operates as follows:

FUJ00098154

1. KM Controller broadcasts to each client a request containing a digest of the life-time stock of CAPU keys (this includes all keys, even revoked ones). The digest is signed with the KI private key.
2. On receipt of the request, the client checks the digest supplied in the request against the CAPU keys held in its filestore. Each pair of transport files should be identical (both the raw key and the key tag must agree).
3. If the check has revealed a mismatch, the client raises an alert via the NT event mechanism, then:
   The client sends an appropriate acknowledgment to the KM controller and stops, then:
   The Pathway Key Manager instigates identification and resolution of the problem and gives a manual confirmation to the KMC when the problem has been resolved. The Pathway Key Manager can test that a CAPU check problem has been resolved by repeating the CAPU check for the failing client.

State transition diagrams for this protocol are shown in Figure 26. The diagrams involve the following events.

Name	Description
CAPU	This event is associated with dispatch or receipt of the CAPU check material. It is qualified by the name "Client" of the recipient.
AckOK.CAPU	The acknowledgment that the CAPU check has been passed.
AckNotOK.CAPU	The acknowledgment that the CAPU check has been failed.

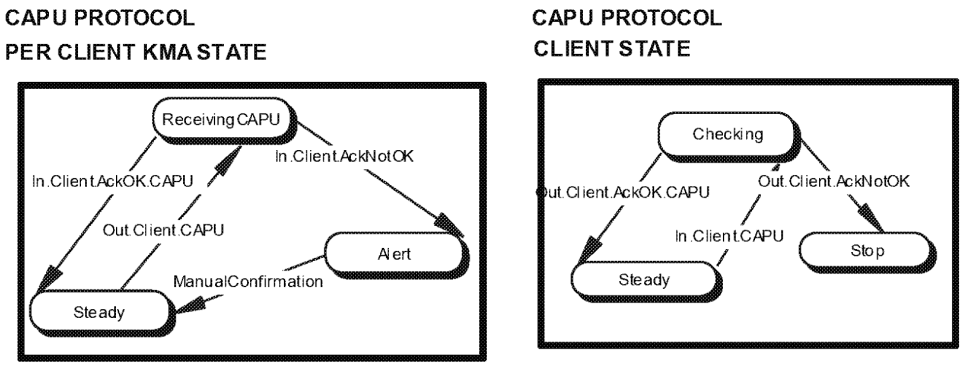


Figure 26. CAPU Protocol State Transitions

3.12.6 Third-party crypto material

The storage of third-party keys is dependent on the application. At NR2+, the third-party keys shown in the following table have been identified. The goal of the current design is to provide simple but general mechanisms that support these keys and have the potential to support other third-party key material in later releases.

Key	Storage	Algorithm	Format	Packaging
VPN	TeamWARE Crypto Encrypted NT file	RSA	application-defined	See below

FUJ00098154

DLLKA	PMMC	application-defined	Raw bit pattern + key tag encrypted under PIN	See section 3.12.1
DLLKB	Riposte (per client)	application-defined	application-defined	none

The VPN key may be transmitted via either of two routes: under normal operation via the automatic channel and in exceptional circumstances via the interactive channel. The exceptional circumstances are when the counter PC cannot access an existing VPN key, so that Riposte communications are not available (e.g., roll-out, swap-out, lost PMMC/PIN). When sent via the automatic channel, the VPN key material is delivered protected under TK using a variant of the confidential key protocol of section 3.12.2. The difference in the protocol is that the payload of the VPN confidential key capsule is the VPN key material, which must be copied into NT filestore. When no existing VPN key is available,, the VPN key material is delivered as part of the payload (**Keys**) of the interactive exchange protocol of section 3.12.1.

Third-party key material held on the PMMC is transferred as part of the payload (***Keys***) of the protocol described in section 3.12.1 above and so no separate delivery protocol is required.

Third-party key material held in Riposte encrypted under TK (either in the Layer 7 format or an application defined format) may be delivered and managed using the protocols and selection policy of section 3.12.2. (There is no example of this at NR2+.)

In some cases, third-party key material needs to be delivered in part by the interactive channel and in part by the automatic channel (e.g., DLLKA and DLLKB). In this case, there is a dependency between the two parts. Fortunately, in current and anticipated examples, the dependency is that a confidential key capsule depends on an item on the PMMC. Substituting DLLKA for TK, the protocol of section 3.12.2 will manage this dependency automatically.

Third-party crypto material with more complex installation requirements than those discussed above are not further considered in the NR2+ design.

3.13 Interface Specifications

In general, the detailed specification of the main system interfaces will appear in the design documents for the component that generates the data on the interface. The following lists the interfaces that are to be defined and gives references for the detailed specifications.

1. KM controller-Automatic distribution channel: the control logic and data format of the interface table in the left half of Figure 8. See "KM Automatic Channel Detailed Design" [KMACDES]
2. Automatic distribution channel-KM client agent: the APIs that the KM distribution receiver and the KM client agent offer one another to implement the data flow at the bottom left of Figure 8. See "KM Automatic Channel Detailed Design" [KMACDES]
3. KM client agent-Automatic monitoring channel: the control logic and input format for the KM monitoring dispatcher in Figure 8. See "KM Automatic Channel Detailed Design" [KMACDES].
4. Automatic monitoring channel-KM application: the control logic and data format for the interface table in the right half of Figure 8. See "KM Automatic Channel Detailed Design" [KMACDES]

FUJ00098154

5. KM client agent-crypto application: the API presented to the crypto library as shown in Figure 8. See "Key Management Client Agent Design" [KMCAGDES].
6. PO configuration data-KM controller: the control logic and data format of the feed of PO configuration data as shown in Figure 8. See "KMA Design" [KMAPDES].
7. Help Desk-KM controller: the control logic and data format of the feed of recovery requests from the Help Desk as shown in Figure 9. See "KMA Design" [KMAPDES].
8. KM controller-interactive distribution channel: the API to access the KMA used by the KMC Diffie-Hellman module. See "Detailed Design of KM Interactive Channel" [KMICDES].
9. Interactive distribution channel-KM client agent: the API that the Client Diffie-Hellman module offers the KM client agent. See "Detailed Design of KM Interactive Channel" [KMICDES].
10. KM client agent-PMMC agent: the control logic and data format of the NT files used to communicate information about PMMC key change requests as shown in Figure 16 and Figure 20. See [PMMCADES].
11. PMMC agent-KM client agent: the control logic and data format of the NT files and in-memory data structures used to communicate information about the PMMC as shown in Figure 16 and Figure 20. See [PMMCADES].
12. Key store booter-KM client agent: the control logic and data format of the NT files and in-memory data structures used to communicate information about TK. See Figure 16 and Figure 15. See [PMMCADES].
13. KM controller-PMMC agent: the end-to-end interface for the data passed over the interactive channel. See [PMMCADES].
14. KM controller-KM client agent: the end-to-end interface for the data passed over the automatic channel. See [KMCAGDES].

Other internal interfaces will be defined in lower level design documents.

FUJ00098154

3.14 Component Summary

The following table lists alphabetically the software components described in section 3 of this document together with a subsection reference for further information on that component.

CAPU check handler	3.5
Certification authority	3.2.4
Client D-H Module	3.8
CRL handler	3.5
Help Desk GUI	3.2
Help Desk Processor	3.2
Key destroy handler	3.5
Key dispatch agent	3.5
Key generators	3.2.3
Key install handler	3.5
Key load/unload module	3.5
Key store booter	3.4
KM application	3.2.2
KM distribution loader	3.3
KM distribution receiver	3.3
KM monitoring dispatcher	3.6
KM monitoring handler	3.6
KM reboot manager	3.9
KMC D-H Module	3.8
PoLo GUI	3.9

FUJ00098154

3.15 Volumetrics

The basic volumetric requirements on the system may be derived from the following information:

1. Size of the representation of each item of key material (may depend on location, e.g., Riposte attribute grammar adds an overhead on top of the Layer 7 key transport file format).
2. Keys required at each platform
3. Required rate of routine key change for each key
4. Expected rate of emergency key change for each key and platform (i.e., recovery from compromise or PMMC or PIN loss).
5. Maximum expected latency period for public keys.

Performance requirements on the design are given in section 5.1 below. The following tables give space budgets for the protocol messages (not including Riposte overheads) and required overall throughput for implementing the protocols identified in 3.12 for the NR2+ keys and clients as identified in section 4. The figures are given for routine key changes in the steady state and then for a disaster recovery situation (delivery of a new confidential key, all public key certificates and a CRL to all outlets). During roll-out the key changes for counter PCs amount to approximately 1.5 times the figures for steady-state key changes. During NR2-NR2+ migration the key changes for counter PCs amount to approximately 3.5 times the figures for steady-state key changes.

Some PKCs (at NR2+, only the PAPU PKC) are sent out to all counters at the same time prior to a change corresponding private keys (at NR2+, PAPR). The table PEAK EVENT TRAFFIC below gives the volumes for NR2+.

In the event of a major key compromise or some other major problem,, the KMS might be used to assist in a disaster recovery exercise to deliver key material to the whole Pathway estate in a short period of time. As a worst case for performance estimating purposes, the table headed DISASTER RECOVERY below shows the volume of traffic required to deliver a complete new set of automatic channel key material to the PO outlets.

ASSUMPTIONS

PA compromises per annum	1
CA stock	10
Average no. of non G/W PCs per PO outlet	1
Expected CAPU attacks/errors	1.E-04
Number of outlets	19,500
Days per year for PMMC changes	190
Days per year for automatic changes	365
Days per month (PMMC)	15.83
Days per month (auto)	30.42

FUJ00098154

1691 **AVERAGED EVENT TRAFFIC**

Event	Payload bytes	Quantity	Interval (months)	Events per outlet per day	Total events per day	Total bytes per day
Non-Riposte Events						
OpenDoorReq	100	1	24	0.0026	51.32	5,132
IntExchAck	100	1	24	0.0026	51.32	5,132
					102.63	**10,263**
Riposte Key Deliveries						
PMMCKeyChangeReq	100	1	24	0.0014	26.71	2,671
ConfK	750	2	24	0.0027	53.42	40,068
CRL	1,000	1	12	0.0027	53.42	53,425
PKC	750	2	20	0.0033	64.11	48,082
CAPU	128	1	3	0.0110	213.70	27,353
					411.37	**171,600**

Event	Payload bytes	Quantity	Interval (months)	Events per outlet per day	Total events per day	Total bytesper day
Riposte Acknowledgments						
Ack.Installed	100	2	24	0.0027	53.42	5,342
Ack.Received	100	2	24	0.0027	53.42	5,342
Ack.PKC	100	2	20	0.0033	64.11	6,411
Ack.CRL	100	2	12	0.0055	106.85	10,685
AckOK.CAPU	100	2	3	0.0219	427.38	42,738
AckNotOK.CAPU	100	1.00E-04	3	1.10E-06	0.02	2
PMMCKeyChangeAck	100	1	24	0.0014	26.71	2,671
NewPMMCAck	100	1	24	0.0014	26.71	2,671
IntExchAck	100	1	24	0.0014	26.71	2,671
PMMCKeyChangePrompt	100	1	24	0.0014	26.71	2,671
					812.05	**81,205**

1692

FUJ00098154

1693 **PEAK EVENT TRAFFIC**

Event	Type	Payload bytes	Quantity	Interval (months)		Total events	Total bytes
Routine (PAPU to all outlets)							
PKC	Riposte	750	1	21		**19,500.00**	**14,625,000**
Acks	Riposte	100	1	21		**19,500.00**	**1,950,000**

1694

DISASTER RECOVERY

(via Riposte)

ConfK, CRL, PKC	4000	5			**97,500.00**	**78,000,000**
Acks	1000	10			**194,998.05**	**19,499,805**

FUJ00098154

4. RELEASE NR2+ IMPLEMENTATION

4.1 Protection domain management outlines

In this section, we consider the specific key distribution mechanisms for each protection domain. In addition to the business-oriented cryptographic applications identified in section 1.2, the Key Management system is itself a cryptographic application and introduces protection domains of its own in addition to those depicted in Figure 2. Each protection domain involves a number of physical platforms and these are identified in section 4.1.1. Section 4.1.2 discusses each of the business protection domains and section 4.1.3 identifies and discusses the KM protection domains. The main purpose of this discussion is to identify the KM software inventory for each client platform and to show how the various software components of section 3.14 are used to realise this software inventory.

In each protection domain, a diagram is given showing the relevant clients and the distribution channels used to supply key material to each client. As part of the platform specification for each platform in each protection domain appropriate software components as listed in section 3.14. A table showing the correspondence will be provided in a later edition of this document (an earlier attempt to give a table for each protection domain was found to be rather unsuccessful).

The keys to be managed in the NR2+ system are listed in the following table, which also includes a few items that are delivered by KM and may conveniently be thought of as "key material" for the purposes of this section.

FUJ00098154

1713

Key name	Protection Domain	Application area	Algorithm	Comments
APPR	AP	Automated Payment	DSA	
APPU	AP	Automated Payment	DSA	
CAPR	CA	Certification Authority	DSA	1024-bit
CAPS	CAPS	Customer Accounting and Payment Strategy	Red Pike	
CAPU	CA	Certification Authority	DSA	1024-bit
CK	(n/a)	Communications Key	Red Pike	
CMS	CMS	Card management service	Red Pike	
CRL	(any DSA user)	KMS certification revocation list	Other	
DLLKA	L&G enabling	L&G Enabling	Other	
DLLKB	L&G enabling	L&G Enabling	Other	
EVPN	Utimaco VPN	Virtual private network exception key	RSA	
FEK	FEK	Filestore Encryption Key	Red Pike	
FTPPR	AP Client	Automated payment file transfer protocol	DSA	
FTPPU	AP Client	Automated payment file transfer protocol	DSA	
GDK	L&G code	L&G Code Encryption	Red Pike	
KIPR	KI	Key Issue	DSA	
KIPU	KI	Key Issue	DSA	
KMA	KMA	Key management application key	Red Pike	
NVPN	Utimaco VPN	Virtual private network normal key	RSA	
PAPR	PA	Payment Authorisation	DSA	
PAPU	PA	Payment Authorisation	DSA	
POCL TIPPR	POCL TIP	POCL transaction information processing and reference data	DSA	
POCL TIPPU	POCL TIP	POCL transaction information processing and reference data	DSA	
POK	POK	Post Office Key	Red Pike	
PWY TIPPR	PWY TIP	POCL transaction information processing and reference data	DSA	
PWY TIPPU	PWY TIP	POCL transaction information processing and reference data	DSA	
Rambutan Prompt	Rambutan	Rambutan key management	Other	
SIPR	SI	Software Issue	DSA	
SIPU	SI	Software Issue	DSA	
TK	TK	KMC traffic key	Red Pike	
VPN CRL	Utimaco VPN	Virtual private network CRL	RSA	

1714 Notes:

1715 1. CK is a transient key generated during certain key transfers; it belongs to no particular protection
1716 domain.

FUJ00098154

2. The KM support for the Rambutan domain comprises only a reminder service for the Pathway Key Manager.

4.1.1 Platform Definitions

The physical platforms known to the KM Controller in the NR2+ steady state are shown in Figure 27. This figure shows all the platforms that feature in the key distribution diagrams in sections 4.1.2 and 4.1.3 below. The physical locations of non-Pathway sites other than PO outlets are not shown in the diagram for simplicity. Wherever possible the names used in the figure are those used in "Technical Environment Description" [TED]. Further information on the platforms is given in [KMPLATFORMS].

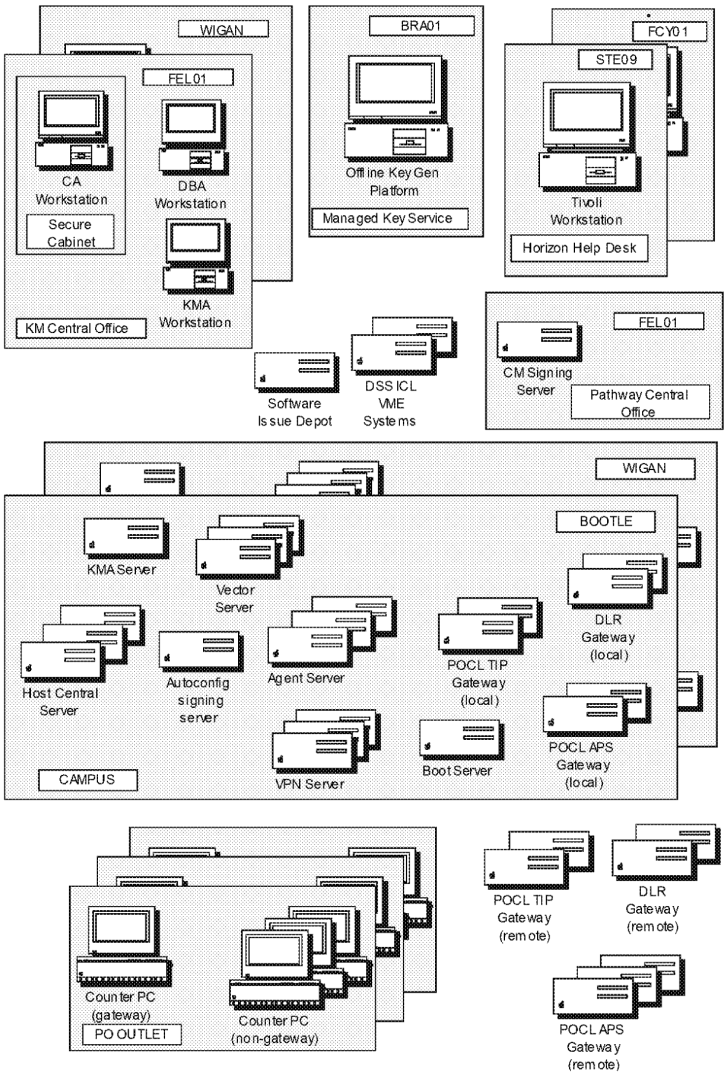


Figure 27. KM System and Client Platforms

FUJ00098154

4.1.2 Business protection domains

4.1.2.1 AP

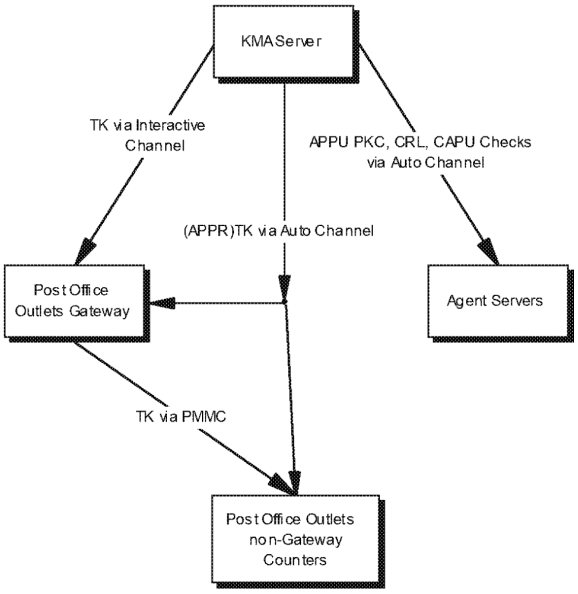


Figure 28. AP Key Distribution

4.1.2.2 AP Client

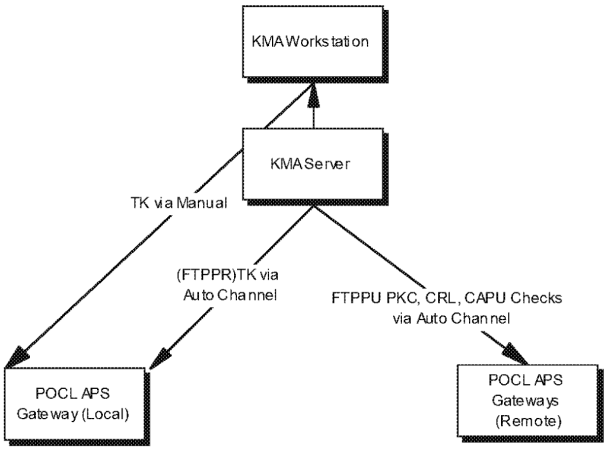
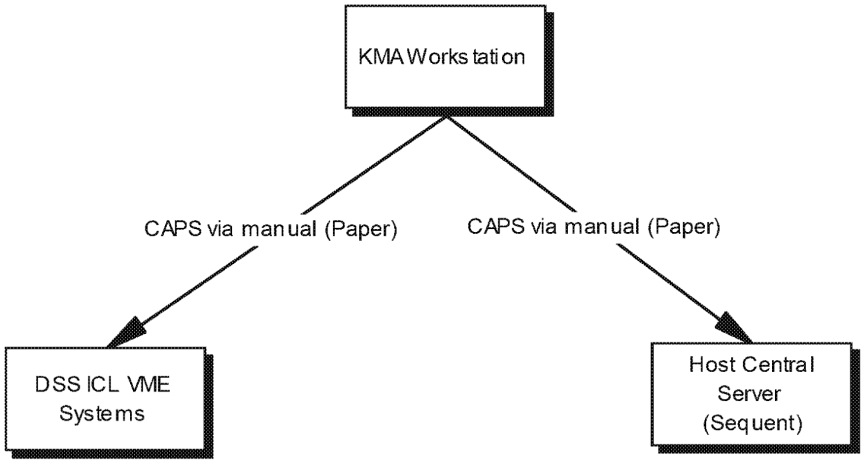


Figure 29. AP Client Key Distribution

FUJ00098154

1738 *4.1.2.3 CAPS*

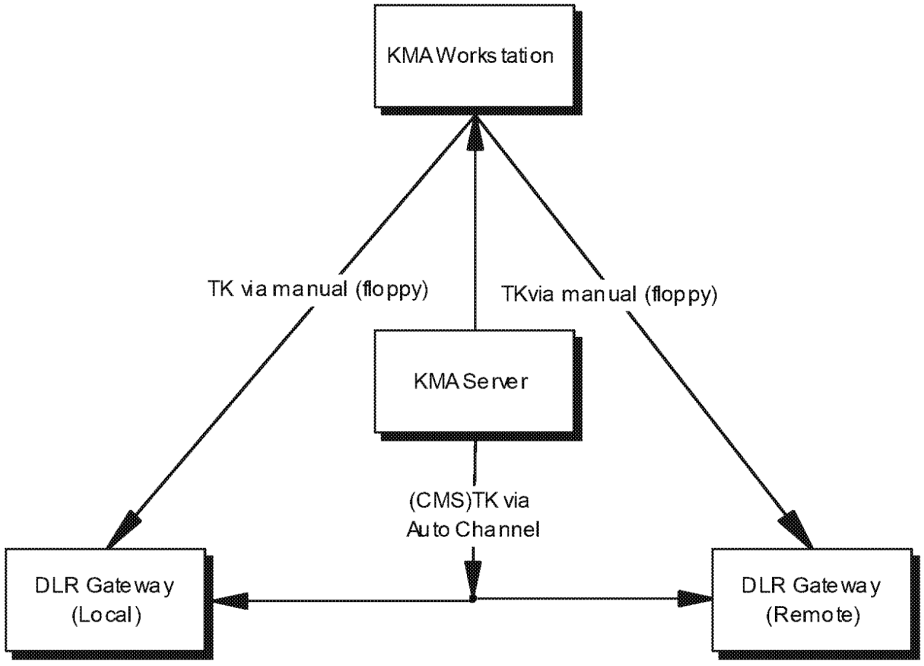
1739



1740

1741 Figure 30. CAPS Key Distribution

1742 *4.1.2.4 CMS*



1743

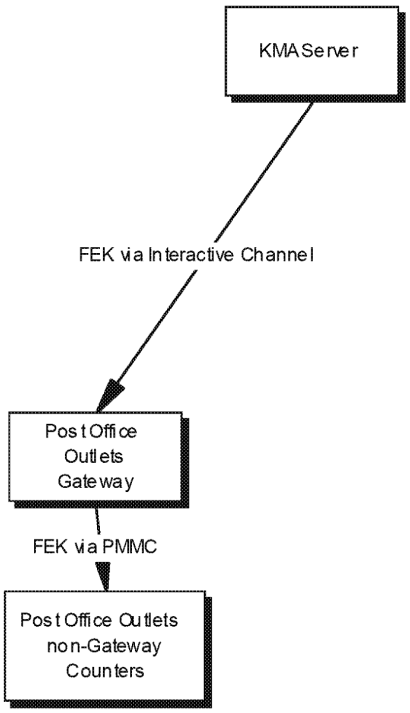
1744

1745 Figure 31.CMS Key Distribution

FUJ00098154

1746 *4.1.2.5 FEK*

1747



1748

1749 Figure 32. FEK Key Distribution

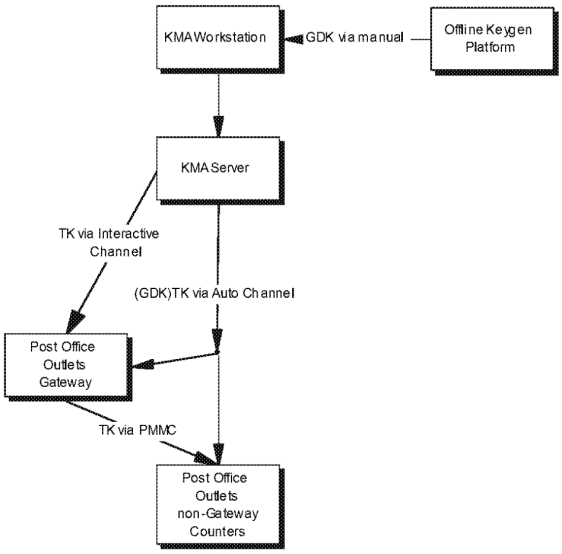
1750

FUJ00098154

1751

1752 *4.1.2.6 L&G Code*

1753

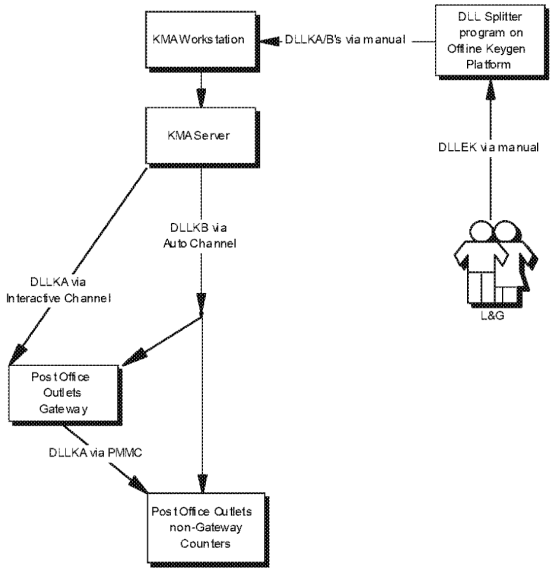


1754

1755 Figure 33. L&G Code Key Distribution

1756 The key GDK is generated on the Offline Key Generation platform as part of the code encryption process
1757 described in [LANDGCRYPTO].

1758 *4.1.2.7 L&G Enabling*



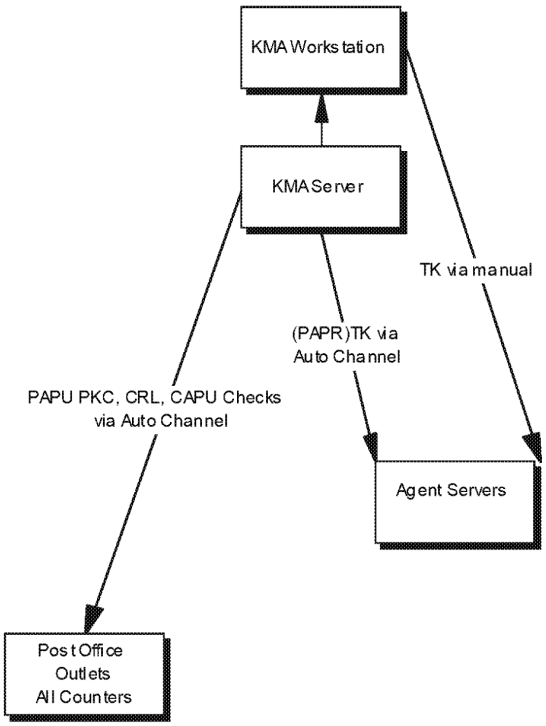
1759

1760 Figure 34. L& G Enabling Key Distribution

FUJ00098154

1761 *4.1.2.8 PA*

1762

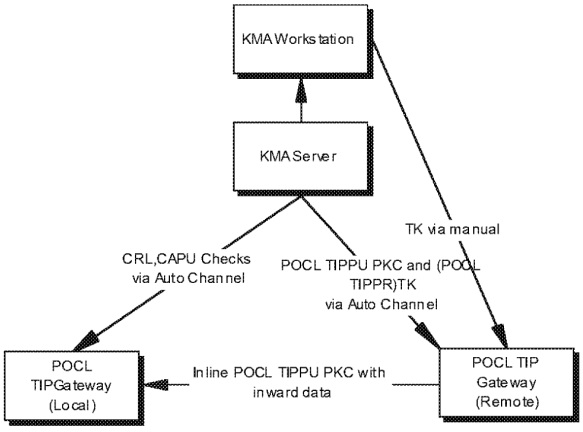


1763

1764 Figure 35. PA Key Distribution

1765 *4.1.2.9 POCL TIP*

1766



1767

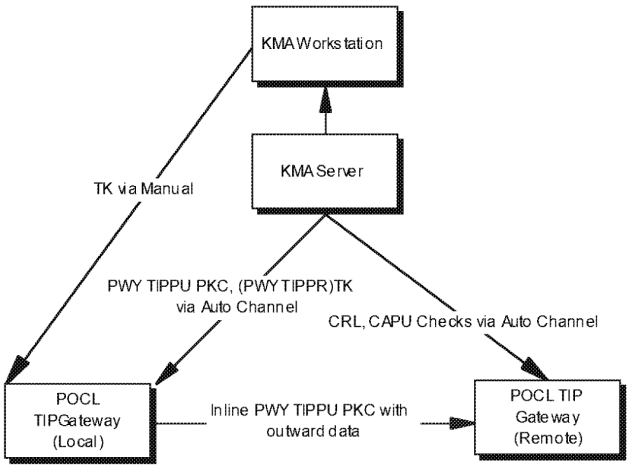
1768 Figure 36. POCL TIP Key Distribution

1769

FUJ00098154

1770 *4.1.2.10 PWY TIP*

1771



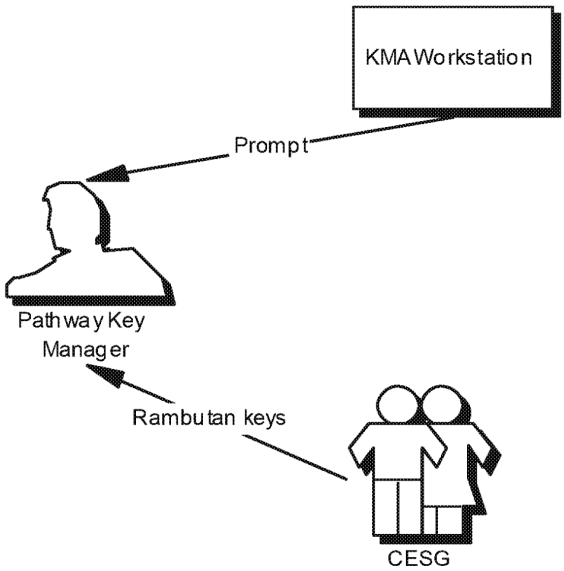
1772

1773 Figure 37. PWY TIP Key Distribution

1774

1775 *4.1.2.11 Rambutan*

1776

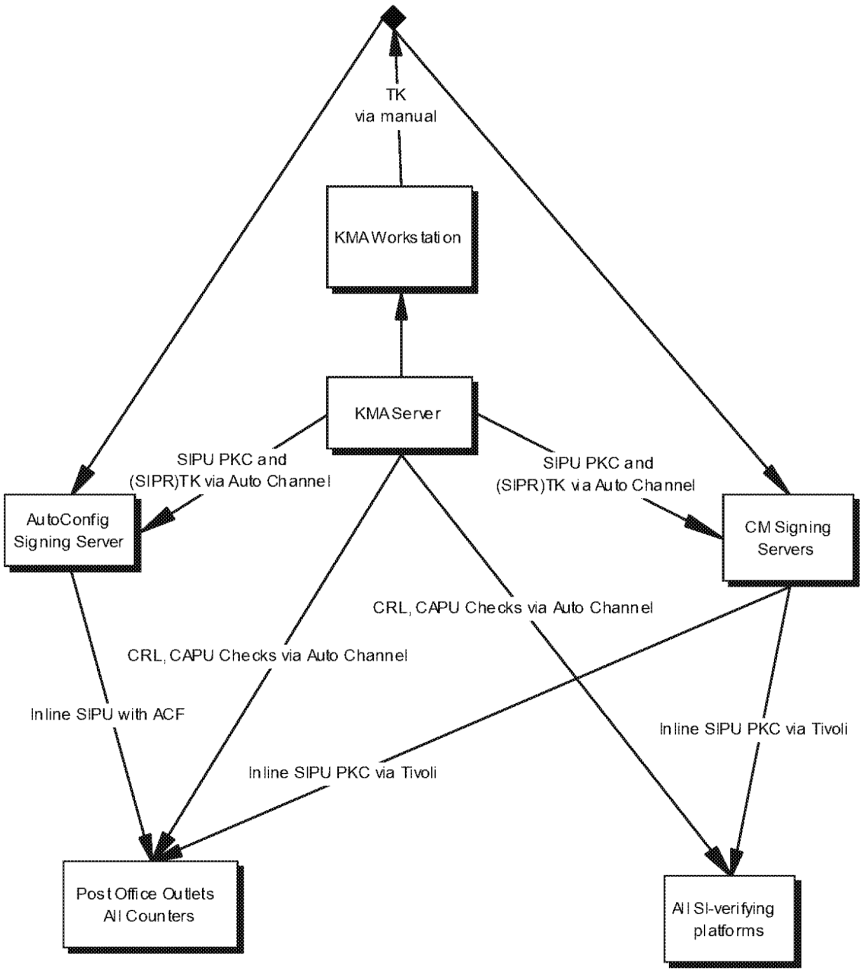


1777

1778 Figure 38. Rambutan Key Distribution

FUJ00098154

1779 *4.1.2.12 SI*



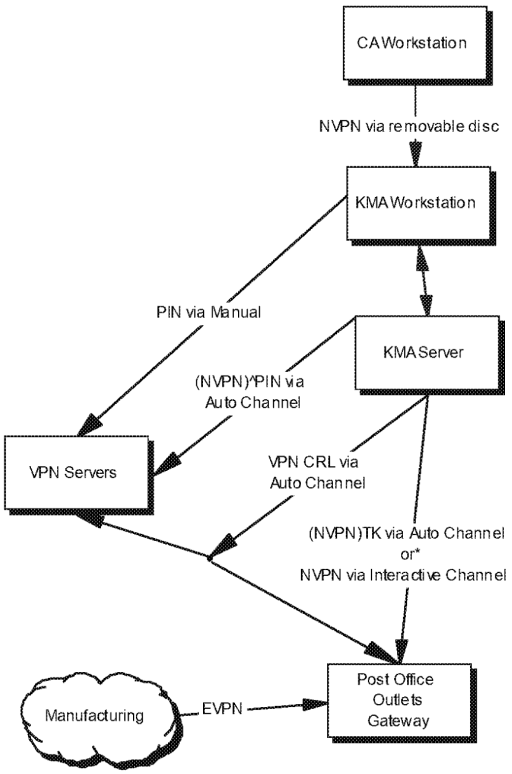
1780

1781

1782 Figure 39. SI Key Distribution

FUJ00098154

1783 *4.1.2.13 Utimaco VPN*



1784

1785 Figure 40. Utimaco VPN Key Distribution

1786 Legend: * - interactive channel at bootstrap, Auto channel otherwise (see section 3.12.6).

1787 In the above diagram, the notation (NVPN)^PIN means the NVPN key protected by the Utimaco
1788 software using triple-DES encryption with the securely managed PIN as the key. The value NVPN
1789 sent to the PO outlets is in fact also encrypted in the same way by the Utimaco software but using an
1790 unmanaged global key., Since the latter encryption is not security-relevant, it is not shown in the
1791 diagram (as far as the KMS is concerned the NVPN value sent to the outlet is the confidential key
1792 material that is to be delivered).

1793

FUJ00098154

4.1.3 KM protection domains

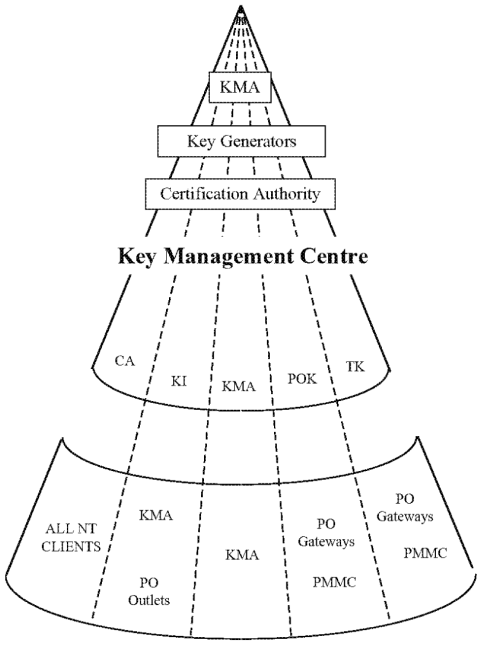


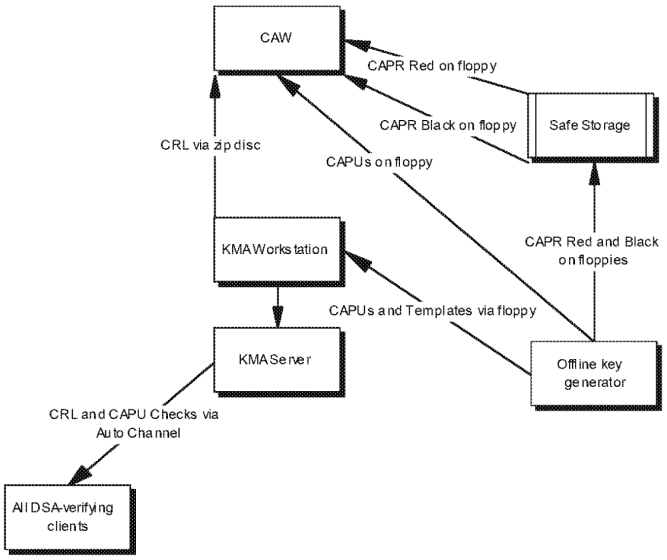
Figure 41. KMS Protection Domain "fan diagram"

FUJ00098154

1798

1799 *4.1.3.1 CA*

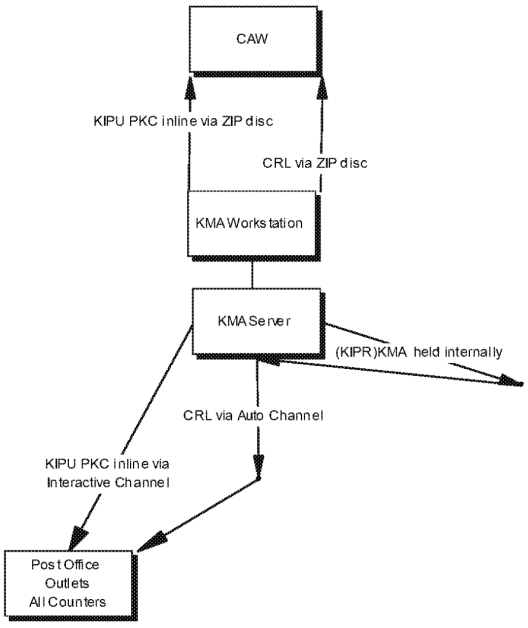
1800



1801

1802 Figure 42. CA Key Distribution

1803 *4.1.3.2 KI*



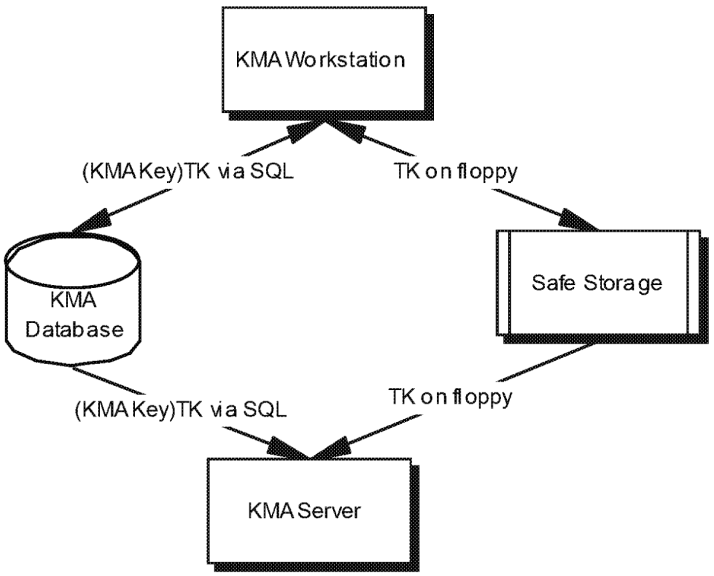
1804

1805 Figure 43. KI Key Distribution

FUJ00098154

1806 *4.1.3.3 KMA*

1807

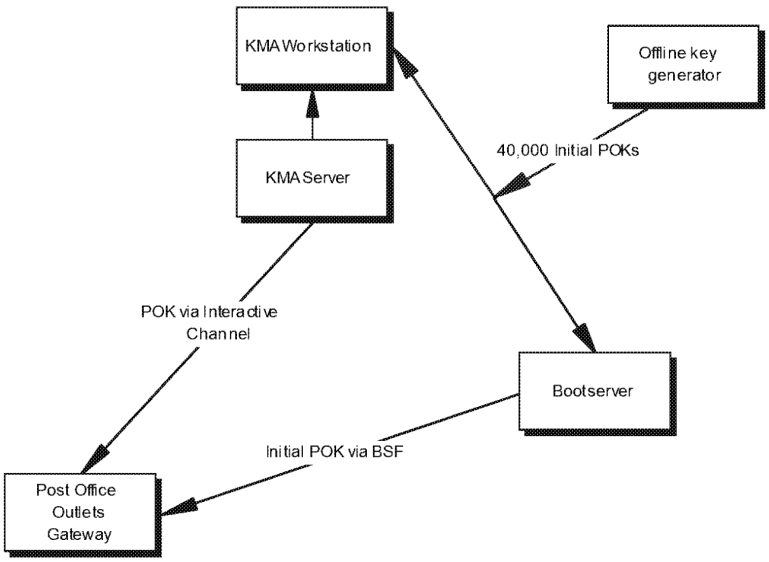


1808

1809 Figure 44. KMA Key Distribution

1810 *4.1.3.4 POK*

1811



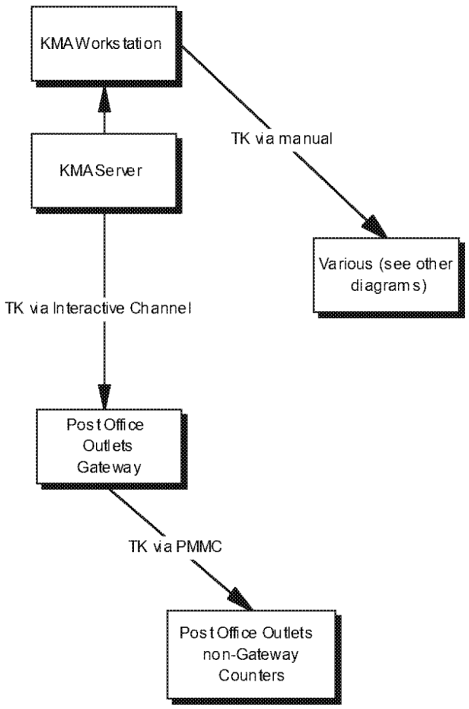
1812

1813 Figure 45. POK Key Distribution

FUJ00098154

1814 *4.1.3.5 TK*

1815



1816

1817 Figure 46. TK Key Distribution

1818

FUJ00098154

4.1.4 Component Distribution

The following table shows which software components are needed to service which key deliveries to which clients (the table is in two parts due to a limitation in the interface between Word and Excel)

Protection Domain	Key name	Client	Distribution Mechanism	CAPU check harvester	Certification authority	Client D-H module	CRL harvester	Help Desk GUI	Help Desk Processor	Key destroy harvester	Key dispatch Agent	Key generators	Key install harvester	Key load/unload module	Key store booter	KM application	KM distribution loader	KM distribution receiver	KM monitoring dispatcher	KM monitoring harvester	KM reboot manager	KMC D-H module	POLO GUI
AP	APPR	PO outlet	Riposte									Y	Y	Y		Y	Y	Y	Y	Y			
AP	APPU	Agent Server	Riposte	Y	Y							Y	Y	Y		Y	Y	Y	Y	Y			
AP	CAPU	Agent Server	Riposte	Y	Y								Y	Y		Y	Y	Y	Y	Y			
AP	CRL	Agent Server	Riposte	Y	Y		Y			Y						Y	Y	Y	Y	Y			
AP Client	CRL	POCL APS Gateway (remote)	Riposte	Y	Y		Y			Y						Y	Y	Y	Y	Y			
AP Client	FTPPR	POCL APS Gateway (local)	Manual											Y	Y	Y							
AP Client	FTPPU	POCL APS Gateway (remote)	Riposte	Y	Y							Y	Y	Y		Y	Y	Y	Y	Y			
CA	CAPR	CAW	Manual									Y		Y		Y							
CA	CAPU	ALL NT Clients	Riposte	Y	Y							Y	Y	Y		Y	Y	Y	Y	Y			
CA	CAPU	CAW	Other													Y							
CA	CAPU	KMA Server	Other											Y		Y							
CA	CRL	KMA Server	Riposte	Y	Y		Y			Y						Y	Y	Y	Y	Y			
CA	CRL	ALL NT Clients	Manual				Y			Y					Y	Y							
CAPS	CAPS	DSS ICL VME System	Manual									Y				Y							
CAPS	CAPS	Host central server	Manual													Y							
CMS	CMS	DLR Gateway (remote)	Manual									Y		Y	Y	Y							
CMS	CMS	DLR Gateway (local)	Manual											Y	Y	Y							
FEK	FEK	PO outlet	Interactive			Y		Y	Y			Y		Y		Y					Y	Y	Y
KI	KIPR	KMA Server	Other									Y		Y		Y							
KI	KIPU	CAW	Manual									Y				Y							
KI	KIPU	PO Outlet	Interactive											Y		Y						Y	Y
KMA	KMA	KMA Server	Manual									Y			Y	Y							
KMA	KMA	KMA Workstation	Manual												Y	Y							
L&G code	GDK	PO outlet	Riposte	Y	Y							Y	Y	Y		Y	Y	Y	Y	Y			
L&G code	GDK	Code encrypter TBD	Manual													Y							
L&G enabling	DLLKA	PO outlet	Riposte	Y	Y							Y	Y	Y		Y				Y	Y		
L&G enabling	DLLKB	PO outlet	Interactive	Y	Y	Y		Y	Y			Y	Y	Y		Y						Y	Y
PA	CAPU	PO outlet	Riposte	Y	Y								Y			Y	Y	Y	Y	Y			
PA	CRL	PO outlet	Riposte	Y	Y		Y			Y						Y	Y	Y	Y	Y			
PA	PAPR	Agent servers	Riposte	Y	Y								Y	Y		Y	Y	Y	Y	Y			
PA	PAPR	Agent servers	Manual												Y	Y							

PA	PAPU	PO outlet	Riposte	Y	Y							Y	Y	Y		Y	Y	Y	Y	Y			
POCL TIP	CAPU	POCL TIP Gateway (local)	Riposte	Y	Y											Y	Y	Y	Y	Y			
POCL TIP	CRL	POCL TIP Gateway (local)	Riposte	Y	Y		Y			Y						Y	Y	Y	Y	Y			
POCL TIP	POCL TIPPR	POCL TIP Gateway (remote)	Riposte	Y	Y							Y	Y	Y		Y	Y	Y	Y	Y			
POCL TIP	POCL TIPPU	POCL TIP Gateway (local)	Riposte	Y	Y							Y	Y	Y		Y	Y	Y	Y	Y			
PWY TIP	CAPU	POCL TIP Gateway (remote)	Riposte	Y	Y								Y			Y	Y	Y	Y	Y			
PWY TIP	CRL	POCL TIP Gateway (remote)	Riposte	Y	Y		Y			Y			Y			Y	Y	Y	Y	Y			
PWY TIP	PWY TIPPR	POCL TIP Gateway (local)	Riposte	Y	Y							Y	Y	Y		Y	Y	Y	Y	Y			
PWY TIP	PWY TIPPU	POCL TIP Gateway (remote)	Riposte	Y	Y							Y	Y	Y		Y	Y	Y	Y	Y			
POK	POK	PO outlet	Interactive			Y		Y	Y							Y					Y	Y	Y
Rambutan	Prompt	Pathway key manager	Other													Y							
SI	CAPU	All SI-verifying platforms	Riposte	Y	Y											Y	Y	Y	Y	Y			
SI	CRL	All SI-verifying platforms	Riposte	Y	Y		Y			Y						Y	Y	Y	Y	Y			
SI	SIPR	Autoconfig signing server	Manual												Y	Y							
SI	SIPU	Autoconfig signing server	Riposte	Y	Y							Y	Y	Y		Y	Y	Y	Y	Y			
SI	SIPR	CM signing servers	Manual												Y	Y							
SI	SIPU	CM signing servers	Riposte	Y	Y							Y	Y	Y		Y	Y	Y	Y	Y			
TK	TK	PO Outlet	Interactive			Y		Y	Y					Y		Y						Y	Y
TK	TK	Agent servers	Manual									Y			Y	Y							
TK	TK	POCL APS Gateway (local)	Manual									Y			Y	Y							
TK	TK	POCL APS Gateway (remote)	Manual									Y			Y	Y							
TK	TK	POCL TIP Gateway (local)	Manual									Y			Y	Y							
TK	TK	POCL TIP Gateway (remote)	Manual									Y			Y	Y							
TK	TK	Autoconfig signing server	Manual									Y			Y	Y							
TK	TK	CM signing servers	Manual									Y			Y	Y							
Utimaco VPN	NVPN	PO outlet	Riposte									Y	Y			Y	Y	Y	Y	Y			
Utimaco VPN	NVPN	PO outlet	Interactive			Y		Y	Y							Y						Y	Y
Utimaco VPN	EVPN	PO outlet	Other									Y				Y							

FUJ00098154

4.2 Key management application

4.2.1 Key scheduling

The KMA instigates the creation and/or distribution of replacement keys whenever keys in service are approaching the end of their lifetime. The periods of validity for keys in each protection domain are defined in the Key Management Requirements [KMREQ].

The distribution of a public key certificate must precede the distribution of the corresponding private key.

4.2.1.1 Pre-delivered keys

A stock of CAPU keys is pre-delivered into all clients that need them via the manufacturing process.

These keys do not therefore need to be generated or distributed to those platforms when the corresponding private keys are due for change. However, see the later note about confirmation copies.

4.2.1.2 Pre-generated keys

The private keys CAPR corresponding to CAPU are pre-generated and held securely off-line. The KMA does not, therefore, instigate their generation but calls for each CAPR to be introduced from secure storage when it is time to replace the current one.

The Red Pike key GDK used is pre-generated and delivered manually to the facility that encrypts the L&G code.

4.2.1.3 Just-in-time keys

The KMA instigates the generation and distribution of all keys in all protection domains other than CA, POK (for the initial POK values, see Figure 45), L & G Code, L&G Enabling and Rambutan. I.e., keys in all the other protection domains are generated just-in-time.

4.2.1.4 Third-party keys

Keys in the following protection domains are supplied by third parties.

L&G Enabling	supplied by Landis & Gyr. This key is stored and distributed by the KMA. This key is not subject to routine replacemenmt. It is only replaced at the instigation of the supplier, or at the Key Manager's discretion in the case of a key compromise. The replacement may involve a co-ordinated change to the installed (and protected) L&G code. Note that the protection arrangements for this key are unusual (see [LANDGCRYPTO]).
Rambutan	supplied by CESG. These keys are not stored or distributed by the KMA, but the KMA does prompt and track their manual handling. The KMA prompts for their replacement at regular intervals as recommended by the supplier.
VPN	These keys are provided by the CA workstation which includes the bought-in Utimaco key generation and certification system.

4.2.1.5 CAPU confirmation

At configurable intervals, the KMA broadcasts a digest of all the CA public keys to all managed clients that use them. The clients then compare the broadcast values with the pre-delivered values and take action on any discrepancy.

FUJ00098154

4.2.1.6 Latency

The policies of section 2.6 are supported in that a configurable maximum expected latency period is associated with each public/private key pair. This period is a factor in determining the interval between replacement of the private key and expiry of the public key certificate.

4.2.2 Key routing

The tables of sections 4.1.2 and 4.2 show the routes the KMA uses to distribute keys.

4.3 Certification Authority

All public keys generated by the KM service other than CA itself are certified. With the exception of third party keys, these keys are all generated in Layer 7 format, and all PKCs are to be used in clients whose crypto code is based on Layer 7. Therefore the CA application is implemented using the Layer 7 cryptographic library.

In the case of VPN keys, the certification is via Utimaco's key generation and certification product which is integrated into the CA application. See "Integrating Utimaco Code" [INTUTIMACO] for more details.

Detailed design of the CA application is documented in "Detailed Design for Certification" [KMCAWDES].

4.4 Key generators

The following tables relates protection domains to the key generators or other sources of keys which will serve them.

	AP	SI	PA	FEK	CAPS	CMS	POCL TIP	PWY TIP	AP Client	L&G Code	L&G Enable	VPN	Rambutan
Layer 7 DSA	✓	✓	✓				✓	✓	✓				
Layer 7 Red Pike						✓				✓			
ICL Red Pike (CAPS)					✓								
TeamWARE Crypto Red Pike (FEK)				✓									
Utimaco												✓	
3rd party supply											✓		✓

	CA	KI	KMA	POK	TK
Layer 7 DSA	✓	✓			
Layer 7 Red Pike			✓	✓	✓

FUJ00098154

4.4.1 Layer 7 DSA key generator

4.4.1.1 Implementation

The Layer 7 DSA key generator will be implemented using the Layer 7 cryptographic library, which provides key generation functions. As shown in figure Figure 13 instances of this key generator are available both on the KMA workstation and on the KMA server. In addition a physically isolated instance of this key generator is used by the Managed Key Service in a List-X secure environment at ICL BRA01 to generate the CA private keys.

4.4.1.2 Function

This process produces an asymmetric key pair for use with the DSA signature algorithm. It is both a *key generator* and a *secure key packaging* process: it encrypts the private key under the KEK using the Red Pike algorithm. Therefore it requires a KEK as input.

The CA keys will be generated with 1024-bit length. All other DSA keys are 768-bit.

4.4.1.3 Inputs

1. A DSA key template containing the computational constants P, Q and G. Note that different templates will be needed for different key lengths.
2. A Red Pike 64-bit key, in the form of a red key file, to be used as the KEK.
3. Random numbers supplied by a Comscire hardware random number generator.

The DSA constant data values known as P, Q and G will be supplied by CESG.

4.4.1.4 Outputs

The key generator produces its outputs in a format that is convenient for the KMA (see "KMA Design" [KMAPDES]). The keys are delivered to the clients in the following formats:

1. Private DSA key: Layer 7 key transport file format (containing the private key encrypted under the KEK).
2. CA only: Layer 7 public key file containing the unprotected public key, CAPU.
3. Other public DSA keys: public key certification containing the public key protected under the CA key.

4.4.2 Layer 7 Red Pike key generator

4.4.2.1 Implementation

The Layer 7 Red Pike key generator will be implemented using the Layer 7 cryptographic library, which provides key generation functions.

4.4.2.2 Function

This process produces 64-bit keys for the Red Pike symmetric algorithm. They may be used as a KEK (for input to the Layer 7 DSA generator) or as a DEK.

4.4.2.3 Inputs

1. Random numbers supplied by a Comscire hardware random number generator.

FUJ00098154

2. Optionally, a Red Pike KEK.

4.4.2.4 Outputs

The key generator produces its outputs in a format that is convenient for the KMA (see "KMA Design" [KMAPDES]). The keys are delivered to the clients in either of the following formats:

1. Layer 7 red key file containing the unprotected Red Pike key
2. Layer 7 key transport file containing the generated key encrypted under the KEK.

4.4.3 CAPS

4.4.3.1 Implementation

There are two cryptographic implementations in the CAPS domain: one custom implementation of the Red Pike algorithm on VME platforms and another on Unix (Dynix) platforms. Both accept keys in the same format, which is described in the CAPS cryptographic interface definition [CAPSINTSPEC].

4.4.3.2 Function

A newly generated key value will be encrypted using the previously installed key as a key-encryption-key (KEK) and Red Pike as the key encryption algorithm.

The key value and some additional control data are packaged as an alphanumeric coding of the binary values and printed on hard copy.

4.4.3.3 Inputs

1. The value of the previously installed key, retrieved from the CAPS key journal.
2. A 64-bit random number supplied by a Comscire hardware random number generator.

4.4.3.4 Outputs

The KMA uses the output of the key generator to produce a printed string, as specified in the CAPS cryptographic interface definition [CAPSINTSPEC].

4.4.4 FEK

4.4.4.1 Implementation

Each post office Filestore Encryption Key is used with a proprietary filestore encryption system called TeamWARE Crypto. The FEK is currently (in release 1c and 2 legacy) generated as a simple binary number, as a by-product of a Layer 7 Red Pike generation. It is then repackaged by the client before installation in the TeamWARE Crypto functions. These processes are open to redesign.

4.4.4.2 Function

A new online key generator will be used at the Key Management Controller (rather than on the Post Office counter), which will produce Filestore Encryption Keys in the same manner as above, but using Comscire hardware-generated random numbers. This generator will not securely package the FEK. Protection will be left to the distribution channel.

FUJ00098154

4.4.4.3 Inputs

1. A 64-bit random number supplied by a Comscire hardware random number generator.

4.4.4.4 Outputs

Simple binary red pike key (64 bits)

4.4.5 Utimaco VPN

VPN keys are generated and packaged using Utimaco products integrated into the CA system. See "Integrating Utimaco Code" [INTUTIMACO] for details.

4.5 Distribution and Monitoring Channels

4.5.1 Automatic distribution and monitoring channels

The automatic distribution and monitoring channels are used for all clients that have keys managed via the Riposte service. At NR2+ the channels conform with the description in sections 3.3 and 3.6 above. See "KM Automatic Channel Detailed Design" [KMACDES] for more information. For convenience, this part of the design documentation also covers the automatic communications mechanisms required between the nodes in a PO outlet to address the synchronisation issue discussed in section 3.10.

4.5.2 Interactive distribution channel

At NR2+, this channel is used to deliver key encryption keys and other material destined for the PMMC to PO gateways PCs.

The design of the interactive distribution channel at NR2+ conforms with the description in section 3.8. See "Detailed Design of KM Interactive Channel" [KMICDES] for more information. Note that that document covers the mechanism for transporting a set of PMMC and other keys to a gateway PC. It does not cover the physical dissemination of key material within a PO outlet using the PMMC (which is described in "PMMC Agent Design" [PMMCADES] and "KM Client Agent Design" [KMCAGDES]).

4.5.3 Manual distribution channels

The following list relates key clients to the keys which they receive via manual channels.

CM signing server	The SI red key file, which is the KEK for the SI private key, is delivered on portable physical medium to the Cryptographic Key Custodian of the CM signing server.
BPS loader agents	The PA red key file, which is the KEK for the PA private key is delivered on portable physical medium to the Cryptographic Key Custodian at the Pathway campuses.
CAS VME	The Red Pike key for data encryption on the CAS VME platform is delivered by printed copy to the Cryptographic Key Custodian at the EDS site(s). Red Pike encryption using the previous value of the key provides a simple integrity check and some measure of confidentiality.

FUJ00098154

CAS Oracle platform	The Red Pike key for data decryption on the CAS Oracle database platform is delivered by printed copy to the Cryptographic Key Custodian at the data centre. Red Pike encryption using the previous value of the key provides a simple integrity check and some measure of confidentiality.
DLR gateways (local and remote)	The TK file containing the key encryption key for the CMS Red Pike key will be delivered to the Cryptographic Key Custodian at the relevant sites.
POCL APS gateways (local and remote)	The TK file containing the key encryption key for the FTP private key will be delivered to the Cryptographic Key Custodian at the relevant sites.
POCL TIP gateway (local and remote)	The TK file containing the key encryption key for the PWY TIP private key will be delivered to the Cryptographic Key Custodian at the relevant sites.
Autoconfig and CM signing servers	The TK file containing the key encryption key for the SI private key will be delivered to the Cryptographic Key Custodian at the Pathway campuses.
Boot server	A one-off delivery of 40,000 POK key/keytag pairs will be made from the Managed Key Service at ICL BRA01 to the Cryptographic Key Custodian at the Pathway Campuses.
VPN servers	The TK file containing the key encryption key for the NVPN key will be delivered to the Cryptographic Key Custodian at the Pathway campuses. The VPN PIN is delivered manually.
CA workstation	A one-off delivery of 20 diskettes each containing a CA private key will be made from the Managed Key Service at ICL BRA01 to the Pathway Key Manager at ICL FEL01. These are taken from safe storage and used to install the CA private key on the CAW according to policy.
KMA server	The KMA key is manufactured on diskette by the Pathway Key Manager at ICL FEL01 and distributed on diskette to the Cryptographic Key Custodian at the Pathway campuses.

The operation of manual key channels will be detailed in procedure documents.

4.6 Key Management Client Agent

4.6.1 DSA private key

A DSA private key is delivered down the automatic distribution channel as a key transport file containing the encrypted private key, protected using the client's key encryption key TK. The key encryption key is delivered either via the manual channel (non-PO platforms) or the interactive channel (PO platforms).

4.6.1.1 Installation

Installation (i.e., activation) of a DSA private key delivered via the KM client agent occurs at the point in time when both the private key capsule and the TK that encrypts are first available at the client (see the protocol description in section 3.12.2). Installation of itself requires no movement of the key material in persistent data stores.

The key encryption key, TK, is loaded either by a key store booter (see section 3.4) or by the PMMC Agent (see section 3.9). The policy described in section 3.12.2 ensures that the latest confidential key that can be decrypted using the available TK will be used when a crypto application requires the signing key. Installation of a key occurs at the point when both the key and the corresponding value of TK are

FUJ00098154

available, at this point, the old signing key is unloaded (see below) and this will cause subsequent signing calls to use the new key.

4.6.1.2 Loading

Loading is automatically invoked by any of the following events, depending on platform and application configuration:

a) an application calls an explicit key-loading API;

b) an application calls a cryptographic signing function, which in turn calls the key-loading module if necessary.

The policy described in section 3.12.2 ensures that an appropriate key is loaded according to the TK provided at boot time. The physical procedures followed by the key custodian are intended to ensure that the correct PMMC or diskette is used.

4.6.1.3 Unloading

The key is unloaded using the Layer 7 facilities (e.g., STOR_RemoveKeyxxx). A semaphore is used to lock out the crypto applications while this is being done.

4.6.1.4 Off-line storage

Except when the private key is being loaded, the key encryption key will be stored in a physical safe on the same site as the key client platform. Only the Cryptographic Key Custodian and the Cryptographic Key Handlers will have access to this safe.

4.6.1.5 Online storage

When delivered via the automatic channel, the key part which was delivered (e.g. the key transport file) is stored on the fixed disc of the key client platform.

4.6.1.6 Revocation

There will be no separate revocation process. Installation of a new key implicitly revokes the previous key by overwriting the configuration details.

4.6.1.7 Destruction

A DSA private key is destroyed by deleting the two key parts. An interactive process will allow the Cryptographic Key Custodian to delete the part which is in online storage. Manual procedures will require the custodian to return the medium containing the other part to the Pathway Key Manager for obliteration.

4.6.2 DSA public key certificate

4.6.2.1 Installation

No installation process is required (although receipt is acknowledgeed following the protocol of section 3.12.3).

Where a certificate is delivered to a platform after installation, the automatic distribution channel will add the PKC to a collection known to the key load/unload module.

FUJ00098154

4.6.2.2 Loading

The cryptographic verification functions will invoke the PKC loading process when required. The process will use the current CAPU to verify the certificate, will check the expiry date of the certificate and will also search the current certificate revocation list for the key tag. If all is well, the public key, not the entire certificate, is loaded into process memory for use by the verification functions.

If the CA signature on the certificate cannot be verified with the current CAPU, loading fails and the certificate is not available to the cryptographic verification functions. A security event is logged and subsequent attempts to verify signatures using the key in the certificate will return verification failure response codes to the calling application.

If the certificate has expired or been revoked, at NR2+, the calling application will be notified that the certificate is invalid. The expiry date should be checked against the later of the system clock and the timestamp on the CRL currently in memory (so that if either this platform or the KMC thinks the certificate has expired, it has expired).

When a new CRL arrives, the CRL handler will unload all keys revoked by the CRL. Thus the verify code does not have to check the key on every verification.

4.6.2.3 Unloading

The key is unloaded using the Layer 7 facilities (e.g., STOR_RemoveKeyxxx). A semaphore is used to lock out the crypto applications while this is being done.

4.6.2.4 Off-line storage

None.

4.6.2.5 Online storage

All certificates will be stored in a single collection (e.g. individual files in a single directory) on the fixed disk of the key client platform.

4.6.2.6 Revocation

The revocation process will receive certificate revocation lists (CRL) from the KMA delivered via the automatic distribution channel. Each message lists all the key tags of all the public keys that are currently revoked and not expired. Each new CRL will therefore be cumulative, with newly revoked certificates added and those which have expired removed. A delivered CRL is not accepted if its date and time stamp is older than the one currently in use.

A CRL may also revoke CA public keys. Since these are not kept in certificates and do not, therefore, have a securely marked expiry time, they will never be removed from the CRL.

Every CRL is signed with the CA private key that is active in the CAW at the time of signing. The signature includes the key id of the relevant CA public key to check the signature. A client must check that the CA key used to sign an incoming CRL is a CA key and that it has not been revoked. The client must not use the CRL unless these checks are passed.

It is the responsibility of the loading process and the cryptographic verification functions to enforce the revocation policy.

4.6.2.7 Destruction

A certificate is destroyed by deleting it from the collection. This is done by the KMC.

FUJ00098154

4.6.3 DSA CA public key (CAPU)

4.6.3.1 Installation

In normal operation at NR2+, there is no installation process for CA public keys. All platforms that will use public key certificates are installed with a pre-generated stock of CAPU when they are built.

The KMA routinely distributes copies of the current CAPU collection to all CAPU clients for the purpose of integrity assurance. The client must compare the received values with the installed set. If there is any disagreement, the client must raise a security alert.

Platformsthat are being migrated from NR2 to NR2+ receive their stock of CA public keys via the Tivoli software distribution mechanism rather than at manufacture. See section 6.

4.6.3.2 Loading

The loading process will check the current certificate revocation list for the id of the CAPU being loaded. If the CAPU has been revoked, loading will fail and the CAPU will not be available for verification of certificates.

4.6.3.3 Unloading

The key is unloaded using the Layer 7 facilities (e.g., STOR_RemoveKeyxxx). A semaphore is used to lock out the crypto applications while this is being done.

4.6.3.4 Off-line storage

None.

4.6.3.5 Online storage

The stock of CAPU will be stored in a single collection (e.g. individual files in a single directory) on the fixed disk of the key client platform.

4.6.3.6 Revocation

The revocation process will receive certificate revocation lists (CRL) from the KMA delivered via the automatic distribution channel. Each message lists the identifiers of all CAPU which have ever been revoked.

A CRL which revokes a CAPU will be signed with a later CA private key, typically the next CAPR in order. Verification of a new CRL should be carried out with respect to the existing CRL.

4.6.3.7 Destruction

To simplify CAPU checking, old CAPUs are never deleted.

4.6.4 Layer 7 Red Pike Data Encryption Keys on FTMS Gateways

Note that this section does not apply to any keys in PO outlets.

4.6.4.1 Installation

The DEK is delivered on the automatic channel encrypted under a TK value. Installation occurs when the corresponding TK file is delivered via the manual channel. The Cryptographic Key Custodian or Cryptographic Key Handler installs the TK by rebooting the gateway inserting the TK diskettte when

FUJ00098154

prompted. Subsequent data encryption or decryption calls will automatically use the latest DEK that can be decrypted using the available TK.

4.6.4.2 Loading

The DEK is loaded on demand when the FTMS application first attempts to decrypt or encrypt a file. The key store booter arranges to keep the TK loaded into memory at all times so that it is available to decrypt the DEK, which is held encrypted in the Riposte Persistent object store.

4.6.4.3 Unloading

The key is unloaded either by reboot or automatically under control of the KM client agent if a new confidential key capsule containing a new Red Pike key and encrypted under the current TK value arrives on the automatic channel (see section 3.12.2).

4.6.4.4 Off-line storage

Except when the TK key is being loaded, the diskette containing the TK key file will be stored in a physical safe on the same site as the key client platform. Only the Cryptographic Key Custodian and the Cryptographic Key Handlers will have access to this safe.

4.6.4.5 Online storage

The DEK is stored encrypted under TK in the Riposte Persistent Object Store.

4.6.4.6 Revocation

There will be no separate revocation process. Installation of a new key implicitly revokes the previous key by overwriting the configuration details.

4.6.4.7 Destruction

Manual procedures will require the custodian to return the TK file to the Pathway Key Manager for obliteration. There is no automated security-relevant destruction for the DEK.

4.7 PMMC Agent

4.7.1 Post office FEK

4.7.1.1 Installation

The installation process for the FEK will operate first during roll-out to install the first FEK, thus placing the specified parts of filestore under encryption. The process will present the FEK to the TeamWARE Crypto library as the first encryption key.

Subsequently, whenever a replacement FEK is delivered by the interactive channel, the installation process will present the new FEK to a "change key" API in the TeamWARE Crypto library. This will initiate re-encryption of the protected filestore under the new key. The re-encryption is carried out without making an in-clear copy of the protected data on disc.

In both cases the installation process will also send a copy of the latest FEK to be securely stored off-line on the PMMC; see "Off-line storage", below, for details of the storage process and protection. There are two reasons for this.

FUJ00098154

1. The only copy of the installed FEK which is held on a post office workstation is in memory local to the TeamWARE Crypto library. Hence, when the workstation is switched off the working copy is lost. When the machine is next started, the FEK must be re-inserted from some external medium. The PMMC is that medium.
2. The PMMC is also the medium for delivering the FEK to non-gateway counter PCs. The installation process is thus initiating onward distribution of the FEK to any secondary counter workstations in the post office. The protocol by which the KMA tracks the status of this delivery is described in section 3.10.2

The installation process will save the obsolete FEK on the PMMC because it must be loaded at secondary workstations in the process of installing a new FEK.

4.7.1.2 Loading

The loading process will prompt the POM to insert the PMMC and enter the current PIN. It will then retrieve the encrypted copy of the FEK from the PMMC, use PIN to decrypt the FEK, then present it to the appropriate API in the TeamWARE Crypto library.

4.7.1.3 Unloading

The FEK is unloaded when all processes which use the TeamWARE Crypto library have terminated.

4.7.1.4 Off-line storage

The PMMC is used for off-line storage of a copy of the FEK and other keys. This copy must be encrypted to protect it from discovery by an attacker who gains possession of the card. The key used to encrypt the FEK must be one which can be reconstructed only from a secret known to the POM, so that an attacker is unlikely to acquire it. This secret comprises a 64-bit PIN generated on the gateway PC via Layer 7 facilities using local software entropy.

The off-line storage process for FEK will take the following steps:

a) generate a Personal Identity Number (PIN) to be kept by the POM;

b) encrypt the FEK with the PIN;

c) write the encrypted copy onto the PMMC.

The PIN is the secret which the POM holds. It is a 64-bit random binary value which the storage process will generate. For the convenience of the POM, the storage process will map this value to a 15-character alphanumeric string and print it on the post office receipt printer. The POM will be instructed by training and by reminders from the storage process to keep this hard copy out of sight in a safe place.

The complete inventory of keys and key tags held on the PMMC is as follows. These are held encrypted on the card using Red Pike encryption with the PIN as the key using the integrity check specified in section 3.11.

$TK_{CURRENT}$

TK_{OLD}

$DLLKA_{CURRENT}$

$DLLKA_{OLD}$

$FEK_{CURRENT}$

FEK_{OLD}

POK

FUJ00098154

4.7.1.5 Revocation

There is no explicit revocation process. A FEK is implicitly revoked by installation of a new FEK.

4.7.1.6 Destruction

A FEK is destroyed by deleting the encrypted copy from the PMMC and by installing a new FEK in TeamWARE Crypto.

4.8 Non-NT Clients

The non-NT client software at NR2+ is identical with that at NR2. The NR2 design documentation may be consulted for the definitive design of the key management features provided. The rest of this section summarises these features for convenience of reference but is neither complete nor definitive.

4.8.1 DSS ICL VME Systems

The VME platform receives the CAPS key via a manual distribution route in the form of a character string printed on paper. The string represents the new key encrypted under the previously installed key.

4.8.1.1 Installation

Installation is an interactive process which is invoked and operated by the Cryptographic Key Custodian when the CAPS encryption functions are inactive.

The process will prompt the custodian to type in the characters printed on the paper. Using the previously installed key, the process will decrypt the new key and store it in an obfuscated form in the appropriate VME user object node. The process will delete the preceding key from the object node.

A manual procedure will direct the custodian to destroy the paper copy of the new key.

4.8.1.2 Loading

The CAPS encryption functions will load the key directly from the user object node. There will be no separate loading process.

4.8.1.3 Off-line storage

None.

4.8.1.4 Online storage

The obfuscated key is held in a user object node accessible only to the Cryptographic Key Custodian and the CAPS encryption process.

4.8.1.5 Revocation and destruction

The key is revoked by installation of two new keys in succession. The reason for this double change is that a compromised key would enable an attacker to decrypt the next key in sequence, which would be the active key after only a single change. Performing a double change whilst handling both new keys securely breaks this follow-on attack.

FUJ00098154

4.8.2 Host Central Servers

The Host Central Servers are Sequent Dynix platforms hosting the CAS Oracle database. They receive the CAPS key via a manual distribution route in the form of a character string printed on paper. The string represents the new key encrypted under the previously installed key, plus other control information.

4.8.2.1 Installation

Installation is an interactive process which is invoked and operated by the Cryptographic Key Custodian when the CAPS encryption functions are inactive.

The process will prompt the custodian to type in the characters printed on the paper. Using the previously installed key, the process will decrypt the new key and store it in an obfuscated form in filestore protected by access controls (see "Online storage", below).

The installation process will operate a two-position key ring: it will place the newly installed key in the "current" position, the preceding key in the "expiring" position, and will delete the key previously in the "expiring" position.

The process will mark the key in the "expiring" position with an expiry time taken from the control information carried with the new key. After that expiry time has elapsed, loading functions (see below) will refuse to load the key.

A manual procedure will direct the custodian to destroy the paper copy of the new key.

4.8.2.2 Loading

The CAPS decryption functions will load keys directly from filestore. There will be no separate loading process. The decryption functions must load the "current" key, and must load the "expiring" key only if the expiry time has not elapsed.

4.8.2.3 Off-line storage

None.

4.8.2.4 Online storage

The obfuscated key is held in filestore accessible only to the Cryptographic Key Custodian and the CAPS encryption process.

4.8.2.5 Revocation

The key is revoked by installation of two new keys in succession. The reason for this double change is that a compromised key would enable an attacker to decrypt the next key in sequence, which would be the active key after only a single change. Performing a double change whilst handling both new keys securely breaks this follow-on attack. Both keys used in the double change will carry control information indicating immediate expiry of the preceding key.

4.8.2.6 Destruction

The online copy of a key is deleted when the second successive key is installed.

FUJ00098154

5. SYSTEM QUALITIES

5.1 Performance

5.1.1 Overall

- NR2+ roll-out key supply rate: 300 offices/week
- NR2-NR2+ migration key supply rate: 400 offices/week (in addition to roll-out key supply rate)
- Roll-out key response time: <5 minutes
- Routine key change (single key, single client, worst case by manual distribution, assuming prompt action by key custodian): 49 hours. - includes key generation, certification, distribution, and installation. *[Worst case assumptions: 1 hour certification + 24-hour courier + 24 hours before the next operational window for installation]*.
- Emergency key change (single key, single client, worst case by manual distribution, assuming immediate action by key custodian): 9 hours.*[Worst case assumptions: 1 hour certification + 6-hour courier + 2 hours installation time]*
- Rate of change (over and above roll-out supply): 20,000 outlets and 40,000 counters every 2 years in the steady state.
- The time taken to revoke a public key certificate is policy-dependent; a trade-off between cost of compromise and cost of discarding legitimate messages (those "in the pipeline" which were signed by the revoked private key). The system will be capable of revoking a public key certificate
  (a) at a post office counter within 1 hour, assuming that the WAN connection is available, the gateway workstation is running, the counter workstation is running, the postmaster log-on is complete and assuming that there is no backlog of business data that must be processed before the key material begins to arrive, (i.e., if the counter is off-line, the revocation will be complete within 1 hour of its coming back on-line, not allowing for any business messages that are queued).
  (b) at a campus or remote FTMS client within 1 hour assuming that the Riposte Message Service is available.
- Time to revoke CA public key: same capability as for PKCs (above).
- Rate of initiation of revocation (broadcast): all post offices and Tivoli-connected data centre platforms within 5 minutes.

Note: In the case of changes to the keys held on the PMMC, the key custodian is the POM; the worst case figures above assume that POM acts when prompted. If the POM does not cooperate promptly, the time to carry out the key change is outside the control of the KM system (and indeed outside the control of ICL Pathway Ltd.).

The registry usage of NR2 counter builds is a source of concern and there is a Pathway policy for NR2+ software to use the registry sparingly. No firm budgets for registry usage have been provided. Detailed designs for all software components that use the registry should provide an estimate of their registry usage (number of entries, total size of data stored in registry).

5.1.2 KMA

- Key Manager and Help Desk GUI response time (command acknowledgements, query results): 2 seconds

FUJ00098154

Sizing of a host system suitable to support the KMS database is difficult before the applications are written. However a 2 processor system with modern processors (eg 350MHz Pentium II) would provide more processing power than the SE70 used for the PAS/CMS system.

The number of disks required will depend heavily on how well the application data is cached. This is hard to determine before the applications are written and tuned. For this reason it is recommended that a "design feedback" phase is included in the KMS plans which can be used to tune the database (e.g. adding additional indexes) and to determine the number of disks required.

5.1.3 Key generators

- DSA key pairs will be generated at better than 1000 bits (key length) / second (e.g. ~1 second for a 1024-bit key).
- Red Pike keys will be generated at better than 1/second.

Key generation will require specific performance testing.

5.1.4 CAW

- Rate of throughput: 500 key certifications / hour, sustainable over 40 hours (20,000 certifications).

5.1.5 Post office client processes

The overall requirement on KM and the desk-top application code is understood to be as follows:

- Time from start of POM log-on to start of Riposte desktop at roll-out, including installation of all requisite keys for commencement of business (assuming ISDN connectivity and KM Controller performance OK): <10 minutes

The performance at start-up of the desk-top applications is outside the scope of this document. The time spent in KM code under these circumstances (including ISDN and KM Controller availability within 30 seconds; but not including time spent waiting for operator intervention) should be at most 1 minute.

For normal business services that use cryptographic protection (e.g. payment of benefit at a Post Office) KMS should have only a minimal impact. The main issue is to ensure that the functionality that KMS adds to these functions does not impact performance.

The only performance work that needs to be done for these areas is by regression testing. The counter transactions have had witnessed benchmarks done for the purposes of calculating SLAs for transaction times at R1c and NR2. It is assumed this will also be true at R2+ and this should be the route for testing the effect of KMS.

It has been agreed with Customer Services (Jan Ambrose) that the target for counter transactions should be that KMS adds no more that 0.1 seconds to the current execution time of the transaction.

The other function on the Post Office that uses KMS is TeamCrypto which is used to encrypt the Riposte message store. This needs to be specifically tested to ensure that the time taken to re-encrypt the message store when the key is changed and the impact that this has on the counter system is acceptable. It is suggested that this is tested as part of the large outlet testing at Feltham.

KMS also puts an additional load on the Gateway PC due to adding of a new message port. The performance impact of this is believed to be small (there are already several message ports on the Gateway PC) but this needs to be explicitly tested.

FUJ00098154

5.1.6 Data Centre Operations

There are several data centre operations that use encryption (e.g. file transfer on the TIP link). The effect of using KMS, as opposed to a fixed key, should be small. These services need to be regression tested.

The target for these should be that KMS increases the time for the operations by less than 1%. If this target cannot be met for a given operation then whether or not that is acceptable will need to be specifically sized.

In order to support KMS, Riposte will be installed on data centre systems that use it. This should have minimal impact providing that the systems have sufficient memory to support the Riposte service. At least128Mbytes of memory is recommended for these systems with an absolute minimum of 64Mbytes.

5.1.7 Automatic Channel

KMS uses Riposte to distribute most of the data for KMS. Although Riposte will be used for distribution to non Post Offices this is ignored in the discussions below since it is very small compared to the data for 19,500 Post Offices.

Each Post Office has a number of items of data distributed via Riposte. These keys are detailed in the table below together with their data size, number of entries (for revocation lists) and the associated message size in Riposte (a fixed overhead of 200 bytes per message is assumed).

Key Name	Bytes Per Entry	Num of Entries	Message Size (Bytes)	Riposte Messages +	Notes
APPR	720	2	1,640	1	APS Private Key at the Outlet
GDK	250	1	450	1	
DLLKA	250	1	450	1	
CAPU	650	10	6,700	4	Public Key Certificate
CRL	20	10*	400	1	Revocation List
PAPU	650	4	2,800	2	Payment Authorisation Public Key
VPN CRL	63	200*	13,050	7	Global Revocation List for VPN (250 byte overhead)
Total			**25,490**	**17**	

Legend: * - Assumed worst case; + -A maximum of 2Kbytes per message

The total size of these items is around 25 Kbytes. This is small compared to the reference data in a Post Office (around 11Mbytes) and can therefore be safely ignored both at the outlet and on the correspondence servers.

The number of messages is also small. With 2Kbytes of data for a BLOB (binary large object) being held per message, there are around 17 messages per Post Office. This again is very small when compared to reference data and can be ignored.

Note: The VPN CRL is a global list for all 19,500 Post Offices. This could cause an issue if the number of entries in the list significantly exceed the estimated 200 above.

FUJ00098154

5.1.8 KMS Loading

The interactive loader agent for KMS (to load messages from the KMS host system into Riposte) should be capable of loading at least 60 messages per second (this is based on the performance obtained from other similar agents). Higher loading rates than this would be possible using multiple agents.

If all the KMS data defined above for 19,500 outlets had to be loaded this would take about 1.5 hours. This is the absolute worst case and will not actually occur – since this is acceptable then other scenarios are not considered further.

As the KMS data is held in persistent objects in Riposte there could be significant index activity while the messages are being loaded. With only a single interactive agent this should not impact other operations on the correspondence server but this will need to be confirmed through testing.

The KMS host system must also be capable of supporting the reading of 60 messages per second by the agents.

Since the time to load key material into Riposte is short there is no need to limit the number of key changes that can happen per night (as far as loading is concerned).

The performance KMS loading needs to be explicitly tested. A representative Riposte message store will have to be used for this (e.g. it must have significant persistent objects to ensure that the persistent object index is not cached).

5.1.9 Interactive Channel

The key factor in determining the effect of the interactive channel on the network and key management centre is the number of concurrent sessions. This is difficult to estimate as it depends on the frequency of PMMC/PIN recovery operations (as well as the more predictable roll-out and migration rates). This will be very peaky and early mornings when the Post Master needs to log onto a counter position will be the busiest time.

There are two areas where this could cause a problem. The first is the number of concurrent TCP/IP connections at the Key Management Centre which may exceed the capabilities of the box. The second is the number of ISDN routers ports that will be occupied with the connections as this could impact normal service if it gets significant.

Since the peaks cannot be determined and because a large number of concurrent connections could have a detrimental effect, it is recommended that the number of concurrent connections is limited with additional connections beyond that being refused and the ISDN connection being shut down. The Post Master could be asked to try again later if this occurs.

The number of allowed connections should be tuneable. An initial value of 50 is reasonable. The maximum number of concurrent connections that the Key Management Centre can support will need to be explicitly tested.

5.2 Availability and resilience

Defining the overall availability and resilience attributes of the KM clients is outside the scope of this design. The KMS is only responsible for enabling recovery of missing key material on the clients.

A study to analyse the overall resilience has been undertaken taking the first baseline for this design document as its starting point. The following points are noted at this stage:

- Resilience of the Riposte messaging system is known to be high; therefore no additional measures will be included in the Key Management system to cover communication faults.

FUJ00098154

- The Horizon Help Desk must achieve certain SLAs for restoring PO counter PCs without assuming the availability of the ISDN networks.
- The KMA and its database will be mirrored between the main and backup sites using EMC hardware replication of the filestore. The architecture is very similar to that used for the host servers. During normal operation, the mirror is up and running its operating system, but the DBMS and KMA NT services are not running. Fail-over is via operator intervention following similar procedures to those used for fail-over of the host servers.
- Clients using the interactive distribution channel will be configured with the IP addresses of the KMA servers and the standbys and will access whichever IP address permits a connection. (The LAN in each campus is dual, so each server at each campus has 2 IP addresses).
- PO gateway PCs will additionally be configured with the IP of two VPN recovery servers, one at each campus.
- The Key Manager's primary workstation will be at FEL01, normally available continuously.
- There will be secondary workstations in physically secure areas at FEL01 and at either Wigan or Bootle. In the event of loss of the primary workstation, the secondary can be brought into use at no more than 4 hours notice at any time and will then be available continuously until a new primary is installed.
- With the exception of the CAW (which must not be network connected), all processes in the Key Management Centre will be monitored by Tivoli, which will raise appropriate alarms if a process stops running.
- The CAW at FEL01 will normally be available continuously.
- A secondary CAW will be available at the same site as the Key Manager's secondary workstation whenever the secondary workstation is in use.
- In the event of failure of either KM workstation or either CAW the system will be recoverable or replaceable in less than 1 hour.

5.3 Usability

- The KMA will present a GUI at the KM workstation, supporting the inspection of all management data held in the KM database, and the initiation, control and tracking of all key management operations except certification.
- The KMA will support unattended batch operations, subject to any relevant security restrictions.
- The CAW will present a GUI providing control of the certification process.
- The certification process will support *attended* batch operations, subject to any relevant security restrictions.
- The key management system will operate unattended and according to constraints specified by the Key Manager when issuing post office keys at roll-out or subsequently changing them.
- The KMA will prompt the Key Manager some time (configurable) before a key delivered via the manual channel needs to be changed. After the due time for the change, the KMA will "nag" the Key Manager daily until completion of the operation is confirmed.
- The system will allow the Key Manager to change any key at any time regardless of schedule.

FUJ00098154

- User interfaces will check all input for validity and consistency.
- Other than in PO outlets, key management procedures will not require the involvement or understanding of anyone other than (i) the Pathway Key Manager, (ii) Cryptographic Key Custodians, (iii) Cryptographic Key Handlers.
- In a PO outlet, the POM will act as key custodian; the user interface presented to the POM will guide the POM through all key management processes, without requiring special training or documentation.
- The system will require Cryptographic Key Handlers to take action (e.g. inserting key disks) only in the most infrequent circumstances consistent with security of the material.
- During key change, the system will not demand action of a Cryptographic Key Custodian before that person has received all the key material.
- The existing PoLo interface will not change except as required by new functionality.
- The PoLo interface will communicate with either the engineer or the POM only in terms which are familiar to those roles - no jargon.
- Routine key management operations will not require interruption of business at post offices during opening hours.

5.4 Security

Defining the overall security attributes of the KM clients is outside the scope of this design. The KMS design is only responsible for enforcing appropriate security policies on the KM Controller platforms.

The following comments apply to the KM Controller platforms and their network connections.

- All parts of the KM database which contain private or symmetric key values will be encrypted for security.
- The Pathway LAN and the links between the campuses are presumed insecure for key material.
- The links between FEL01 and each campus are presumed to be secured by Rambutan encryption.
- The link between the KMA and the Tivoli Workstations at the help desks are to be secured by Rambutan encryption.
- The user community of the KMC is assumed to be as described in section 5 of [KMREQ]. The design of all KMC components must be compatible with access control policies implementing the requirements of [ACP] and section 5 of [KMREQ].
- The design of the CAW presupposes strong physical security to avoid compromise of the CA private key. The magnetic media containing a CA private key must never be loaded into a platform with a network card.
- The security event management policies of [KMREQ] are followed in all KM software.
- Audit of the KMA server and the KMA workstations is via NT event logging mechanisms following agreed Pathway and Crypto team policies.
- Audit procedures for the CAW based on standard NT systems management facilities will be defined as part of the CAW design and implementation.

FUJ00098154

- User authentication for all KMC platforms is managed by a systems manager working under the direct supervision of the Pathway Key Manager.

5.5 Manageability

See section 7.

5.6 Potential for change

While there is a potential for change as described in this section, the current design and implementation plans for KMS do not provide any specific software support for operational staff to introduce new protection domains. To introduce a new domain, the development team will need to be involved in upgrading database and metadata, completing the impact analysis across all affected platforms and introducing required updates through PCMS.

The current design and implementation does enable certain clients to be introduced in certain existing protection domains without the involvement of the development team. The cases supported are: PO counter PCs, Agent Servers, AP clients.

- Within any capacity limits set by the available data stores and communication channels, the system will accommodate the introduction of additional DSA and Red Pike protection domains beyond those currently specified in requirements and architecture.

- The system is designed such that the cost of introducing an additional protection domain to existing key clients is expected to be the sum of the following costs:
  (i) adding the management data for the new protection domain to the existing KMA database schema;
  (ii) acquiring and installing base key material from CESG;
  (iii) designing and implementing key client processes for the new protection domain;
  (iv) extending manual procedures to accommodate the new key material.

- The cost of introducing a new key client to an existing protection domain is expected to be the sum of the following costs:
  (i) adding management information for the new client to the existing KMA database schema;
  (ii) designing and implementing distribution channels to the new client;
  (iii) porting key client processes to the new client.

- The system will be capable of handling arbitrary cryptographic material for specific post office counter applications to the following extent:
  (a) unstructured storage of the material by the KMA under any protection afforded by the DBMS;
  (b) unstructured distribution of the material to all post offices under common protection with all other post office key material;
  (c) application-specific installation of the material at all post offices.
  The costs of introducing each instance of application-specific cryptographic material will be assessed case by case.

- The system may be capable of accommodating new key material of types other than DSA or Red Pike. The cost of introducing such material will be at least the sum of the following costs:
  (i) design and implementation of a key generator;
  (ii) redesign of the KMA database schema to accommodate management information for the new key type, with consequent regression testing of the whole KM system;
  (iii) introduction of the management information to the amended schema;
  (iv) design and implementation of key client processes for the new key material.
  Further costs may also apply:

FUJ00098154

(v) acquisition of approval for key processes from CESG;
(vi) acquisition of base key material from CESG;
(vii) redesign and revision of key distribution channels to accommodate new type of material.

- The system will not prohibit an upgrade which replaces Red Pike with a stronger algorithm using a longer key for protection of key material within KMS and its clients. In the absence of information about the new algorithm, it is not possible to estimate the likely costs.
- The cost of adapting the KM client software to run in the absence of Riposte to support verification of in-line signatures, e.g, for SI, should amount to no more than reconfiguration and repackaging.
- Where data structures are passed between software components running on different platforms, the data format will include a version identifier. The version identifier should be encoded so that it may readily be extracted from the data stream without knowledge of the platform that produced the data. Thus, where appropriate, Intel-specific data layouts are permissible provided the possibility of an upgrade to support other architectures is catered for.

5.7 Year 2000

All code and keys produced will be "year 2000" compliant. Wherever underlying software supports it, UTC dates should be used following Pathway's standards. Relevant Pathway policies include the following statements:

1. All externally procured products are supported by unequivocal vendor compliance statements.
2. All date data items will conform with the Pathway Design standard of using a full 4 digit year.
3. All subcontracted work will require compliance to the relevant Pathway standards of design, development and testing.

FUJ00098154

6. MIGRATION

The detailed implementation of migration will be described in a separate document [KMMIG]. This section outlines the scope and major considerations.

6.1 Scope

When live deployment of NR2+ begins, all data centre platforms and post office counters will have been upgraded to NR2. Therefore, there is no requirement for migration from release 1c.

Migration must bring all NR2 platforms up to NR2+ key management. This includes the adoption of existing (NR2) keys under the NR2+ management regime, as well as introducing keys which are new for NR2+.

At the start of migration some 8,000 post offices are expected to be in live operation with R2 software, and roll-out will be continuing at the rate of 300 offices per week. The process of migration must therefore be designed to proceed in parallel with roll-out. Old stock NR2 and new stock NR2+ platforms may well be rolling out concurrently, and the KM design must cater for this. Note that migration may well proceed in several phases (see section 1.1).

6.2 Business impact

Migration must cause the minimum possible discontinuity to business at any single post office. The acceptable out-time is yet to be determined.

In mitigating impact, the design for migration must pay attention to the use of communication bandwidth as well as any installation and initialisation processing.

Migration cannot be implemented as a "big bang" process, in which all platforms are expected to change version at the same time. It will be possible to continue managing keys for NR2 platforms (centre or post office) using the NR2 mechanisms without degradation whilst the NR2+ KM system manages the keys for platforms which have been upgraded to NR2+.

6.3 Platform design impact

Migration will certainly entail the introduction of new code on some platforms, notably the post office counters. It may also entail upgrades to existing cryptographic functions (e.g. signature verification routines must be revised to handle public key certificates, rather than plain public keys).

6.4 Application impact

Where existing cryptographic functions must be revised (see 6.3), the intention is to keep the API unchanged. Where the functions are packaged in DLLs, there will be no impact on calling applications. If any applications were to be statically linked to cryptographic functions (we know of none currently), they would need to be re-linked and re-installed.

New functions for key management are not expected to be visible to R2 applications.

FUJ00098154

6.5 Upgrading key management software

6.5.1 Method

Unlike the migration to R2, R2 post-offices will be not be migrated to NR2+ by swap-out. Instead, Tivoli remote software installation will be used to

(a) upgrade any existing key management modules which are carried forward to NR2+,

(b) install new key management modules, and

(c) remove obsolete key management modules.

6.5.2 Ordering

The order in which key management software will be migrated should follow Pathway's policies and requirements. In particular, the central platforms and software should be migrated first.

6.5.3 Compatibility

Central NR2+ key management software will need to be backwards compatible with NR2 operations.

6.5.4 Down time

In keeping with normal remote configuration management policy, these revisions will be made during periods when minimum disruption will be caused to the business of the affected platform.

6.6 Upgrading keys

6.6.1 Introducing the CA public key stock

The key installation processes on upgraded platforms will operate a once-only rule allowing the stock of CA public keys to be installed from an online distribution.

6.6.2 Adopting existing keys

Some NR2 keys will simply continue in use when a platform is upgraded to NRN (e.g. PAPR, SIPR). In these cases it will only be necessary for the Key Management Centre - specifically the KMA - to record the details of these keys for future management operations.

6.6.3 Revising existing keys

Some NR2 keys will continue in use in an altered form; e.g. the public keys, which must all be certified for NR2+. It is anticipated that the NR2 forms will be replaced with NR2+ forms by means of Tivoli remote software installation at the same time that any affected cryptographic functions are upgraded by the same means.

6.6.4 Introducing new protection domains

The following protection domains will be introduced to certain NR2 platforms at NR2+. This means that in the process of upgrading platforms the cryptographic functions for these domains will be installed. They will then need to receive keys from the KMA to enable operation.

AP	Private keys on the post office workstations; PKC at the AP harvester.

FUJ00098154

L&G Code	Symmetric key at the post office workstations.
L&G Enabling	Supplied key at the post office workstations.
Utimaco VPN	Own asymmetric key set and server PKC at the post office workstations. (The servers themselves are new platforms for NR2+ and are not, therefore, migrating.)

As each platform is upgraded the KMA database will be updated to reflect its new status and the necessary keys will be scheduled for distribution. In the case of PO outlets, the upgraded PO will appear to the KMA in much the same way as a newly installed PO and the database will be updated via the feed of PO configuration data (see Figure 8). In the case of other platforms, the update will be done via manual intervention by the KMA database administrator (see "KMA Design" [KMAPDES]).

6.6.5 Compatibility

Adopted keys remain unchanged, and so raise no compatibility issues.

Keys in newly introduced protection domains do not need to be backward compatible with any previous functions.

Revised keys might be incompatible with the R2 functions which used them. This will certainly be the case with public key certificates, which will be incompatible with R2 signature verification functions.

TSC will deliver revised compatible cryptographic functions where necessary. These must be installed concurrently with the revised key forms.

6.6.6 Ordering

Separate protection domains (PA, SI, CMS, etc.) are managed largely independently. Subject to further study, it is not expected that there will be technical constraints on the order in which protection domains migrate from NR2 to NR2+. This subject will be treated in greater depth in the detailed implementation document for migration.

6.7 Changing key management operations

Because migration will be incremental, it will be possible to continue managing a diminishing community of key clients by NR2 techniques while concurrently operating the NR2+ management regime for upgraded key clients. This means that a subset of keys (NR2 keys in operation during the period of migration) will be concurrently managed by both systems. Synchronisation between the two systems during key changes will be a salient issue.

FUJ00098154

7. SYSTEM MANAGEMENT

Specification of the system management of the KM clients is outside the scope of this design. The KM software at each client will be managed via the same mechanisms as are used for other software on that client.

- With the exception of the CAW, the Key Management Centre platforms are managed by Tivoli remote system management.
- Software updates for the Key Management Centre (except CAW) are installed remotely by Tivoli software distribution.
- All automated key management processes on NT platforms (including the CAW) log application events which assist in the detection and diagnosis of faults, within the constraints of applicable policies.
- The CAW is managed by use of standard NT systems management interfaces under the direct supervision of the Pathway Key Manager.
- See section 5.2 for a discussion of the management of fail-over for the KMA server.
- The policy for NT event logging to be applied on all Tivoli-managed NT platforms is defined in [LOGREQ]. The details of the implementation of this policy will be passed to ICL Outsourcing who can then put in place Tivoli event management scripts to gather and process the NT event records.

FUJ00098154

8. TESTING

Specification of test strategies for each component of the KM system will be defined in their detailed design documents. Some general suggestions and constraints follow:

- A test schedule for each protection domain may be derived from the abstract KM data flow model shown in Figure 7. Each bubble in this diagram represents an individually testable component. Integration testing within a protection domain will generally best be done in the following order:
    1. Client
    2. Client + distribution channel
    3. Client + distribution channel + monitoring channel
    4. Client + distribution channel + monitoring channel + KMC
- Test rigs to simulate application software will be required to allow testing of the clients.
- To test the key change protocols extensive accelerated life-cycle testing will be required during integration testing.
- Testing must not be carried out using live key material. Conversely, test key material should not be used in the live system. Manual procedures will be developed to enforce this separation at all stages of the development life cycle.
- It is a design constraint that adequate testing at all levels should be possible without using live key material.

FUJ00098154

9. DEPENDENCIES

The success of this design depends on external parties to provide or to assist with the following:

- Provision of the feed of PO configuration data
- Utimaco software enhancements
- Riposte on Campus NT clients
- Boot server/autoconfig design integration
- Help desk design integration
- Help desk user cooperation during detailed requirements analysis
- KMA user cooperation during detailed requirements analysis
- VPN physical architecture
- Statistics on the volatility of the client inventory.
- API for driving MemoView.

FUJ00098154

10. ASSUMPTIONS AND RISKS

10.1 Assumptions

The following assumptions have been made concerning this development. The risk of these assumptions not being valid is discussed below:

ASS	DESCRIPTION
A1	It must be possible to derive from Pathway systems management policy a latency period appropriate for the SI keys (see section 2.6.3.2).
A2	The P, Q and G values that parametrise DSA signing can be pre-delivered as part of the static configuration of the clients that need them (see section 2.7.4)
A3	The P and B values that parametrise the Diffie-Hellman algorithm can be pre-delivered as part of the static configuration of the clients that need them (see section 2.7.2)
A4	The physical security which protects the CAPS key delivery is enforced as strongly for NR2+ as for earlier releases.
A5	The software issue process must use the SI signature on software packages delivered to PO outlets to protect software issue via Tivoli against tampering. Adequate tamper-proofing must be provided, e.g., via access control and procedures, for all non-outlet platforms.
A6	An API will be available at the KMA for driving MemoView.
A7	A data feed of roll-out data will be supplied in time for keys and PKCs to be generated for the rolled-out platform.
A8	Utimaco configuration will be enhanced or customised to support migration from NR2 to NR2+
A9	A local Riposte service will be available on Campus NT clients (but not the KMA itself)
A10	The boot-server and autoconfig process will be as described in 3.9.2
A11	The delivery of initial POKs to the boot server described in section 4.5.3 is technically feasible and acceptably secure.
A12	The KM System is only responsible for generating and distributing VPN keys; it is not responsible for controlling activation of VPN on outlets or servers.
A13	TeamWare Crypto or a similar product is available for swap-file encryption on all NT platforms that use confidential keys.
A14	As implied by [KMREQ], the extensibility required of the AP Clients protection domain involves only addition of new gateways with identical software builds and cryptographic keys, not addition of new DSA signing keys to identify parties other than ICL Pathway.
A15	It is assumed that a range of Riposte group ids can be allocated, distinct from those used for any other purpose in Pathway, and sufficient for all the campus and remote FTMS gateway KM clients that use the automatic channel.
A16	There is no requirement for the NR2+ Audit sytem to take on any KMS data other than

FUJ00098154

the existing mechanism for handling Security events. In particular the Audit requirements as expressed in HADIS are aimed at Business applications rather than Infrastructure applications.

The only Auditing requirements on KMS are therefore, those specified in [SFS]with respect to Security related events. These are generated as NT Events, which are picked up by the existing Tivoli mechanism to forward them to the Security Audit system.

Notes: if assumption A5 does not hold, then the KM System can offer no guarantees of cryptographic protection on any platform managed via Tivoli.

10.2 Risks

The following risks are associated with this development:

RISK	SEVERITY	DESCRIPTION
R1	MED	All assumptions (see section 10.1) are valid
R2	HIGH	Key management services are required for an application whose data flows are incompatible with the policies of section 2.6.3.2.
R3	LOW	Key management services will be required for a future business application whose data flows are incompatible with the policies of section 2.6.3.2.
R4	MED	Detailed design work will reveal a fundamental flaw in the migration strategy.
R5	HIGH	The VPN architecture will change to requiring bespoke KM code to police recovery connections
R6	MED	A synchronisation problem (e.g., in the VPN servers or in the SI signing servers) will arise that cannot be handled by manual procedures.