Secure Audit Logging Systems

with Privacy Preserving

Richard Kramer, Member IEEE – Oregon State University
How does someone know they have been HACKED!… and WHO did it!?
“An audit trail” that was maintained by the database company NGP VAN appears to show that four Sanders staffers conducted 25 specialized searches of the Clinton campaign's data, including queries for "turnout" and "primary priority" in a 40-minute window.”
“The incident was discovered after the hospital conducted an EHR [Electronic Health Record] audit back in October 2014. When it was first discovered only 14 individuals had had their PHI compromised.”
Contributions / Agenda:

- Provide a survey of Secure Audit Logging and review some important foundational work:
  - Schneier [3], Crosby [4], Goyal [5],

- Provide a detailed review of recent key publications:
  - **Multi-level user security with privacy preserving** - Se Eun Oh, et al., “**Privacy-preserving audit for broker-based health exchange**”, ACM, Proceedings of the 4th ACM conference on data and application security and privacy, 2014 [8,9]

- Identify potential **Future Work and applications for the benefit of Audit Logging for EHR (Electronic Health Records) related events**

- Provide an up-to-date list of Audit Logging tools and systems... some of them are **FREE!** [10]
What is an Audit Log?

Secure Audit Logs

... are logs that **securely store security related information and events**.

Examples include [1]:
- Reading critical files
- Account changes
- OS changes
- Major application changes
- Remote access
- Application transactions such as recording the sender / recipients of emails

Audit Logs are required by the government:
- Healthcare (HIPAA)
- Financial
- Legal
- Privacy Regulations
What Generates an Audit Log?

- Audit Logs are generated from a wide variety of aggregated sources including antivirus software, firewalls, intrusion detection systems, policy making systems [8], and the like.

Example [2]:

### Intrusion Detection System

```
[**] [1:1407:9] SNMP trap udp [**]
[Classification: Attempted Information Leak] [Priority: 2]
03/06-8:14:09.0882119 192.168.1.167:1052 -> 172.30.128.27:162
UDP TTL:118 TOS:0x0 IP:29101 IpLen:20 DgmLen:87
```

### Personal Firewall

```
3/6/2006 8:14:07 AM,"Rule "Block Windows File Sharing" blocked (192.168.1.54, netbios-ssn(139))." Rule "Block Windows File Sharing" blocked (192.168.1.54, netbios-ssn(139)). Inbound TCP connection. Local address,service is (KENT(172.30.128.27),netbios-ssn(139)). Remote address,service is (192.168.1.54,39922). Process name is ""System"."
```

### Antivirus Software, Log 1

```
```

### Antivirus Software, Log 2

```
24020301234,16,3,7,KENT,userk,.........,16777216,"Virus definitions are current.".0,0,0,0,0,0,0,0,0,0,0
```

### AntiSpyware Software

```
DGO Exploit: Data source object exploit (Registry change, nothing done) HKEY USERS\S-1-5-19\Software\Microsoft\Windows\CurrentVersion\Internet Settings\Zones\0\1004\W-3
```
It’s not enough to simply have an Audit Log

The Audit Log needs to be secure.

Securing Audit Logs is of the utmost importance because “Bad guys” seek to cover up their malicious activity.

Ideally -

1) We can **prevent alteration** of the logs
2) We can **verify, via analysis** that the logs have not been changed
3) We **only decrypt portions of the log to preserve privacy**

**The objective of Secure Audit Logging Systems is to protect Audit Logs from being compromised.**
Overview of the Art

Historically, a number of foundational papers have considered various systems to ensure the privacy and security of Audit Logs:

- **Schneier (1999), “Secure Audit Logs to Support Computer Forensics”** – Provides methods and systems for protecting an Audit Log such that the Audit Log is secure, even if the server that the Audit Log resides on, is compromised [3].

- **Crosby et al (2009)** – “Efficient Data Structures for Tamper-Evident Logging”. In short, Crosby introduced efficient data structures for **tamper-evident logging** [5] - only parts of the data is exposed [4], thus protecting private information.

- **Goyal et al (2006)**, “Attribute-based Encryption for fine-grained access control of encrypted data”. Protects privacy of the information in the Audit Log **based on attributes and user access levels** [5].
Overview of the Art - *Securing Audit Logs*

**Schneier** uses a “Hash Chain”, where new entries added to the log are hashed on top of previously hashed log entries [3].

Thus if a “bad guy” that took over a log server at some time, \( Y_j \), he could not go back and alter the log at time \( Y_{j-1} \) and before

\[ \text{Time } j-1 \]

\[ \text{Time } j \]
Overview of the Art - Securing Audit Logs

Schneier “Hash Chain”:

\[ Y_j = H(Y_{j-1}, E_{K_j}(D_j), W_j), \]

where \( Y_{j-1} \) is based on

\[ Y_{j-1} = H(Y_{j-2}, E_{K_{j-1}}(D_{j-1}), W_{j-1}) \]

and so on.

Where:

- \( W \) = log entry type (e.g., File Accessed, Permissions changed, etc.)
- \( D \) = log entry data
- \( Y \) = hash chain entry
- \( Z = MAC \) (Message Authentication Code)

\[ Y_{j-1} = H(Y_{j-2}, E_{K_{j-1}}(D_{j-1}), W_{j-1}) \]
Overview of the Art - **Detecting Tampering of an Audit Log**


In short, Crosby introduced efficient data structures for tamper-evident logging [4].

- Crosby taught that it was pointless to have tamper resistant logs, if nobody ever looks at the logs to determine if they have been tampered with. Thus Crosby developed “tamper evident logs”

**Thus:**

- Crosby introduced the notion of a “commitment” which he calls a “snap shots” of the Audit Log up to a certain point in time

- Crosby assumes an “untrusted logger”, where he used the clients to verify that the “commitments” being provided by the logger are true
Overview of the Art - *Detecting Tampering of an Audit Log*

**Crosby method in a nutshell:**
- The “tamper evident log” is based on **Merkle trees**, where the leaves represented the data (events), and the roots contain hashes.

**Tree (or part of it) = a tamper evident summary of the data**

- **CLIENT** Requests to validate log history
- **CLIENT** compares from its history versus the pruned branch

Take new tree, delete nodes and rebuild – Do old (saved) and rebuilt hashes match?
Overview of the Art - *Detecting Tampering of an Audit Log*

**Crosby method in a nutshell:**
- The Merkle Tree nodes are essentially a series of *one-time signatures* (i.e., Lamport, etc.)

- Only data from “pruned trees” that contain the portion of the tree structure and related hashes being checked needs to be sent/checked

**Crosby further provides:**
- *Privacy preserving (“Private” search) by Audit Logging and exposing attributes about an event*, but *not the entire event contents itself*
Overview of the Art - Hierarchical Identity-Based Encryption for Audit Logs


Goyal uses “Hierarchical Identity-Based Encryption (HIBE). HIBE provides the ability to selectively decrypt Audit Log “attributes” based on the access control level privileges granted to a specific user [5].

- Thus provides privacy at a hierarchical access control level.

For example, the following attributes may have different access control levels, or overlapping access control levels, so that users may or may not decrypt some or all of the information:

- Name
- Date
- Source IP address
- Destination IP address
- Protocol
- Or other attribute based data

Goyal’s implementation is based on a “tree structure” where Goyal called the attributes “leaves”, and the nodes of the tree consisted of logical “AND”s and “OR”s related to access right privileges (e.g., based on leaves, a user is logically allowed or denied access).
Overview of the Art - **Hierarchical Identity-Based Encryption for Audit Logs**

**Goyal’s** Encryption/Decryption key allows privacy for a specific set of attributes, thus *preserves privacy by limiting access* to Audit Log data by those not authorized to see specific attributes:

\[ D = f(M, Pk, \gamma_1...\gamma_n) \]

Where:
- \( D \) = Decryption Key,
- \( M \) = Message
- \( Pk \) is the public key information generated from a Master Key (MK)
- \( \gamma_1...\gamma_n \) are the attributes (file accessed, OS system configuration changed, whatever….)

**Pro:**
- Provides some elements of ability to search on encrypted data (attributes) and privacy for the encrypted Message \( M \) and access level.

**Con:**
- While at the same time the disadvantage of the system is that *the set of attributes is sent in clear text.*

---

Secure Audit Logging Systems with Privacy Preserving - Richard Kramer – Oregon State University
The Current State – Secure Audit Logging Systems with Privacy Preserving


A good paper that addresses privacy preserving at the security level to augment the “Oh et al” paper (shown below).


A good paper on privacy preserving for Health Care Exchanges at the application level.
The Current State of the Art for Secure Audit Logging Systems – Privacy Preserving

Hartung (2014) – Builds on Crosby: “Secure Audit Logs with Verifiable Excerpts” or “SALVE” [6].

Hartung provides:

1) Verification - of an “Excerpt” is provided for BOTH:
   - Completeness
   - Correctness

2) Privacy preserving - in that only “Excerpts” of the log are audited, thus the remainder of the Audit Log remains private.
Audit Logging System Compromises

Schemes to secure Audit Logs using signatures have been broken

… and schemes using secret keys (sk) have been broken [6 at pg. 6]
Audit Logging System Compromises

Schemes to secure Audit Logs using signatures have been broken

... and schemes using secret keys (sk) have been broken [6 at pg. 6]

And the removal of log entries / tricks to accept modified logs or reordering message attacks are known [id.]

So counters and epoch markers have been added [id.]
Audit Logging System Compromises

Schemes to secure Audit Logs using signatures have been broken

... and schemes using secret keys (sk) have been broken [6 at pg. 6]

And the removal of log entries / tricks to accept modified logs or reordering ordering attacks are known [id.]

So counters and epoch markers have been added [id.]

And yet still, truncation attacks exist [id.]
The Current State of the Art for Secure Audit Logging Systems - Privacy Preserving

What about everyone else’s confidential information?

Entries made in log files … along with everyone else’s banking info too!

Hartung’s verifiable “Excerpts” solves the problem. Excerpts are Audit Log records that entail specific:

- “Categories” (e.g., Bank Account Opened, Deposit Make, Name, etc.)
- Epochs (T states) from one Audit Log message(s) entry state to the next
Security Scheme

In contrast to Hash Chains, Hartung “chains” together Secret Keys that are then used to create unique signatures.

- Each new “Secret Key” $Sk_i$ at state “i” is based on the prior Secret Keys ($Sk_{i-1}, Sk_{i-2} \ldots$) where previous keys are DELETED:

$$Sk_i = f (Sk_{i-1}, Sk_{i-2}, Sk_{i-3}, Sk_{i-4} \ldots)$$
Hartung calls his cryptology scheme a “Categorized Key-Evolving Audit Log Scheme”
Security Scheme

When verification of an excerpt is desired, the functions “EXTRACT” and “VERIFY” respectively create a unique signature for an excerpt and verifies the integrity of the excerpt as follows:

\[
\sigma', \text{Excerpt} \leftarrow \text{EXTRACT}(sk_i, M_{o,j}, \sigma_{0,j}, V), \text{ where “Extract” also produces a unique } pk \text{ (Public Key).}
\]

True / False \leftarrow \text{VERIFY}(pk, V, \text{Excerpt}, \sigma')

Where:
- \(sk_i\) is the Secret Key for epoch \(i\)
- \(M_{o,j}\) is the Message Log excerpt
- \(\sigma_{0,j}\) is the previous signature (\(\sigma_0 \ldots \sigma_j\)) that is created for a specific excerpt
- \(\sigma_E\) is the excerpt signature generated using the private key \(sk_i\)
- \(\sigma' = \sigma_{0\ldots j} \sigma_E\)
- \(V\) is a set of categories, named (\(v_0 \ldots v_j\)) for the excerpt (Bank Account Opened, Deposit Make, Name, etc.)
Example Scenario for Audit Log “$M$”

- **EM:0** = Epoch time marker at $T = 0$

- **EM:1** = Epoch time marker at $T = 1$

Categories ($v =$ name : $c_v =$ counter)

- $C_{ALL}(v_{ALL}: 0)$
- $C_1(v_1: 0)$
- $C_{ALL}(v_{ALL}: 1)$
- $C_1(v_1: 1)$
- $C_2(v_2: 0)$
- $C_{ALL}(v_{ALL}: 2)$
- $C_2(v_2: 1)$
- $C_n(v_n: 0)$

- **KeyGen($T$) $\rightarrow$** $sk_0, pk$

- **Update($sk_0, M, \sigma_0$) $\rightarrow$** $sk_1, pk$

- **AppendAndSign($sk_1, M, m_1, \sigma_0$) $\rightarrow$** $\sigma_1$, signs $\sigma_1$ with $sk_1$

- **AppendAndSign($sk_1, M, m_2, \sigma_0..1$) $\rightarrow$** $\sigma_2$, signs $\sigma_2$ with $sk_1$

- **AppendAndSign($sk_1, M, m_3, \sigma_0..2$) $\rightarrow$** $\sigma_3$, signs $\sigma_3$ with $sk_1$

- **new C**

- **new C**
Example Scenario for Audit Log “M”

\[ C_{ALL}(v_{ALL}: 0) \]
\[ C_{1}(v_{1}: 0) \]

\[ EM: 1 \]
= Epoch time marker at \( T = 1 \)

\[ C_{ALL}(v_{ALL}: 1) \]
\[ C_{1}(v_{1}: 1) \]
\[ C_{2}(v_{2}: 0) \]

\[ EM: 2 \]
= Epoch time marker at \( T = 2 \)

\[ C_{2}(v_{2}: 1) \]
\[ \ldots \]
\[ C_{n}(v_{n}: 0) \]

\[ C_{n}(v_{n}: 0) \]
\[ \ldots \]
\[ C_{ALL}(v_{ALL}: 2) \]
\[ C_{2}(v_{2}: 1) \]

\[ M1 \]
\[ \sigma 1 \]
\[ (sk1) \]

\[ M2 \]
\[ \sigma 2 \]
\[ (sk1) \]

\[ M3 \]
\[ \sigma 3 \]
\[ (sk1) \]

\[ Update(sk1, M, \sigma_{0..3}) \rightarrow sk2, pk \]

Switching to EM:2, sk2. Increment all counters:
\[ C_{ALL}(v_{ALL}: 3), C_{1}(v_{1}: 2), C_{2}(v_{2}: 2), C_{n}(v_{n}: 1) \]
**Example Scenario for Audit Log “M”**

**EM:1**

- = Epoch time marker at $T = 1$

**Categories (v = name : cv = counter)**

<table>
<thead>
<tr>
<th>Category</th>
<th>Value</th>
<th>Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>$C_{ALL}(v_{ALL}: 0)$</td>
<td>$C_I(v_I: 0)$</td>
<td>$m_1$</td>
</tr>
<tr>
<td>$C_{ALL}(v_{ALL}: 1)$</td>
<td>$C_I(v_I: 1)$</td>
<td>$m_2$</td>
</tr>
<tr>
<td>$C_{ALL}(v_{ALL}: 2)$</td>
<td>$C_2(v_2: 0)$</td>
<td>$m_3$</td>
</tr>
</tbody>
</table>

Switching to EM:2, $sk_2$. Increment all counters:

- $C_{ALL}(v_{ALL}: 3)$, $C_I(v_I: 2)$, $C_2(v_2: 2)$, $C_n(v_n: 1)$

**EM:2**

- = Epoch time marker at $T = 2$

**Excerpt**

$Extract(sk_2, m_{1..2}, \sigma_{1..2}, C_2) \rightarrow \sigma_E$ signed with $sk_2$

$Verify(pk, C_v, E, \sigma' (\sigma' = \sigma_{1..2}, \sigma_E)) \rightarrow True / False$
Performance

Pros:

- **Forward Integrity**
- **Privacy Preserving** when contrasting the entire Audit Log to an Excerpt

Cons:

- Seemingly **large** signature Audit Log file signature and Excerpt signature which concatenate previous signatures and is a function of message size and categories
- **Slower computational** time as compared to the more efficient BAF, LogFAS approaches (ECE 599 – Winter 2017 term).
The Current State of the Art for Secure Audit Logging Systems – Health Care Exchanges

Oh et al. provides a unique application using HIBE (Hierarchical Identity Based Encryption) – see Golay (2006) above

For the management and auditing of EHRs (Electronic Health Records) based on authorization “levels”.
Privacy Preserving Data Management

- Enhance security with Hierarchical Identity based Encryption (HIBE) to allow limited access to relevant external documentation

Authorization Level

\[ L3 = ID1 || ID2 || ID3 \]
\[ L2 = ID1 || ID2 \]
\[ L1 = ID1 \]

- Provider ID
- Name of Insurance Company
- Medical Service
- Insurance Plan
- Observation Type
- Observation Value (Personal Details)

Secure Audit Logging Systems with Privacy Preserving - Richard Kramer – Oregon State University [8,9]
Access rights are accomplished in layers and embedded within the cryptography system:

\[ \text{Cipher Text } Enc_{ID_i}(D_i) = HIBE.\text{Encrypt}(Pub, ID_i, D_i) \]

Where:

- \( D_i \) = Data for a specific level (\( D_1 \) is least sensitive, \( D_n \) is most sensitive)
- \( ID_i \) = The identity level, where \( ID_2 = id_1, id_2, \) and so on
- \( pk \) = Public parameters generated at the same time the Master Key is generated during setup
### Hierarchical Encryption

#### Billing Table

<table>
<thead>
<tr>
<th>patient-id</th>
<th>HCO-id</th>
<th>date-of-bill</th>
<th>Security Level</th>
</tr>
</thead>
<tbody>
<tr>
<td>eeb728473e1949a</td>
<td>Carle07RQ12</td>
<td>2013:09:08:10:18:41</td>
<td>$Enc_{ID_{1,1}}(cs1010)$, $Enc_{ID_{1,3}}(HEMOGLO...)$, $Enc_{ID_{1,3}}(73.8)$</td>
</tr>
<tr>
<td>d99486a44ca64cb</td>
<td>Provena01AV98</td>
<td>2013:09:17:02:48:29</td>
<td>$Enc_{ID_{2,1}}(ra1010)$, $Enc_{ID_{2,3}}(MCH,Auto...)$, $Enc_{ID_{2,3}}(279)$</td>
</tr>
<tr>
<td>42210b2417d74b1</td>
<td>NWM0329W2</td>
<td>2013:10:21:11:47:22</td>
<td>$Enc_{ID_{3,1}}(pq1010)$, $Enc_{ID_{3,3}}(PLATELET...)$, $Enc_{ID_{3,3}}(11.6)$</td>
</tr>
</tbody>
</table>

#### Authorization Access Levels

$ID_{\text{row#}, \text{level#}} = \text{Patient ID} \| \text{HCO ID} \| \text{Date of Medical Bill} \| \text{Sensitivity Level}$

- $ID_{1,3} = \text{eeb728473e1949a} \| \text{Carle07RQ12} \| 2013:09:08:10:18:41 \| \text{level1}$
- $ID_{1,2} = \text{eeb728473e1949a} \| \text{Carle07RQ12} \| 2013:09:08:10:18:41 \| \text{level1} \| \text{level2}$
- $ID_{1,1} = \text{eeb728473e1949a} \| \text{Carle07RQ12} \| 2013:09:08:10:18:41 \| \text{level1} \| \text{level2} \| \text{level3}$

The actual security methods to accomplish this are not addressed, thus Hartung and Goyal

[8,9]
The Current State of the Art for Secure Audit Logging Systems – *Health Care Exchanges*

Implements “Audit Trail and Node Authentication” (ATNA) as part of HIBE (Hierarchical Identify Based Encryption):

- ATNA not only logs events (e.g. a record has been accessed)
  - Part of Audit system is to determine and log *WHY* the record was accessed

- Uses an algorithm called **REDUCE**\(^1\) to look for log violations based on a “policy formula”

---

Uses **REDUCE** (basically an algorithm/policy language) with Explanations

\[\begin{align*}
\text{Conj clause} & : \quad C := \bigwedge_i \varphi_i \\
\text{Disj clause} & : \quad D := \bigvee_i \varphi_i \\
\text{Formula} & : \quad \alpha := \langle \ell \rangle P \mid \langle \ell \rangle T \mid \langle \ell \rangle \bot \mid \langle \ell \rangle C \mid \langle \ell \rangle D \\
& \quad \mid \langle \ell \rangle \forall \bar{x}. (c \supset \varphi) \mid \langle \ell \rangle \exists \bar{x}. (c \land \varphi) \\
& \quad \mid \sigma \triangleright \varphi \\
\text{Generalized form.} & : \quad \varphi := \alpha \mid \text{expl}(\top, \gamma) \mid \text{expl}(\bot, \gamma) \\
\text{Explanation} & : \quad \gamma := \ell \mid \ell \circ \gamma \mid \gamma_1 \oplus \gamma_2 \mid \sigma \triangleright \gamma
\end{align*}\]
Privacy Preserving Data Flow

- Verify legitimacy of access with logic-based audit algorithm

Explanation

<table>
<thead>
<tr>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Dr. Wagner accessed patient's record because Dr. Wagner was referred to.</td>
</tr>
<tr>
<td>Dr. Deboraski accessed patient's record because Dr. Deboraski was referred to.</td>
</tr>
<tr>
<td>NP Dyer accessed patient's record because NP Dyer prescribed patient</td>
</tr>
<tr>
<td>NP Dyer accessed patient's record because NP Dyer prescribed patient</td>
</tr>
</tbody>
</table>
Audit data Collector (AC) Path:  
1. ATNA logs  
2. External documentations  
3. Encrypted external doc and ATNA logs  

Access Analysis (AA) Path:  
1. Provider ID, Patient ID and Event time  
2. Provider ID, Patient ID and Event time  
3. SQLITE database  

Supplement Resolution (SR) Path:  
1. Residual policy  
2. ID(s)  
3. Secret key(s)  
4. SQLITE database  

Explanation Creator (EC) Path:  
1. Explanations  
2. Human-readable explanations  
3. Report
Audit Algorithm

\[ \text{REDUCE}[\mathcal{L}(\log), \varphi_n \text{ (privacy preserving policy n)}] = \varphi'_n \text{ (output)} \]
Policy Logic

Providers requesting or accessing to an EHR for “treatment” needs to be verified if a relationship exists between p1, p2 and q and authorization level to see information in the medical bill.

\[ \varphi_{pol} = \langle DISC \rangle \]

\[ \forall p_1, p_2, m, q, t, ty, va, tp, vl, o, p, c \]

Restriction (c)

Formula (\( \varphi \))

Level 1
Level 2
Level 3
Example Scenario

\[ \varphi_0 = \forall p1, p2, m, q, t, ty, va, tp, vl, o, p, c \\
(\text{send}(p1, p2, m, t) \land \text{tagged}(m, q) \land \text{includes}(m, ty, va, t) \\
\land \text{patientInfo}(q, tp, vl, t) \land \text{organization}(p2, o, t) \\
\land \text{insuranceInfo}(q, p, c, t)) \\
\Rightarrow \exists t'. \text{medical-bill}(q, b, t') \\
\land ((\text{time-in}(t, t', t+365) \\
\land \text{insurance}(q, p, c, o, t') \\
\land \text{visits-in-bill}(q, p2, vl, o, t') \\
\lor \text{observes-in-bill}(q, p2, ty, va, o, t'))) \]
Example Scenario

Audit Algorithm

\[ \varphi_0 = \forall p1, p2, m, q, t, ty, va, tp, vl, o, p, c \]
\[ (send(p1, p2, m, t) \land tagged(m, q) \land includes(m, ty, va, t) \]
\[ \land patientInfo(q, tp, vl, t) \land organization(p2, o, t) \]
\[ \land insuranceInfo(q, p, c, t)) \]
\[ \supset \exists t'. \text{medical-bill}(q, b, t') \]
\[ \land ((time-in(t, t', t+365) \land \]
\[ \varphi_1 = \exists insurance(q, p, c, o, t') \]
\[ \land (visits-in-bill(q, p2, vl, o, t') \]
\[ \lor observes-in-bill(q, p2, ty, va, o, t'))) \]

Audit Agent

Secure Audit Logging Systems with Privacy Preserving - Richard Kramer – Oregon State University
Example Scenario

Audit Algorithm

\[ \varphi_1 = \text{insurance}(\text{Grace}, \text{PPO}, \text{WA02}, \text{CARLE}, \text{DOA}) \land (\text{visits-in-bill}(\text{Grace}, \text{Kosta}, \text{O}, \text{CARLE}, \text{DOB}) \lor \text{observes-in-bill}(\text{Grace}, \text{Kosta}, \text{NERVOUS SYSTEM, DEPRESSION, CARLE, DOB})) \]

\[ \varphi_0 = \forall p1, p2, m, q, t, ty, va, tp, vl, o, p, c \]
\[ \land (\text{send}(p1, p2, m, t) \land \text{tagged}(m, q) \land \text{includes}(m, ty, va, t)) \]
\[ \land \text{patientInfo}(q, tp, vl, t) \land \text{organization}(p2, o, t) \]
\[ \land \text{insuranceInfo}(q, p, c, t)) \]
\[ \lor \exists t'. \text{medical-bill}(q, b, t') \]
\[ \land ((\text{time-in}(t, t', t+365)) \land \text{insurance}(q, p, c, o, t')) \]
\[ \land (\text{visits-in-bill}(q, p2, vl, o, t') \lor \text{observes-in-bill}(q, p2, ty, va, o, t'))) \]
Example Scenario

Audit Algorithm

\[ \varphi_2 = \langle \text{POL/DISCLOSE} \rangle \text{True} \]

\[ (\text{send}(\text{Smith}, \text{Kosta}, \text{D1}, \text{date-of-access}) \land \text{tagged}(\text{D1}, \text{Grace}) \land \ldots \land \text{visits-in-bill}(\text{Grace}, \text{Kosta}, \text{O,CARLE,DOB})) \]

Dr. Kosta accessed Grace’s record on date-of-access because Grace visited Dr. Kosta

\[ \varphi_1 = \]

\[ \forall p1, p2, m, q, t, ty, va, tp, vl, o, p, c \]

\[ (\text{send}(p1, p2, m, t) \land \text{tagged}(m, q) \land \text{includes}(m, ty, va, t) \land \text{patientInfo}(q, tp, vl, t) \land \text{organization}(p2, o, t) \land \text{insuranceInfo}(q, p, c, t)) \]

\[ \supset \exists t'. \text{medical-bill}(q, b, t') \]

\[ \land ((\text{time-in}(t, t', t+365)) \land \text{insurance}(q, p, c, o, t') \land (\text{visits-in-bill}(q, p2, vl, o, t') \lor \text{observes-in-bill}(q, p2, ty, va, o, t'))) \]
The Current State of the Art for Secure Audit Logging Systems – *Health Care Exchanges*

**Oh et al (2014) – Summary**

**Pro:**
- Solid application of HIBE in Health Care to solve a clear problem.
- Preserves privacy within the Audit Logging system domain.

**Cons / Future work:**
- Security appears to only be guaranteed within the Audit Logging system domain, not back to the original sources.
- A need exists to secure against potential alteration of the Audit Log including securing the explanation log, and the policies used to interrogate the Audit Log.
Future Work

Research such papers as Attila A. Yavuz, Jorge Guajardo, “Dynamic Symmetric Searchable Encryption with Minimal Leakage and Efficient Updates on Commodity Hardware” for future applications in health care Audit Logging.

For example, files represent patient records, and attributes/key words can be used for searchable items such as a composite set of addresses, bills due, etc.

<table>
<thead>
<tr>
<th>$I'$</th>
<th>$I$</th>
<th>...</th>
<th>$n$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$l$</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>...</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$I$</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$m$</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Each Patient’s EHR

Patient Attributes = bill due, amount, address, etc.
Audit Logging Tools and Systems

Secure Audit Log Tools (e.g., What kind of secure audit log tools are available in the literature?).

The below is a list of Audit Logging Tools:

<table>
<thead>
<tr>
<th>Number</th>
<th>Product / Company Name</th>
<th>Link</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Splunk (Free download/trial)</td>
<td><a href="https://www.splunk.com/en_us/download-5.html">https://www.splunk.com/en_us/download-5.html</a></td>
</tr>
<tr>
<td>3</td>
<td>ipswitch (was WhatsUpGold)</td>
<td><a href="https://www.ipswitch.com/solutions/log-and-event-management">https://www.ipswitch.com/solutions/log-and-event-management</a></td>
</tr>
<tr>
<td>4</td>
<td>TIBCO</td>
<td><a href="http://www.tibco.com/products/event-processing/loglogic-for-machine-data">http://www.tibco.com/products/event-processing/loglogic-for-machine-data</a></td>
</tr>
<tr>
<td>6</td>
<td>SolarWinds Log &amp; Event Manager (LEM)</td>
<td><a href="http://www.solarwinds.com/log-event-manager">http://www.solarwinds.com/log-event-manager</a></td>
</tr>
<tr>
<td>7</td>
<td>ManageEngine EventLogAnalyzer</td>
<td><a href="https://www.manageengine.com/products/eventlog/">https://www.manageengine.com/products/eventlog/</a></td>
</tr>
<tr>
<td>8</td>
<td>Tripwire</td>
<td><a href="http://www.tripwire.com/">http://www.tripwire.com/</a></td>
</tr>
<tr>
<td>9</td>
<td>NetIQ</td>
<td><a href="https://www.netiq.com/products/sentinel-log-manager/">https://www.netiq.com/products/sentinel-log-manager/</a></td>
</tr>
</tbody>
</table>

Updated list of Audit Log tools based on the 2014 article “Top 47 Log Management Tool” at link: https://blog.profitbricks.com/top-47-log-management-tools/ [10]:

Secure Audit Logging Systems with Privacy Preserving - Richard Kramer – Oregon State University
## Audit Logging Tools and Systems

<table>
<thead>
<tr>
<th>Number</th>
<th>Product / Company Name</th>
<th>Link:</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>InTrust / Dell Software</td>
<td><a href="https://software.dell.com/products/intrust/">https://software.dell.com/products/intrust/</a></td>
</tr>
<tr>
<td>11</td>
<td>Veriato (was SpectorSoft)</td>
<td><a href="http://www.veriato.com/products/veriato-server-manager">http://www.veriato.com/products/veriato-server-manager</a></td>
</tr>
<tr>
<td>13</td>
<td>LogRhythm</td>
<td><a href="https://logrhythm.com/index.html">https://logrhythm.com/index.html</a></td>
</tr>
<tr>
<td>14</td>
<td>TNT Software (was ELM Enterprise Manager)</td>
<td><a href="https://tntsoftware.com/">https://tntsoftware.com/</a></td>
</tr>
<tr>
<td>16</td>
<td>Netwrix Auditor</td>
<td><a href="https://www.netwrix.com/event_log_management.html">https://www.netwrix.com/event_log_management.html</a></td>
</tr>
<tr>
<td>18</td>
<td>Sumo Logic</td>
<td><a href="https://www.sumologic.com/application/">https://www.sumologic.com/application/</a></td>
</tr>
<tr>
<td>19</td>
<td>Novell Sentinel Log Manager – Merged with NetIQ, see above</td>
<td><a href="https://www.netiq.com/products/sentinel-log-manager/">https://www.netiq.com/products/sentinel-log-manager/</a></td>
</tr>
<tr>
<td>24</td>
<td>Elasticsearch ELK Stack</td>
<td><a href="https://www.elastic.co/products">https://www.elastic.co/products</a></td>
</tr>
<tr>
<td>25</td>
<td>Logscape</td>
<td><a href="http://logscape.com/">http://logscape.com/</a></td>
</tr>
<tr>
<td>26</td>
<td>Sawmill</td>
<td><a href="https://twitter.com/Sawmill">https://twitter.com/Sawmill</a></td>
</tr>
<tr>
<td>27</td>
<td>Event Sentry</td>
<td><a href="http://www.eventsentry.com/">http://www.eventsentry.com/</a></td>
</tr>
<tr>
<td>28</td>
<td>BalaBit syslog-ng</td>
<td><a href="https://www.balabit.com/network-security/syslog-ng">https://www.balabit.com/network-security/syslog-ng</a></td>
</tr>
</tbody>
</table>
## Audit Logging Tools and Systems

<table>
<thead>
<tr>
<th>Number</th>
<th>Product / Company Name</th>
<th>Link:</th>
</tr>
</thead>
<tbody>
<tr>
<td>29</td>
<td>CorreLog</td>
<td><a href="https://correlog.com/?vsmaid=35">https://correlog.com/?vsmaid=35</a></td>
</tr>
<tr>
<td>30</td>
<td>Papertrail</td>
<td><a href="https://papertrailapp.com/">https://papertrailapp.com/</a></td>
</tr>
<tr>
<td>34</td>
<td>SemaText Logsene</td>
<td><a href="https://sematext.com/logsene/">https://sematext.com/logsene/</a></td>
</tr>
<tr>
<td>37</td>
<td>LOGalyze</td>
<td><a href="http://www.logalyze.com/">http://www.logalyze.com/</a></td>
</tr>
<tr>
<td>43</td>
<td>Scalyr</td>
<td><a href="https://www.scalyr.com/?gclid=CODq0sK9t74CFfe47MgoddzQAaA">https://www.scalyr.com/?gclid=CODq0sK9t74CFfe47MgoddzQAaA</a></td>
</tr>
<tr>
<td>44</td>
<td>Graylog2</td>
<td><a href="https://www.graylog.org/">https://www.graylog.org/</a></td>
</tr>
</tbody>
</table>

Secure Audit Logging Systems with Privacy Preserving - Richard Kramer – Oregon State University
Application Videos/ Demo Links

Tripwire:

https://www.demochimp.com/app/view/p/8ffjhbx7

(check as “very important” = “Integrity Monitoring” and “Policy Management”, and check others as “Not Important”)

Splunk:


References

Background / Historical References:


Primary References:


Tools References:

Questions?

THANK YOU!