社団法人 情報処理学会 研究報告
IPSJ SIG Technical Reports

2007－ＱＡＩ－25（10）
2007／10／26

P2P分散ファイル共有基盤を活用した自己暗号化法によるプライベートストレージ

遠藤 大礎† 川原 圭博† 浅見 徹†

† 東京大学大学院 〒113-8656 東京都文京区本郷7-3-1
E-mail: †{endo,kawahara,asami}@akg.t.u-tokyo.ac.jp

あらまし 我々はこれまでモバイル端末の内部情報保護のために自己暗号化法による分散ストレージを提案してきた．しかしユーザがその場でそこにある端末を使うユビキタスコンピューティング環境では，モバイル端末はスタンドアロンでの利用だけでなく，情報を持ち運ぶセキュアなストレージとしての機能も求められる．しかし処理能力の大きい端末との協調動作を求められる場合，モバイル端末における処理が大きなボトルネックになりうる．そこで本稿では端末とネットワークストレージの役割分担を見直し，端末における処理負荷を軽減することを目指す．サーバがデータを自己暗号化法によりs個の分散データに暗号化し，すべての分散データがネットワークストレージ内に存在する場合，サーバに不正侵入されてしまうとストレージ内のデータの数をnとしたとき ${}_nC_s$ 通りの試行でリモートデータを復号されてしまう問題が生じる．そこで分散データを置くネットワークストレージをDHTによる広域ストレージとし，ほかのユーザとあえてインフラを共用する．これによりmユーザで分散データをシェアすることで復号への試行が ${}_{(m*n)}C_s$ 必要になり，特定の分散データの組を見つけるのが困難になる．

キーワード 分散ストレージ，モバイル端末，P2P，自己暗号化法

A Private Storage System Using Self-Encryption Scheme over P2P Distributed File Sharing Infrastructure

Hiroki ENDO†, Yoshihiro KAWAHARA†, and Tohru ASAMI†

† The University of Tokyo Hongo 7-3-1, Bunkyo-ku, Tokyo, 113-8656 Japan
E-mail: †{endo,kawahara,asami}@akg.t.u-tokyo.ac.jp

Abstract We have proposed a distributed storage system based on self encryption scheme for the purpose of protecting private information stored in mobile handsets. In a ubiquitous computing environment where people use public computers installed everywhere, mobile handsets, which users always carry, are expected to act as secure file storages to carry private information. However, if the mobile handsets are combined with PCs, the limited computation capability of a mobile handset can slow down the performance of the PCs. In this paper, we review the role of the mobile handsets and the network storage to relax the load of mobile handsets as well as to balance the utility and security of the system. When we allow network servers to split a plain text file into s cipher text files using self encryption scheme, a malicious intruder of the network storage server can decrypt the original text with only cCs trials. To avoid this problem, we share a network DHT based storage infrastructure with other m users. With this modification, the intruder needs (m*n)Cs trials to decrypt a single file, which is quite hard when m is large enough.

Key words Distributed Storage, Mobile Handsets, P2P, Self-Encryption

1. はじめに

ノートPC，携帯電話やPHSに代表されるモバイル端末は常に起動した状態で携帯できることから，所有者の数がますます増加している．さらに，ユーザとコンピュータの関係性は，大型計算機を複数ユーザで共有する "many-to-one" モデルから1人のユーザが一つのコンピュータを用いる "one-to-one" モデル，1人のユーザが複数のコンピュータを用いる "one-to-many" モデ

ルを経て，1人のユーザが一つのホームターミナルを中心に複数のコンピュータを場合によって使い分ける “one-to-one&many” モデルへと移行していると考えることができる．また現在では，モバイル端末の高機能化が進み，ユーザの連絡先やメール，予定表などの個人情報のほか，今までは企業内で物理的にセキュリティを保障された端末で扱われていた業務情報などの重要な情報資産も保存し持ち歩くようになってきた．しかしながらタスクによって1つの端末を選択し利用しているため，端末ごとにそれぞれ異なる情報が蓄積されており，PC，PDA，携帯電話それぞれにデータが分散しがちであるという問題がある．このような問題を解決するためには，全ての情報をネットワークストレージに保存する方法や，U3 technology [1] や，Personal Server [2] などの物理的に情報を携帯する方法がある．情報をネットワーク上のストレージに依存する場合は，現在では Google の Gmail や Microsoft の Hotmail など，企業により大きな容量のネットワーク上のストレージが提供されており，個人ユースを中心に盛んに用いられ始めている．しかし個人情報を含むデータを企業に預けることを懸念する人もいる．また企業ユースにおいては業務情報は，一企業に対する信頼のみに基づき扱うことは望ましくないという考え方もある．一方，情報を携帯する方式ではデータの紛失や盗難の際十分な時間をかけて攻撃されることを考慮したセキュリティが必要になる．

我々はこれまでモバイル端末の内部情報保護を目的とし，自己暗号化法によりモバイル端末のストレージとネットワークストレージに情報を分散配置するストレージシステムを検討してきた．自己暗号化法とは暗号鍵を暗号化対象ファイルから自動的に生成し，内部処理のみに使用する手法であり，セキュアな鍵管理ができる．またユーザの状況によりネットワークストレージにアップロードする分散データのサイズ，鍵の生成や暗号化アルゴリズムをさまざまな方式から選択し，限られたリソースを有効に活用することも可能である．この方式では情報を全てネットワークストレージに保存する場合と比較して通信量を削減することができ，さらに全分散データがないと復号が不可能であることから，端末の紛失や，ネットワークストレージからの情報流出に対しての安全性も高い．

しかしながら，ユーザがその場でそこにある端末を使うユビキタスコンピューティング環境では，モバイル端末はスタンドアロンでの利用だけでなく，情報を持ち運ぶセキュアなストレージとしての機能も求められる．処理能力の大きい端末との協調動作を求められる場合には，無線 LAN での約 20Mbps の通信を仮定すると，本手法はモバイル端末における 1Mbyte あたりの暗号化・復号処理に約 4 秒かかり，ボトルネックとなりうる [3]．このため “one-to-one&many” モデルの環境を想定すると計算能力や通信帯域などが劣るモバイル端末での処理負荷はできるだけ小さいことが望ましい．

そこで本稿では端末における処理をさらに単純化し，ネットワークに保存するリモートデータの暗号化を端末内でなく，ネットワークサーバで行うことにより処理を軽減する．そのため端末はリモートデータを平文のままネットワークサーバに SSL/TLS などのプロトコルを用いて送信する．本方式では M_R の暗号化処理を省略し，暗号化・復号処理はおよそ半分の時間ですむことから 2.5Mbyte のデータの暗号化，アップロードを仮定すると，1.25Mbyte の通信に 0.5[s]，暗号化処理に 10[s]，合計 10.5[s] かかったものが 5.5[s] に短縮されることになり，この効果は大きい．このとき，ネットワークサーバがリモートデータを自己暗号化法により s 個の分散データに暗号化するが，すべての分散データが 1 つのネットワークストレージ内に存在する場合，サーバが不法侵入されてしまうとストレージ内のデータの数 n により ${}_nC_s$ 通りの試行でリモートデータを復号されてしまう．従って本稿では，分散データを置くネットワーク上のストレージを DHT による広域ストレージとし，ほかのユーザとあえてインフラを共用する．そうして m 人のユーザで暗号文をシェアすることにより，復号への試行が ${}_nC_s$ 通りから ${}_{(m*n)}C_s$ 通りになり，特定の分散データの組を見つけるのが困難になる．また DHT によりネットワークを構成することで，特定の企業や組織からは独立した管理になるほか，ファイルの負荷分散や耐障害性という特性も得られる．

分散データの復号のための対応づけについては，“one-to-one&many” モデルでの運用を想定し，ユーザが常に持ち歩くモバイル端末のみが，あるローカルデータに対応するリモートデータの分散データの DHT における Key を保持するものとする．ユーザの所持するモバイル端末がなければ，分散データの組からリモートデータは復号されず，安全性が保たれる．またこれらの対応付けを含めた 2 つのストレージにおける Read/Write 処理と端末内における暗号化，復号をユーザに対して隠蔽し自動的に処理するファイルシステムについて述べる

まず 2. 節では，我々が提案している自己暗号化法の端末内およびネットワークサーバ内におけるファイルの暗号化，復号などの流れについて示す．3. 節ではネットワークストレージを中央サーバの存在しない Pure-P2P ネットワークにより構築するにあたり負荷分散性，耐障害性，拡張性のために用いる DHT について述べる 4. 節では，自己暗号化法の暗号化・復号プロセスの自動化に必要となるモバイル端末の内部ストレージとネットワークストレージにおける分散データをファイルシステムが把握するための構成を述べる．5. 節では，“one-to-one&many” モデルにおいて，モバイル端末のストレージとその他のデバイス連携により自己暗号化法によるセキュアなプライベートストレージを利用する流れについて述べる．

2. 自己暗号化法による分散ストレージ

自己暗号化法は，暗号化対象ファイル M を s 個のデータ $m_1, m_2, \cdots, m_s$ に分割し，それぞれのデータから生成した鍵を用いて自動的に暗号化を施し，暗号化された分散データ $c_1, c_2, \cdots, c_s$ を生成する．この分散データが s 個全て揃わないと元ファイル M の復号は不可能である．

2.1 システム構成モデル

携帯電話網や，SSL/TLS などによりデータの盗聴や改ざん，なりすましが防止されている通信路の存在を前提に，機密ファイルを暗号化し，ローカルストレージとネットワークストレージ，バックアップストレージに分散配置するシステムの構成を

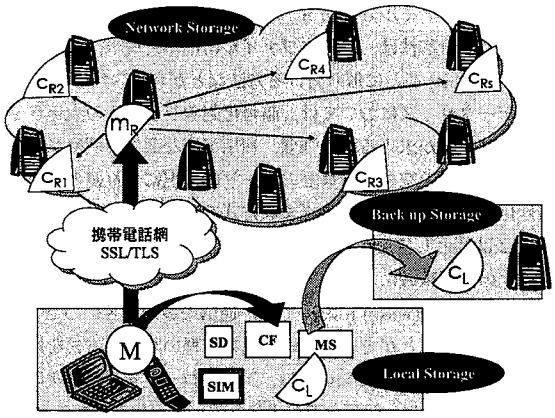

図 1　分散ストレージシステム構成モデル

図 1 に示す．このシステムにおいては，モバイル端末を紛失した場合でも，ネットワークストレージに対するアクセスを遮断することで元データの復元・不正利用を防ぎ，またユーザはネットワークストレージとバックアップストレージの情報を用いることでモバイル端末の元の所有者はデータを容易に復元をすることができる．

2.2　自己暗号化法

自己暗号化法による暗号化において，暗号化対象ファイル M は s 個の分散データに分割される．ここで分割データ m_1 から鍵を生成し，その鍵を用いて次の分割データに対して暗号化を施す．次に生成された暗号データから暗号鍵を生成し，以降これを繰り返し，同様に分割データに対して暗号化を施す．そして最後に生成された k_s を用いて，はじめの分割データに暗号化を施す．この手続きにおいて暗号鍵はコンピュータが内部処理のみに使用し，暗号鍵は使用されしだい自動的に消去する．これによりセキュアな鍵管理ができるのに加え，ユーザが暗号鍵を把握する必要がない．本方式においては暗号化対象ファイルの大きさや性質，ユーザが求める暗号強度，利用可能な通信帯域やローカルストレージの空き容量などのユーザの状況によりネットワークストレージにアップロードする分割ファイルのサイズ，鍵の生成や暗号化アルゴリズムをさまざまな方式から選択し，限られたリソースを有効に活用する．

2.3　自己暗号化法の定式化

自己暗号化法による暗号化において，暗号化対象ファイル M は分割アルゴリズム S により s 個の $m_1, m_2, \cdots, m_s$ に分割され，m_1 からコンピュータが自動的に鍵生成関数 F により暗号鍵 k_1 を生成する．その k_1 を用いて暗号化アルゴリズム E_{k_1} により m_2 を暗号化し c_2 を生成する．次に c_2 から F により暗号鍵 k_2 を生成する．以降同様に分割データに対する暗号化を繰り返す．最後に c_s から生成した k_s を用いて，m_1 に E_{k_1} により暗号化を施し c_1 とする．

復号においては，$c_2, c_3, \cdots, c_s$ から F により暗号鍵 $k_2, \cdots, k_s$ を生成し，復号アルゴリズム D_k を用いて $m_3, m_4, \cdots, m_s$ および m_1 を復号する．そしてに m_1 から k_1 を生成，m_2 を得る．最後に $m_1, m_2, \cdots, m_s$ から結合アルゴリズム S^{-1} により M を結合する．以上から自己暗号化法を表 1 のように定式化する．

表 1　自己暗号化法の定式化

暗号化手続き	式
M の分割	$\{m_1, m_2, \cdots, m_s\} = S(M)$
鍵の生成	$k_1 = F_1(m_1), k_p = F_p(c_p)(p = 2, 3, \cdots, s)$
m_p の暗号化	$c_1 = E_{k_s}(m_1), c_p = E_{k_{p-1}}(m_p)(p = 2, 3, \cdots, s)$
c_p の復号化	$m_1 = D_{k_s}(c_1), m_p = D_{k_{p-1}}(c_p)(p = 2, 3, \cdots, s)$
M の結合	$M = S^{-1}\{m_1, m_2, \cdots, m_s\}$

2.4　自己暗号化法システム構成例

モバイル端末においてリモートデータの暗号化を省略した自己暗号化法により，ネットワークストレージとローカルストレージにそれぞれ m_L と c_L を分散配置するとき，暗号化・復号それぞれについての手続きを示す．

2.4.1　モバイル端末における自己暗号化法 暗号化

図 2 に自己暗号化法による暗号化の手順を示す．

（1）　機密ファイル M を m_L，m_R に S により分割する
（2）　m_R から F_R を用いて暗号鍵 k_R を生成する
（3）　m_R をネットワークストレージにアップロードする
（4）　k_R を用いて E_{k_R} により m_L を暗号化し，c_L を生成
（5）　c_L をローカルストレージ内に保存する
（6）　c_L をバックアップストレージにコピーする

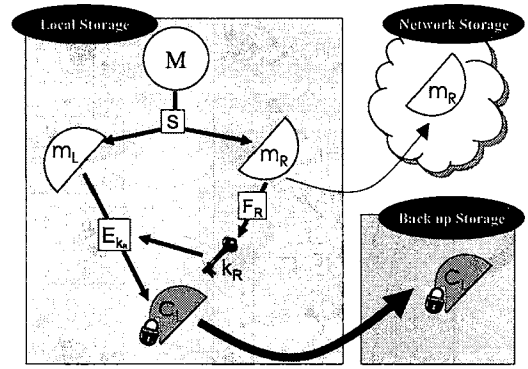

図 2　自己暗号化法 暗号化モデル

2.4.2　モバイル端末における自己暗号化法 復号

図 3 に自己暗号化法による復号の手順を示す．

（1）　ネットワークストレージ上の m_R をダウンロードする
（2）　m_R から F_R を用いて暗号鍵 k_R を生成する
（3）　k_R により c_L を D_{k_R} により復号し，m_L を生成する
（4）　m_L，m_R を S^{-1} により結合し，元ファイル M を得る

2.4.3　バックアップからの復号

端末紛失時には，ローカルデータ c_L もともに失われてしまう．この場合にはバックアップストレージの c_L を端末にダウンロードし，2.4.2 節と同様に復号する．ここでは c_L から復号する際に必要となるパートナである c_R を特定するような分散ファイル間における関連付けが必要であるが，この件については 4. 節で詳しく述べる

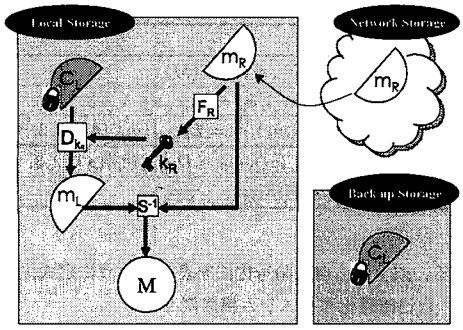

図 3　自己暗号化法 復号モデル

2.4.4　ネットワークサーバにおける自己暗号化法 暗号化

ネットワークサーバにおいて m_R を s 個の分散データ $c_2, c_3, \cdots, c_s$ に暗号化する手続きは図 4 のようになる．m_R は分割アルゴリズム S により s 個の $m_{R_1}, m_{R_2}, \cdots, m_{R_s}$ に分割され，m_{R_1} から F_{R_1} により暗号鍵 k_{R_1} を生成する．その k_{R_1} を用いて暗号化アルゴリズム $E_{k_{R1}}$ により m_{R_2} を暗号化し c_{R_2} を生成する．次に c_{R_2} から F_{R_2} により暗号鍵 k_{R_2} を生成する．以降同様に分割データに対する暗号化を繰り返す．最後に c_{R_s} から生成した k_{R_s} を用いて，m_{R_1} に $E_{k_{R_s}}$ により暗号化を施し c_{R_1} とする．

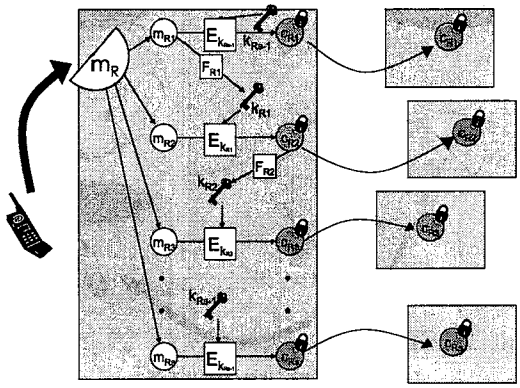

図 4　ネットワークサーバにおける自己暗号化法 暗号化

2.4.5　ネットワークサーバにおける自己暗号化法 暗号化

復号においては，$c_{R_2}, c_{R_3}, \cdots, c_{R_s}$ から暗号鍵 $k_{R_2}, \cdots, k_{R_s}$ を生成し，復号アルゴリズムを用いて $m_{R_3}, m_{R_4}, \cdots, m_{R_s}$ および m_{R_1} を復号する．そしてに m_{R_1} から k_{R_1} を生成，m_{R_2} を得る．最後に $m_{R_1}, m_{R_2}, \cdots, m_{R_s}$ から結合アルゴリズム S^{-1} により M を結合する．

2.5　自己暗号化法アルゴリズム選択

自己暗号化法においては，まずデータの利用方法や大きさ，通信速度などを総合的に考慮する．そしてその上で，状況に応じた様々なアルゴリズムを選択し，モバイル端末向けの負荷の軽い暗号化方式や通信帯域に見合った Upload ファイル容量の選択など，限られたリソースを活用することを狙う．

2.5.1　ファイル分割法 S

ファイル分割法は，単純にファイルの前半と後半に分割する方法や，n[bit] ごとに振り分ける方法などが考えられる．また，分割データサイズについては，暗号化対象ファイルの大きさや性質，ユーザが求める暗号強度，利用可能な通信帯域やローカルストレージの空き容量などのユーザの状況により鍵の生成や暗号化アルゴリズムをさまざまな方式を考えることができる．

表 2 は例として 2[MB] のファイルを暗号化する際，ネットワークの通信帯域による分割データサイズの決定を示したものである．CASE(wide) においては，通信帯域が大きいので m_R を大きくすることができ，処理の比較的軽い暗号化方式を用いる．CASE(narrow) においては，アップロードするデータの容量を減らすために，処理負荷の大きな強い暗号化を施すことで，m_L に対して m_R を小さくする．

表 2　ファイル分割例

	帯域	m_R	m_L
CASE(wide)	大	1,000[KB]	1,000[KB]
CASE(narrow)	小	10[KB]	1,990[KB]

2.5.2　暗号化アルゴリズム E，鍵生成法 F

CASE(narrow) においては，E_{k_R} は，長いデータ m_L を短いデータ m_R を用いて暗号化するため，CBC(Cipher Block Chaining) [4], [5] や CFB(Cipher Feedback) [4] などの利用モードを用いて，AES や Triple DES などの適当な暗号方式によりブロック暗号を施すことも考えられるが，ここでは，携帯網などで効率のよい通信，および処理の軽い暗号化処理の実現のために，暗号化アルゴリズム E は，排他論理和によるストリーム暗号を考える．

ここで，ストリーム暗号の安全性のために，鍵生成アルゴリズム F_R としては，短いデータから，長いランダム列を生成するために，SHA-1，SHA-256 などのセキュアな一方向ハッシュ関数を使用する鍵導出関数 KDF(Key Derivation Function) を使うことなどが考えられる [6]〜[8]．CASE(wide) において F_R は鍵 k_R と m_R が同じ大きさになるため，KDF による鍵生成の他に KDF よりも処理負荷の軽い，排他論理和による攪拌を鍵の生成法として用いる．

3.　DHT

DHT はデータを一意に特定する Key からそのデータの Location を解決するための探索技術である．DHT では P2P の各ノードが全体の知識を保持しなくとも，ノードの協調動作により探索が成立する．また多くの実装では参加ノードの入れ替わりをサポートしており，ノードのネットワークへの参加，離脱への耐性をもつことで必ずしも保障されないノードの集合においても信頼性のおけるネットワークを構成できる．DHT では探索対象となる Content とノードそれぞれに対して Hash 関数により Hash 値を割り当て，分散されたノードが協調して Content を管理し，各 Content は近傍のノードと結び付けられる．

3.1 Chord

DHT には Chord [9],CAN [10],Pastry [11],Kademlia [12] など様々な実装方法が存在するが，ここでは Chord を例とし DHT の動作について述べる．Chord では，$0 \sim 2^N - 1$ までの数値を円周上に配置した円状スキップリストに，各ノードを Hash 関数により並べた時，自身の次に小さい Hash 値を持つノード (predecessor) から自分の Hash 値までの範囲に入る Hash 値に対応する Content に対して責任を持つ．また各ノードは自身の次に大きい Hash 値を持つノード (successor) のアドレスを記憶しておく．ある Key に対する Location の探索・登録を行う場合は，順々に successor をたどり，その Key に対応するノードまで要求を転送する．ただしこのルーティングを効率化するため，successor 以外のいくつかのノードのアドレスも保持する．これを Finger Table といい，N 個のノードのアドレスを保持することで，最悪でも $logN$ 回の通信でルーティングが完了する．またノードの参加や離脱に備えて各ノードは複数の predecessor が保持する情報の複製を保持する．

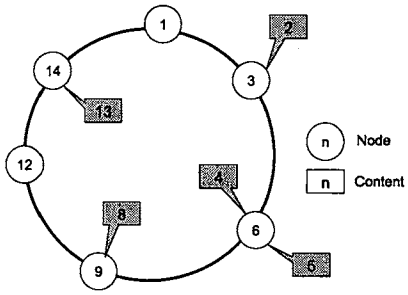

図 5 Chord

3.2 DHT によるネットワークストレージの構成

本稿では，ネットワークサーバが SSL/TLS などのプロトコルを用いて盗聴や改ざん，なりすましに対するセキュリティを施した通信路により，端末から平文のまま送られてきたリモートデータを s 個の分散データに暗号化する．そして分散データを置くネットワーク上のストレージを DHT による広域ストレージとし，ほかのユーザとあえてインフラを共用する．そして m ユーザで暗号文をシェアすることにより復号への試行が ${}_{(m*n)}C_2$ 通りになり，特定の分散データの組を見つけるのが困難になる．またこれに加えて DHT によりネットワークを構成することにより，P2P ノードをユーザのホームサーバ郡などが担当することで特定の企業や組織からは独立した管理になるほか，ファイルの負荷分散や耐障害性ももつことになる．ここでストレージはネットワークサーバ全体で取り扱うが，端末からは平文のリモートデータが送信されることから，リモートサーバにおける暗号化処理のみは，ユーザの所有するサーバなどの信頼の置けるノードを選択し利用する必要がある．ここでの通信の流れを図 6 に示す．

4. ローカルデータとリモートデータの関連付け

本システムの設計においては，情報の暗号化・復号はファイルシステムによる自動化により達成される．そのためには，c_L をローカルディスク上の任意のディレクトリへ移動する場合や，紛失や機種変更などにより端末自体を置換し，新しい端末にコピーした場合などにも c_L に対応する m_R をダウンロードできなければならない．そこで c_L の Hash 値およびファイルサイズを用いて DHT における m_R の分散データの Key を計算する．これによりディレクトリや端末間での移動・コピーを行っても復号が可能になる．また同じ内容のデータが複数存在する場合には *Linkcount* を用いて扱うことで，端末とネットワークストレージ間の不要な通信を削減する．

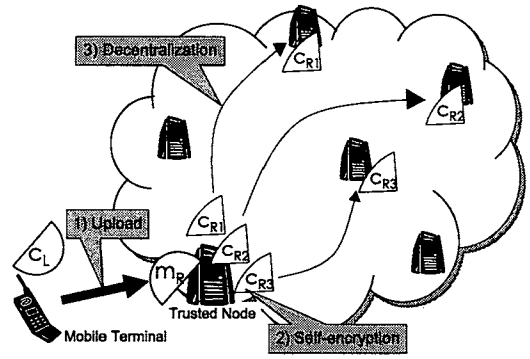

図 6 DHT によるネットワークストレージ

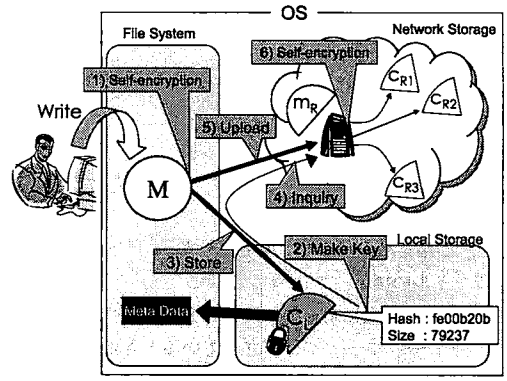

図 7 暗号化手続き

4.1 暗号化手続き

図 7 に暗号化する際の手続きを示す．

（1） M を自己暗号化法により c_L・m_R に分散

（2） c_L の Hash 値およびファイルサイズから Key を計算

（3） c_L をローカルストレージに保存

（4） m_R をすでに DHT が管理しているか問い合わせる

（5） 管理していない場合 m_R をネットワークサーバにアップロードし *LinkCount* を 1 に設定，すでにしていた場合は，*LinkCount* をインクリメントする

（6） サーバが自己暗号化法により m_R を分散

4.2 復号手続き

図 8 に復号化の手続きを示す．

（1） c_L をローカルストレージから読み込む

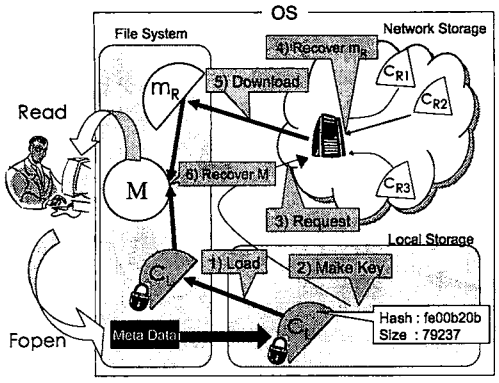

図 8 復号化手続き

(2) c_L から Key を計算

(3) 計算した Key から，該当する m_R をネットワークに対して要求する

(4) サーバが m_R を復号

(5) m_R を端末にダウンロードする．サーバでは *LinkCount* をデクリメントする

(6) c_L と m_R から元ファイルを復号する

4.3 ファイルの複製

機密ファイル M が既に暗号化されており，端末内に c_L，ネットワークストレージに m_R が保存されているときにコピー c_L' と c_R' を生成すること場合は，M を復号した後，複製ファイル M' を生成し，M と M' に対して暗号化を施すという手順を踏むと，大きなオーバーヘッドがかかってしまう．2. 節に示すように，c_L と m_R の関連付けについての処理は，クライアントであるモバイル端末では多くを行わず，ネットワークストレージに担当させる構成が望ましい．この場合には，端末において c_L と同一内容の c_L' を生成し，Key を計算し，ネットワークストレージに送信して，この Key に対応する m_R の *LinkCount* をインクリメントする．

5. P2P 分散ファイル共有基盤に基づくプライベートストレージ

本節では，モバイル端末内のローカルストレージとネットワークサーバにより構成するプライベートストレージをユーザの近くにある端末により用いる流れについて述べる．

5.1 モバイル端末とネットワークストレージ連携によるプライベートストレージ

モバイル端末内のローカルストレージとネットワークストレージを用いて構成したプライベートストレージを他のユーザも使う可能性がある公共端末などにおいてもセキュアに利用するモデルを図 9 に示す．ユーザはモバイル端末から携帯電話網や SSL/TSL によりセキュリティを保証した通信路を通じてサーバに対して m_R の Key を送信する．サーバは復号した m_R に一時的な URI を付け，サーバ上で公開し，モバイル端末に対して連絡する．次にモバイル端末と固定端末においてセキュアな PAN を構築し，WIFI や WUSB などの広帯域無線ネットワークを通じて，端末から m_R の URI と c_L を固定端末と共有する．固定端末はこの URI を用いて m_R を受け取り，c_L とともに M を復号する．一時的な URI を用いて固定端末の m_R へのアクセスを管理し，ダウンロードがあったあとは URI を無効化することで，不特定多数のユーザに使われる公共端末からでもセキュリティを保ったまま情報を利用できる．

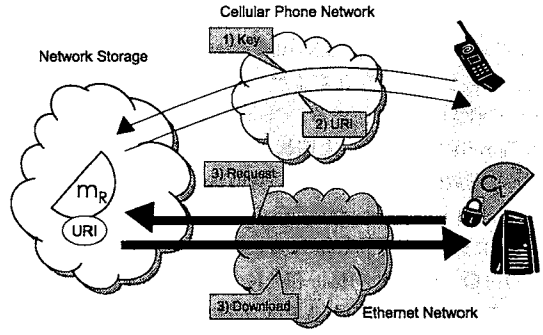

図 9 固定端末との連携

文 献

[1] A. Spruill and C. Pavan, "Tackling the U3 trend with computer forensics," digitalinvestigation 4, pp.7-12, 2007.

[2] R. Want, T. Pering, Gunner Danneels, Muthu Kumar, M.Sundar, and J. Light, "The Personal Server: Changing the Way We Think about Ubiquitous Computing," UbiComp 2002: Ubiquitous Computing : 4th International Conference, Goteborg, Sweden, September 29 - October 1, 2002. Proceedings

[3] 遠藤大礎，川原圭博，浅見徹，"自己暗号化法によるモバイル端末分散ストレージの実装と評価，" 2007 信学会総大，Mar.2007.

[4] "Federal Information Processing Standards Publication 81, Data Encryption Standard (DES)," NIST, 1980.

[5] M. Ballare, A. Desai, E. Jokipii and P. Rogaway, "A Cocrete Security Treatment of Symmetric Encryption: Analysis of the DES Modes of Operation," Proceeding of the 38th Symposium on Foundations of Computer Science, IEEE, 1997.

[6] "FIPS PUB 180-2," http://csrc.nist.gov/publications/fips/fips180-2/fips180-2withchangenotice.pdf, NIST, 2002.

[7] 総務省，経済産業省，"電子政府推奨暗号リスト，" http://www.soumu.go.jp/joho_tsusin/security/img/cryptrec01.pdf, 2003.

[8] "SUMMARY OF ANSI X9.63," NIST, 1999.

[9] I. Stoica, R. Morris, D. Liben-Nowell, David R. Karger, M. Frans Kaashoek, F. Dabek, and H. Balakrishnan, "Chord: A scalable peer-to-peer lookup protocol for internet applications," In IEEE/ACMTrans. on Networking, 2002.

[10] S. Ratnasamy, P. Francis, M. Handley, R. Karp and S. Schenker, "A scalable content-addressable network," Proceedings of the 2001 conference on Applications, technologies, architectures, and protocols for computer communications, p.161-172, Aug. 2001.

[11] A. Rowstron and P. Druschel, "Pastry: Scalable, Decentralized Object Location, and Routing for Large-Scale Peer-to-Peer Systems," Middleware 2001 : IFIP/ACM International Conference on Distributed Systems Platforms Heidelberg, Germany, November 12-16, 2001. Proceedings

[12] P. Maymounkov and D. Mazieres, "Kademlia: A Peer-to-Peer Information System Based on the XOR Metric," Peer-to-Peer Systems: First InternationalWorkshop, IPTPS 2002 Cambridge, MA, USA, March 7-8, 2002. Revised Papers