Fraunhofer
SIT

CASED

F2S2

Schnelle Ähnlichkeitssuche von Fuzzy Hashes

Christian Winter

Fraunhofer-Institut für
Sichere Informationstechnologie
(SIT)

Center for Advanced
Security Research Darmstadt
(CASED)

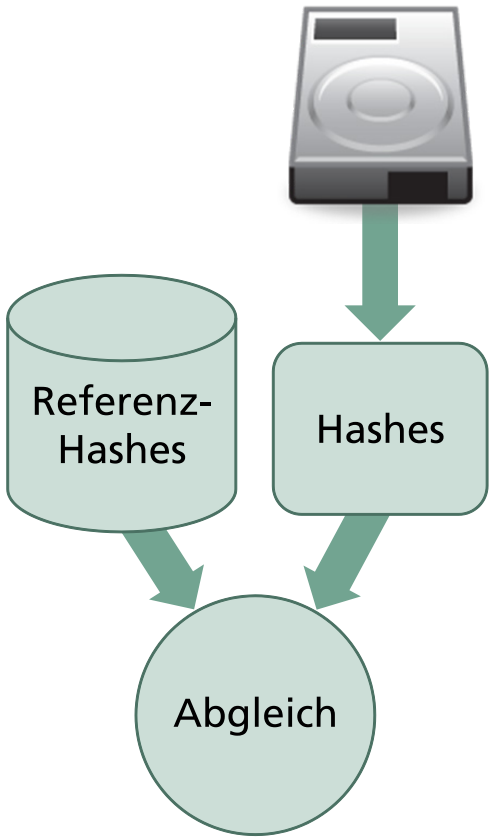
Anwendertag IT-Forensik 2012

Inhalt

1. Black- und Whitelisting
2. Piecewise Hashing
3. Ähnlichkeitssuche
4. F2S2 Software
5. Zusammenfassung und Ausblick

Automatisierung in der IT-Forensik

- Bekannte Inhalte in Ermittlungsfällen automatisiert wiedererkennen
- Blacklisting: Verdächtige/verbotene Inhalte finden
- Whitelisting: Irrelevante Inhalte ausblenden



Vorherrschende Praxis

- Liste von Hashwerten als Black-/Whitelist
- Identifizierung von Dateien durch Vergleich von Hashwerten
- Kryptographische Hashfunktionen (MD5, SHA-1, etc.)

Vorteile kryptographischer Hashfunktionen

- Einfache Umsetzung
- Schnelle Berechnung/Suche
- Klare Antwort (Treffer oder kein Treffer)
- Kollisionserzeugung schwierig
  $\Rightarrow$ Sprung auf die Whitelist schwierig

Nachteile kryptographischer Hashfunktionen

- Kleine Änderung einer Datei $\rightsquigarrow$ Große Änderung am Hash
  $\Rightarrow$ Sprung von der Blacklist sehr einfach
- Nicht erfasste Versionen bekannter Dateien werden nicht erkannt

Fuzzy Hashing

- Ziel: Ähnliche Dateien haben ähnliche Hashwerte.
- Frage: Was heißt ähnlich?

Inhaltsbasierte Hashfunktionen

- Auf einen Medientyp zugeschnitten
- Medienformate müssen interpretiert werden

Datenbasierte Hashfunktionen

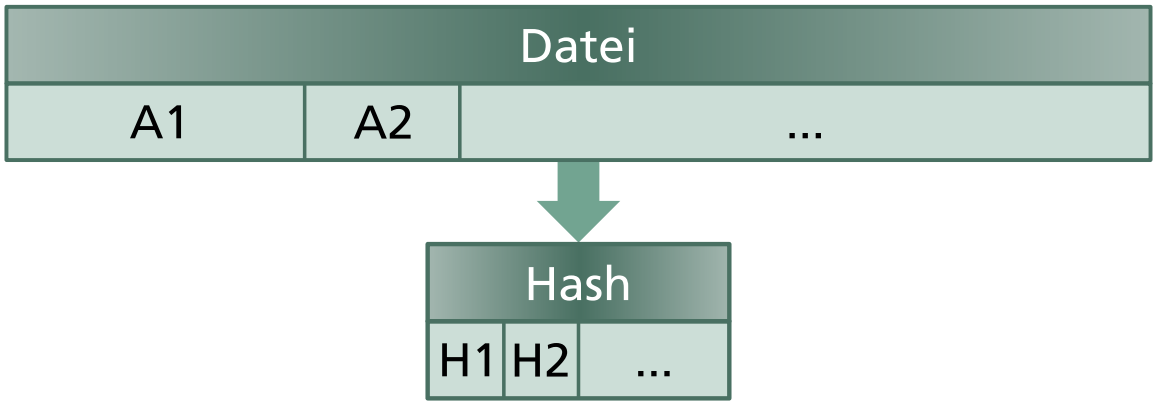
- Verfahren auf Roh-/Binärdaten $\rightsquigarrow$ Medien-/formatunabhängig
- Unterkategorien: Piecewise Hashing; Bloomfilter-basierte Verfahren

Verfahren

- Ein Teilhash pro Abschnitt der Datei
- Gesamthash ist Aneinanderreihung der Teilhashes
- Ähnlichkeitsmaß basiert auf Edit-Distanz

Context Triggered Piecewise Hashing

- Abschnittsgrenzen sind datenabhängig
- Beispiel: ssdeep von Jesse Kornblum



Geeignete Dateitypen

- Plaintext (`.txt`, `.eml`, etc.)
- Binärcode (`.exe`, `.dll`, etc.)
- Office-Dateien (`.doc`, `.xls`, etc.)

Anwendungsgebiete

- Software-Whitelisting
- Beziehungen zwischen Dokumenten (z. B. bei Wirtschaftskriminalität)

Ähnlichkeitssuche

Aufgabe

- Vergleiche Datei aus Ermittlungsfall mit Referenzliste (z. B. Black-/Whitelist)
- Finde ähnlichste Datei in Referenzliste oder
- Finde alle Dateien ähnlicher als vorgegebener Schwellwert etc.

Naiver Ansatz (Brute-Force)

- Vergleiche den Hash mit kompletter Referenzliste

Probleme

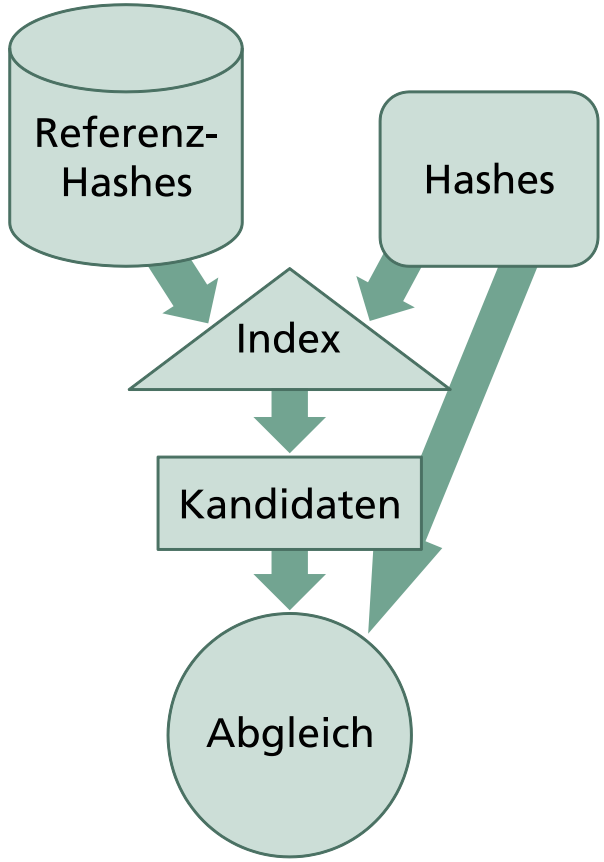
- Naiver Ansatz ist extrem langsam
- Klassische Indexstrukuren unterstützen keine Ähnlichkeitssuche

Ziel

- Indexstruktur dient zur Filterung der Referenzliste
- Ähnlichkeitsvergleich muss nur für kleine Kandidatenliste durchgeführt werden

Notwendige Bedingung

- Indexstruktur muss für Ähnlichkeitsmaß geeignet sein



Kernidee

- Ähnliche Hashes haben gemeinsame *n*-Gramme
- *n*-Gramme bilden die Grundelemente (Schlüssel) des Index
- Beispiel: Hash: `ABCDE`; 3-Gramme: `ABC`, `BCD`, `CDE`

Struktur des Index

- *n*-Gramme werden mit Verweis auf zugehörige Hashes gespeichert
- *n*-Gramme müssen effizient im Index lokalisiert werden können
- Index organisiert *n*-Gramme mittels Hash-Tabelle und sortierten Buckets

Funktionalität

- Erfassung der Referenzliste in Datenbank mit geeigneter Indexstruktur
- Erzeugung, Modifikation und Speicherung von Datenbanken
- Verschiedene Modi zur Ähnlichkeitssuche:
  z. B. alle „Nachbarn“, nächster „Nachbar“, erster „Nachbar“

Technische Eigenschaften

- Implementiert in C++
- Generische Bibliothek für beliebige Piecewise Hashes
- Spezialisierte Bibliothek für ssdeep-Hashes
- Kommandozeilen-Programm für ssdeep-Hashes
- F2S2 GUI für ssdeep-Hashes

Testfall

- Referenzliste vom NIST mit 8.334.077 Hashes
- „Ermittlungsfall“: 195.186 Hashes der Dateien eines Festplattenimages
- Analysecomputer mit Intel i7-930 Prozessor und 6 GiB RAM

ssdeep

- 1477 MiB RAM
- **364 Stunden** Laufzeit

F2S2

- 6811 bzw. 4747 MiB RAM
- 7 min zum Datenbank Erstellen (1 min zum Laden)
- **13 Minuten** für Suche aller „Nachbarn“

Fazit

- Ähnlichkeitssuche erweitert Möglichkeiten der IT-Forensik
- Geeignete Indexstruktur $\rightsquigarrow$ Beschleunigung um Faktor 1700
  $\Rightarrow$ Praktikable Geschwindigkeit

Ausblick

- Evaluierung verschiedener Piecewise-Hashing-Verfahren und verschiedener Ähnlichkeitsmaße
- Client-Server-Modus
- Low-RAM-Modus

Kontakt

Fraunhofer-Institut für
Sichere Informationstechnologie SIT
Rheinstraße 75
64295 Darmstadt
www.sit.fraunhofer.de

Christian Winter
06151 869-259
christian.winter@sit.fraunhofer.de