Diss. ETH No. 22475

# Secure Data Deletion

A thesis submitted to attain the degree of

DOCTOR OF SCIENCES of ETH ZURICH

(Dr. sc. ETH Zurich)

presented by

Joel Reardon

Master of Mathematics, University of Waterloo
born 31.07.1983
citizen of Canada

accepted on the recommendation of

Prof. Dr. Srdjan Čapkun, examiner
Prof. Dr. David Basin, coexaminer
Prof. Dr. Ari Juels, coexaminer
Prof. Dr. Paul Van Oorschot, coexaminer
Dr. Alessandro Sorniotti, coexaminer

2014

# Abstract

Secure data deletion is the task of deleting data from a physical medium so that the data is irrecoverable. This irrecoverability is what distinguishes *secure* deletion from *regular* deletion, which ostensibly deletes unneeded data only to reclaim resources. We *securely* delete data to prevent an adversary from gaining access to it, and so secure deletion is a natural part of the confidentiality of data.

In this thesis, we examine secure deletion in a variety of different systems and different layers: from the hardware level of ensuring a storage medium can efficiently delete data to the system level of deleting data from unreliable and untrusted servers. We examine related work in detail, identify the deficiencies and unsolved problems, and build our own solutions to advance the state of the art. Our analysis provides a framework to reason about secure deletion in general. We organize existing solutions in terms of their interfaces to physical media and further present a taxonomy of adversaries differing in their capabilities as well as a systematization for the characteristics of secure deletion solutions. We then design a system and adversarial model for secure deletion that encompasses the most challenging aspects that we distill from our survey. Our research contributions are then provided within this model.

We consider secure deletion in the context of two main types of storage media: mobile storage and remote (e.g., cloud) storage. At the time that the research was undertaken, both these computational environments represented a significant shift in how most people accessed their data. The lack of secure deletion is a security concern in both settings. Both store sensitive user data and both are vulnerable to adversarial compromise. Despite the massive difference in the scale of these devices, the challenges of secure deletion shares surprising similarities.

Secure deletion for mobile devices means secure deletion for flash memory, as it is currently ubiquitously used in portable storage devices. Flash memory has the problem where the unit of erasure is much larger than the unit of read and write, and worse, erasure incurs a greater cost that is manifested in power consumption and physical wear.

Our first contribution for flash memory is research into user-level secure deletion for flash memory, that is, what can be done by a user on their portable device by simply adding an application. We use a concrete example of an Android-based mobile phone. We show that it provides no guarantees on data deletion and the time data remains increases with the storage medium's size. We propose two user-level solutions that achieve secure deletion as well as a hybrid of both solutions, which guarantees the periodic, prompt secure deletion of data regardless of the storage medium's size.

Our second contribution for flash memory is the Data Node Encrypted File System (DNEFS), a file system extension that provides fine-grained efficient secure data deletion. We implement DNEFS in the flash file system Unordered Block Images File System (UBIFS) and call our implementation UBIFSec. We further integrate UBIFSec in the Android operating system running on a Google Nexus One smartphone. We show that it is efficient; Android OS and applications (including video and audio playback) run normally on top of UBIFSec.

Secure deletion for remote storage means secure deletion for persistent storage, that is, a storage medium that is unable to delete *any* data. The reason to thus model cloud storage is that once the data has left the users' control, users are unable to themselves ensure that access control and secure deletion are performed correctly. To compensate for the persistent storage medium's inability to perform deletion, we assume that the user has access to a small securely-deleting storage medium to manage the encryption keys that permit the secure deletion of data.

Our first contribution for persistent storage is to present a general solution to the design and analysis of secure deletion for persistent storage that relies on encryption and key wrapping. We define a key disclosure graph that models the adversarial knowledge of the history of key generation and wrapping. We introduce a generic update function as a key disclosure graph mutation and prove that the update function achieves secure data deletion; instances of the update function implement the update behaviour of all arborescent data structures. We find that related work fits well in our model and characterize their key dis-

closure graphs. We then design and implement a B-Tree instance within the space of possible securely-deleting data structures and analyse its performance, finding that its overheads are small.

Our second contribution for remote storage considers the problem of an unreliable securely-deleting storage medium, that is, one that may lose data, expose data, fail to delete data, and fail to be available. We build a robust fault-tolerant system that uses multiple unreliable storage media. The system permits multiple clients to store securely-deletable data and provides a means to control policy aspects of its storage and deletion. We present details on the implementation both of the distributed securely-deleting medium as well as a file system extension that uses it. The solution has low latency at high loads and requires only a small amount of communication among nodes.

# Zusammenfassung

*Sicheres Löschen von Daten* ist der Vorgang des Entfernens von Daten von einem physischen Medium, so dass diese Daten unwiederbringlich zerstört sind. Die Unwiederbringlichkeit unterscheidet *sicheres* von *regulärem* Löschen, welches vorgeblich unnötige Daten löscht, um Ressourcen zurückzugewinnen. Wir löschen Daten *sicher*, um einen Widersacher daran zu hindern Zugang zu ihnen zu bekommen. Und damit stellt sicheres Löschen einen natürlichen Bestandteil der Datenvertraulichkeit dar.

In dieser Arbeit untersuchen wir sicheres Löschen anhand einer Auswahl verschiedener Systeme und Schichten: von der Hardwareschicht, welche sicher stellt, dass Daten effizient gelöscht werden, bis zu der Systemschicht, die Daten von unzuverlässigen und misstrauten Servern löscht. Eingehend untersuchen wir andere themenbezogene Arbeiten, identifizieren deren Mängel und ungelöste Probleme und erstellen unser eigenes Lösungskonzept, um den derzeitigen Stand der Technik anzuheben. Unsere Darlegung liefert einen allgemeinen Rahmen, um vernünftig über sicheres Löschen reden zu können. Wir strukturieren existierende Lösungen bezüglich ihrer Schnittstellen zu physischen Medien. Weiterhin präsentieren wir eine Widersachertaxonomie, aufgeschlüsselt nach deren Fähigkeiten, sowie eine Systematisierung der Charakteristika die sicheres Löschen auszeichnen. Danach entwerfen wir ein System- und Feindmodell für sicheres Löschen, welches die anspruchsvollsten Aspekte einschliesst, die wir in unserer Studie extrahieren konnten. Unser wissenschaftlicher Beitrag wird schliesslich in diesem Modell dargestellt.

Wir betrachten sicheres Löschen hinsichtlich von zwei Hauptarten von Speichermedien: mobile Speicher und verteilte (z.B. Cloud-) Speicher. Zum Zeitpunkt dieser Arbeit führten beide Rechenumgebungen zu einer gravierenden Veränderung in der Art und Weise, mit der die

meisten Menschen auf ihre Daten zugreifen. Das Fehlen von sicherem Löschen ist ein Sicherheitsbelang in beiden Umgebungen. Beide verwahren sensitive Nutzerdaten und beide sind verwundbar hinsichtlich feindlicher Übernahme. Trotz der massiven Unterschiede in der Grössenordnung dieser Speichermedien teilen sie sich eine überraschend grosse Anzahl an Gemeinsamkeiten.

Sicheres Löschen auf mobilen Geräten bedeutet sicheres Löschen auf Flash-Speichern, welche derzeit überall auf tragbaren Speichergeräten eingesetzt werden. Flash-Speicher haben das Problem, dass die Einheiten, die gelöscht werden, viel grösser sind als die Einheiten, die gelesen oder geschrieben werden. Dadurch fallen grössere Kosten an, die sich in erhöhtem Energieverbrauch und physischer Abnutzung äussern.

Unser erster Beitrag in Bezug auf Flash-Speicher ist die Erforschung des sicheren Löschen von Flash-Speichern auf Benutzerebene, d.h. was alleiniges Hinzufügen einer Anwendung einem Benutzern auf seinem tragbaren Gerät diesbezüglich ermöglichen kann. Wir benutzen hierfür ein konkretes Beispiel eines auf Android basierenden Mobiltelefons. Wir zeigen, dass keine Garantien für das Löschen von Daten gewährleistet werden können und dass die Dauer, wie lange Daten auf dem Speichermedium bestehen bleiben, mit dessen Grösse des Mediums ansteigt. Wir schlagen zwei Lösungskonzepte auf Benutzerebene vor, welche sicheres Löschen ermöglichen, und weiterhin eine Hybridlösung von beiden, die periodisches und sofortiges, sicheres Löschen unabhängig von der Grösse des Speichermediums garantiert.

Unser zweiter Beitrag hinsichtlich Flash-Speicher ist DNEFS, eine Anpassung am Dateisystem, die ein präzises und effizientes, sicheres Löschen von Daten ermöglicht. Wir implementieren DNEFS in dem Flash-Dateisystem UBIFS und nennen es UBIFSec. Ausserdem integrieren wir UBIFSec im Android Betriebssystem auf einem Google Nexus One Mobiltelefon. Wir zeigen, dass es effizient ist und dass das Android Betriebssystem zusammen mit Anwendungen (einschliesslich Video- und Audio-Wiedergabe) normal unter UBIFSec laufen.

Sicheres Löschen für verteilte Speicher bedeutet sicheres Löschen für dauerhafte Speicher, d.h. einem Speichermedium, auf dem es unmöglich ist *jegliche* Daten zu löschen. Der Grund verteilte Speicher so zu modellieren besteht darin, dass, wenn sich die Daten einmal der Kontrolle des Benutzers entzogen haben, dieser unfähig ist sicher zu stellen, dass Zugangsberechtigungen und sicheres Löschen weiterhin korrekt ausgeführt werden. Um die Unfähigkeit des Löschens auf dem dauerhaften Speichermediums zu kompensieren, nehmen wir an, dass der Benut-

zer Zugang zu einem kleinen, sicher löschenden Speichermedium hat, auf dem die kryptographischen Schluüssel, welche das sichere Löschen erlauben, verwaltet werden.

Unser erster Beitrag im Bezug auf dauerhafte Speicher besteht darin, dass wir ein allgemeines Lösungskonzept in Form eines Entwurfs und einer Analyse von sicherem Löschen präsentieren, welche auf Verschlüsselung und Schlüsselverpackung basiert. Wir definieren einen Schlüsselenthüllungsgraphen, der das feindliche Wissen über die Vergangenheit der Schlüsselerzeugung und Schlüsselverpackung modelliert. Wir führen eine generische Aktualisierungsfunktion als eine Schlüsselverpackungsmutation ein und beweisen, dass die Aktualisierungsfunktion sicheres Löschen von Daten gewährleistet. Instanzen der Aktualisierungsfunktion implementieren das Aktualisierungsverhalten von allen baumartigen Datenstrukturen. Wir stellen fest, dass andere themenbezogene Arbeiten gut in unser Modell passen und wir charakterisieren deren Schlüsselenthüllungsgraphen. Weiter entwerfen und implementieren wir eine B-Baum Instanz als eine mögliche, sicher löschende Datenstruktur und analysieren ihre Leistung, wobei sich herausstellt, dass der zusätzliche Rechenaufwand gering ist.

Unser zweiter Beitrag hinsichtlich dauerhafter Speicher beleuchtet das Problem eines unzuverlässigen, sicher löschenden Speichermediums, d.h. eines, das Daten verlieren kann, Daten preisgibt, scheitert Daten zu löschen oder zeitweise nicht verfügbar ist. Wir erstellen ein robustes und fehlertolerantes System, das viele unzuverlässige Speichermedien benutzt. Das System erlaubt mehreren Klienten sicher löschbare Daten zu speichern und bietet die Möglichkeit die Art und Weise des Speicherns und Löschens genauer zu kontrollieren. Wir präsentieren Details zu der Implementierung sowohl für das verteilte, sicher löschende Medium als auch für eine Dateisystemerweiterung, die es benutzt. Das Lösungskonzept weist eine geringe Verzögerung bei grossen Datenmengen auf und benötigt nur einen geringen Kommunikationsaufwand zwischen den Knoten.

# Resumé

La suppression de données sécurisées est la tâche de supprimer des données d'un support physique tel que les données sont irrécupérables. L'irrécupérabilité est la différence entre la suppression sécurisées et la suppression ordinaire, qui prétend d'effacer des données non nécessaires afin de libérer de l'espace de stockage. Nous supprimons des données d'une manière *sécurisées* afin d'empêcher un adversaire d'y accéder, et donc la suppression sécurisé fait partie naturelle de la confidentialité des données.

Dans cette thèse, nous examinons la suppression sécurisée dans une variété de différents systèmes et différentes couches : à partir du niveau materiel partant d'un support de stockage qui peut supprimer des données de manière efficace jusqu'au niveau système avec la suppression de données sur des serveurs non fiables. Nous examinons en détail les travaux connexes, identifions les lacunes et les problèmes non résolus, et nous construisons nos propres solutions pour faire progresser l'état de l'art. Notre analyse fournit un cadre pour raisonner sur l'effacement sécurisé en général. Nous organisons les solutions existantes en termes de leurs interfaces à support physique et en outre présentons une taxonomie d'adversaires différents dans leur capacités ainsi qu'un systématisation des caractéristiques de sécurité des solutions de suppression. Nous concevons ensuite un système et un modèle d'adversaire pour la suppression sécurisé qui englobe les aspects les plus difficiles que nous distillons de notre enquête. Nos contributions de recherche sont ensuite fournis dans ce modèle.

Nous considérons la suppression sécurisée dans le cadre de deux principaux types de supports de stockage : le stockage mobile et à distance (par exemple, cloud) de stockage. Au moment où la recherche a été entrepris, ces deux environnements informatiques représentaient un important changement dans la faÃğon la plupart des gens accèdent

à leurs données. Le manque de suppression sécurisée est un problème de sécurité dans les deux cas. Les deux stockent des données sensibles de l'utilisateur et les deux sont vulnérables aux compromis d'un adversaire. Malgré l'énorme différence d'échelle de ces dispositifs, les défis de la part de la suppression sécurisée partagent des similitudes surprenantes.

La suppression sécurisée pour appareils mobiles signifie la suppression sécurisée de la mémoire flash, car actuellement ubiquitairement utilisé dans les dispositifs de stockage portables. La mémoire flash a le problème que l'unité d'effacement est beaucoup plus grande que l'unité de lecture et d'écriture, et pire encore, nécessite un coût plus élevé se manifestant dans la consommation d'énergie et de l'usure physique.

Notre première contribution pour la mémoire flash est la recherche au niveau utilisateur pour la suppression sécurisé à base de mémoire flash, ceux qui peut être fait par un utilisateur sur leur dispositif portable en simplement ajoutant une application.

Nous utilisons un exemple concret d'un téléphone mobile basé sur Android. Nous montrons qu'il ne donne aucune garantie sur la suppression de données et que le temps que les données persistent, augmente avec la taille du support de stockage. Nous proposons deux solutions au niveau de l'utilisateur qui permettent d'atteindre l'effacement sécurisé ainsi qu'un solution hybride, qui garantit la suppression périodique, rapide des données, indépendamment de la taille de support de stockage.

Notre deuxième contribution pour la mémoire flash est DNEFS, un changement de système de fichiers qui fournit une suppression de données sécurisé fine et efficace. Nous mettons en œuvre DNEFS dans le système de fichier flash UBIFS et appelons notre mise en œuvre UBIFSec. Nous intégrons UBIFSec dans le système d'exploitation Android fonctionnant sur un Google Nexus One smartphone. Nous montrons qu'il est efficace ; le système d'exploitation et les applications Android (y compris la vidéo et la lecture audio) fonctionnent normalement audessus de UBIFSec.

L'effacement sécurisé pour le stockage à distance est équivalent à la suppression sécurisée pour le stockage permanent, c'est à dire un support de stockage qui n'est pas en mesure de supprimer *n'importe quel* données. Ce modèle doit être choisit car le stockage cloud, une fois que l'utilisateur a perdu le contrôle sur ces données, est incapable de s'assurer que le contrôle d'accès et l'effacement sécurisé sont effectuée correctement. Pour compenser l'incapacité du support de stockage persistant à effectuer la suppression, nous supposons que l'utilisateur a

accès à un petit support de stockage permettant la suppression en toute sécurité afin de gérer les clés de chiffrement permettant la suppression sécurisée de données.

Notre première contribution pour le stockage permanent est de présenter une solution générale pour concevoir et analyser la suppression sécurisée pour le stockage permanent qui repose sur le chiffrement et l'emballage clé. Nous définissons un mécanisme de divulgation de clé à base d'un graphe qui modèle les connaissances de l'adversaire de l'histoire de la génération de clés et de leur emballage. Nous introduisons une fonction de mise à jour générique en forme de divulgation de clé à base de graphe avec mutation et nous prouvons que le mécanisme de mise à jour permet l'effacement sécurisé des données. Nous nous apercevons que les travaux connexes s'intègrent bien dans notre modèle et caractérisent leur graphe de divulgation de clé. Ensuite nous concevons et mettons en œuvre une instance B-Tree dans l'espace des possibles structures de données pour la suppression sécurisé et analysons sa performance, constatant que ces surplus généraux sont faibles.

Notre deuxième contribution pour le stockage à distance considère le problème d'un moyen de stockage à suppression sécurisée non-fiable, qui est celui qui peut perdre des données, exposer des données, ne pas supprimer des données, et ne pas être disponible. Nous construisons un système robuste, tolérant des pannes et qui utilise plusieurs supports de stockage non-fiables. Le système permet à plusieurs clients de stocker des données avec la possibilité de l'effacement sécurisé et fournit un moyen pour contrôler les paramètres de stockage et de suppression. Nous présentons les détails sur la mise en œuvre pour à la fois le moyen de suppression sécurisé distribué ainsi que pour une extension du système de fichiers qu'il utilise. La solution a une faible latence aussi pendant des charges élevées et nécessite peu de la communication entre les nœuds.

# Riassunto

L'attività di secure deletion consiste nel cancellare dati da un dispositivo fisico in modo da renderne impossibile il recupero. L'irrecuperabilità dei dati è ciò che distingue la secure deletion dal normale processo di cancellazione dati, che solitamente cancella i dati solo nel caso in cui le risorse da essi occupate diventino necessarie per altri scopi. L'obiettivo della secure deletion è prevenire l'accesso ai dati cancellati da parte di un avversario. Per questo motivo, l'attività di secure deletion è parte integrante parte della gestione della protezione e riservatezza dei dati.

Questa tesi esamina il problema della secure deletion in diversi sistemi e a diversi livelli: dal livello hardware, dove è necessario avere dispositivi di memorizzazione che possano effettuare la secure deletion in modo efficiente, al livello di sistema, in cui è necessario avere soluzioni che offrono capacità di secure deletion utilizzando server non affidabili. Questo studio presenta una dettagliata analisi delle soluzioni di secure deletion esistenti, ne identifica le mancanze ed i problemi non risolti, e presenta nuove tecniche di secure deletion che migliorano lo stato dell'arte. La nostra analisi presenta un framework per comprendere ed analizzare il problema della secure deletion. Le soluzioni esistenti sono organizzate in base a come queste si interfacciano ai dispositivi di memorizzazione fisici ed in base alle loro caratteristiche. Lo studio presenta anche una tassionomia dei vari adversarial model considerati fino ad ora. Infine, proponiamo un nuovo system model ed un nuovo adversarial model per il problema della secure deletion. Questi modelli considerano gli aspetti più complessi del problema. Tutti i risultati e le soluzioni presentate nella tesi suno esaminati nel contesto di questi modelli.

Questo studio considera il problema della secure deletion in due tipi di dispositivi di memoria: i dispositivi mobili ed i dispositivi di memorizzazione remota (per esempio, i servizi di cloud storage). La

mancanza di metodi di secure deletion è un problema a livello di sicurezza in entrambi gli scenari. Infatti, entrambi i dispositivi possono essere utilizzati per memorizzare dati riservati ed entrambi possono essere compromessi da un avversario. Nonostante le molteplici differenze tra questi dispositivi, le sfide da affrontare in termini di secure deletion sono simili.

In ambito mobile, è necessario effettuare la secure deletion sulle memorie flash, in quanto questo tipo di memorie è estremamente diffuso nei dispositivi mobile. Per le memorie flash, nel caso l'unità di cancellazione sia molto più grande dell'unità di lettura e scrittura si hanno maggiori costi in termini di consumo energetico e usura fisica dei dispositivi.

Il nostro primo contributo in questo ambito analizza la secure deletion a livello utente per le memorie flash, cioè, come un utente possa acquisire capacità di secure deletion semplicemente installando un'applicazione sul proprio dispositivo mobile. Come esempio concreto, abbiamo analizzato uno smartphone con sistema operativo Android. Esso non da nessuna garanzia sulla cancellazione dei dati, e il tempo di permanenza dei dati nella memoria del dispositivo dipende dalla capacità di memorizzazione dello stesso. Per risolvere questo problema, proponiamo due soluzioni che forniscono secure deletion, oltre ad una terza soluzione che combina le due precendenti e garantisce la periodica cancellazione dei dati indipendentemente dalla dimensione della memoria.

Il nostro secondo contributo nell'ambito delle memorie flash è DNEFS, un'estensione per file system che implementa la secure deletion in modo efficiente ed ad elevato livello di precisione. DNEFS è stato implementato nel file system per memorie flash UBIFS. UBIFSec è la versione di UBIFS che supporta DNEFS. Abbiamo integrato UBIFSec nel sistema operativo Android su di un dispositivo Google Nexus One. Attraverso alcuni esperimenti, mostriamo che UBIFSec è sia efficiente; Android OS e varie applicazioni (incluse applicazioni di riproduzione audio e video) vengono eseguite normalmente.

Per ottenere la secure deletion nell'ambito dei dispositivi di memorizzazione remota è necessario cancellare dati da memorie persistenti, che, per loro stessa natura, non sono in grado di cancellare dati. Il motivo che ci ha spinti a modellare in questo modo la memorizzazione di dati tramite servizi cloud è che gli utenti, una volta che i dati non sono più in loro posseso, non sono in grado di controllarne l'accesso da parte di terzi e la loro cancellazione. Per compensare l'incapacità di

cancellare dati delle memorie persistenti, assumiamo che l'utente abbia accesso a un dispositivo, di capacità limitata, che offre secure deletion, per gestire le chiavi di cifratura.

Il nostro primo contributo nell'ambito delle memorie persistenti è una soluzione generale per la progettazione e l'analisi di techniche di secure deletion basate sulla cifratura e sul key wrapping. Tramite un key disclosure graph modelliamo la conoscenza, da parte dell'avversario, della sequenza di generazione e wrapping delle chiavi di cifratura. Proponiamo quindi una funzione di update generica, e dimostriamo che essa garantisce la secure deletion. Tale funzione di update è stata poi specializzata ed implementata per tutte le strutture dati ad albero. Tramite questo modello, è possibile rappresentare le altre tecniche di secure deletion esistenti, ed è possibile caratterizare i loro key disclosure graph. Infine, abbiamo implementato una soluzione che offre secure deletion per B-alberi ed abbiamo analizzato le sue prestazioni, trovando che i costi aggiuntivi dovuti alla secure deletion sono minimi.

Il nostro secondo controbuto considera un dispositivo di memoria non affidabile che offre capacità di secure deletion, cioè, un dispositivo che può inavvertitamente cancellare i dati, esporli a terze parti, non cancellari correttamente, o risultare inattivo. Proponiamo un sistema affidabile realizzato utilizzando vari dispositivi non affidabili. Il sisteme permette a diversi utenti di memorizzare dati e da modo ad essi di controllare vari aspetti della memorizzazione e della cancellazione. Illustriamo vari dettagli sia sull'implementazione del sistema di memorizzazione distribuito che su di un'estensione per file system che utilizza tale sistema. La nostra soluzione ha bassa latenza ad alti carichi di lavoro e richiede solo una minima quantità di comunicazione tra i vari nodi.