

Data Authentication

Ashish Kundu
 cs.purdue.edu/~ashishk

CS 426
 Oct 26, 2009



Outline

- Message Integrity
- Message Authentication
- Complex Messages
- Merkle Hash Technique
- Structural Signatures

2

Message Integrity

- Message $M \in \{0,1\}^*$
- $S \rightarrow R$:
 - $M, H(M)$
 - $H(M)$ can be used to verify if integrity of M is preserved or not.
 - However, does not guarantee where M comes from?
- $S \rightarrow \text{Attacker} \rightarrow R$
 - $M', H(M')$
 - R cannot know whether it received the message that was supposed to come from S .

3

Message Authentication

- MAC, HMAC
 - Rely on symmetric key encryption
- Digital signature (public key)
 - Sign M : $\text{sig}(M)$
 - Does it work?
 - Modify M and $\text{sig}(M)$ such that $\text{sig}(M')$ is a valid signature of M' .
 - Unpadded RSA, Malleable encryption
 - Always sign the hash of a message $H(M)$
 - Sign $H(M)$: Computationally infeasible to modify $H(M)$ in a valid way.

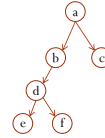
4

Complex Messages

- Set of messages
 - Ex: Online poker, online chess, online games, etc.
- Multi-set: multiple occurrence of the same element
 - Ex: Arrays, Blocks of data
- List: has an order between elements in a multi-set
 - Ex: Arrays, Blocks of data
- Trees
 - XML, Website, Webpage

5

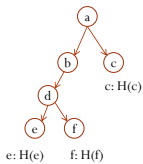
Merkle Hash Technique (MHT)



Case: **Only leaf nodes have contents.**

6

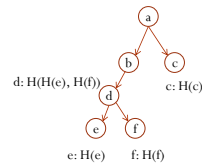
Merkle Hash Technique (MHT)



Case: **Only leaf nodes have contents.**

7

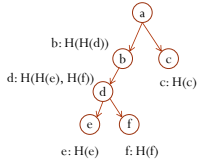
Merkle Hash Technique (MHT)



Case: **Only leaf nodes have contents.**

8

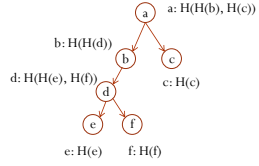
Merkle Hash Technique (MHT)



Case: **Only leaf nodes have contents.**

9

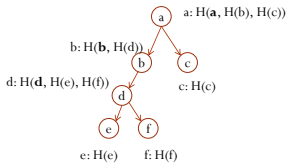
Merkle Hash Technique (MHT)



Case: **Only leaf nodes have contents.**

10

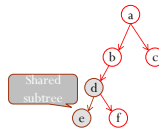
Merkle Hash Technique (MHT)



Case: **All nodes have contents.**

11

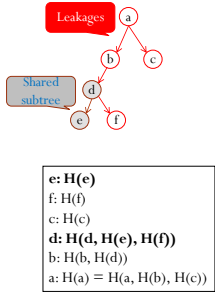
Extent of Leakage in MHT



- e: $H(e)$
- f: $H(f)$
- c: $H(c)$
- d: $H(d, H(e), H(f))$
- b: $H(b, H(d))$
- a: $H(a, H(b), H(c))$

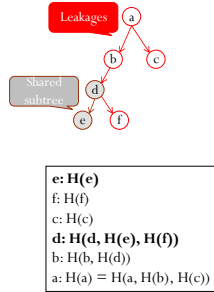
12

Extent of Leakage in MHT



13

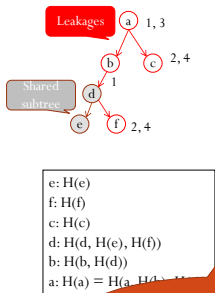
Extent of Leakage in MHT



14

- **Direct Leakages**
 1. Merkle hash (MH) of the root
 2. MH of each sibling of a received node
 3. Hash of the content of each ancestor
 4. MH of a child of each ancestor
- **Indirect (Inferred) Leakages**
 - I. (1) => if received subtree is (or is not) the complete tree (**Comparison attack**)
 - II. (2) =>
 - a. **Ariety of the tree**
 - b. **Number of children of each ancestor**
 - III. (2) or (3) or (4) => If a hidden subtree same as a received subtree (**Comparison attack**)

Extent of Leakage in MHT

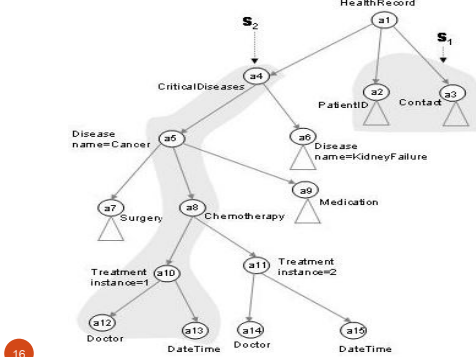


15

- **Direct Leakages**
 1. Merkle hash (MH) of the root
 2. MH of each sibling of a received node
 3. Hash of the content of each ancestor
 4. MH of a child of each ancestor
- **Indirect (Inferred) Leakages**
 - I. (1) => if received subtree is (or is not) the complete tree (**Comparison attack**)
 - II. (2) =>
 - a. **Ariety of the tree**
 - b. **Number of children of each ancestor**
 - III. (2) or (3) or (4) => If a hidden subtree same as a received subtree (**Comparison attack**)

Leakages – How significant/sensitive?

Motivating Example



16

Motivating Example: Semantic Leaks

Linked information during verification of a node	Inference from the linkage in the health-care context
signature of e_{12} AND (e_{13} is sibling of e_{10} OR e_{13} is child of e_{10})	Patient has gone through another Chemotherapy.
e_{13} is to the right of e_{10}	If sibling order represents more information can be derived such as it can be inferred if the chemotherapy referred to by node e_{13} was administered earlier or later than the one referred to by e_{10} .
signature of e_9 AND (e_9 is sibling of e_7 OR e_9 is child of e_7)	Patient has gone through another type of treatment; also inferred is - it to be either Surgery or Medication
e_9 is to the left of e_7	More linkage related to the order such as temporal order
signature of e_8 AND (e_8 is sibling of e_7 OR e_8 is child of e_7)	Patient has gone through another type of treatment; also inferred is - it to be either Surgery or Medication
e_8 is to the right of e_7	More linkage related to the order such as temporal order
signature of e_5 AND (e_5 is sibling of e_2 OR e_5 is child of e_2)	Patient suffers from another critical disease; can be determined which disease it is from the specialty of the hospital
e_5 is to the right of e_2	More linkage related to the order such as temporal order; time of treatment of this disease in this hospital relative to the time of treatment of Cancer

Problem

e: H(e)
f: H(f)
c: H(c)
d: H(d, H(e), H(f))
b: H(b, H(d))
a: H(a) = H(a, H(b), H(c))

Problem Definition: Signature for Trees

- Guarantee integrity of any subtree S in a tree T, while preventing the following information leakages

19

Problem Definition: Signature for Trees

- Guarantee integrity of any subtree S in a tree T, while preventing the following information leakages
 - Signature of a node y in T but not in S
 - Existence of node y in T but not in S
 - Structural relationships not present in the subtree
 - For ordered trees, structural order between a node x in S and y in T but not in S

20

Contributions

- Structural Signatures for Trees

21

Contributions

- Structural Signatures for Trees
 - Provably binding
 - Integrity verification of trees – content, structure
 - Provably hiding
 - Prevents leakage
 - More efficient than Merkle hash technique

22

Structural Signatures

23

Some Simple Observations

24

Randomized Traversal Numbers

- Traversal numbers are not secure:
 - Uniform distance, [1...n]
- Randomized traversal numbers
 - Random distance, Random numbers
 - Order is preserved

29

Randomized Traversal Numbers

- Traversal numbers are not secure:
 - Uniform distance, [1...n]
- Randomized traversal numbers
 - Random distance, Random numbers
 - Order is preserved
- Randomized Post-order Numbers (RPON)
- Randomized Pre-order Numbers (RRON)
- Randomized In-order Numbers (RION)

30

Randomized Traversal Numbers

- Technique-1: Sort N random numbers, N = number of nodes
- Technique-3: Order-preserving encryption
 - Rakesh Agarwal et al.
- Technique-2: $R(i) = R(i-1) + w(i)$, $w(i) = \text{random}$
 $R(0) = w(0)$
 - Is it secure?

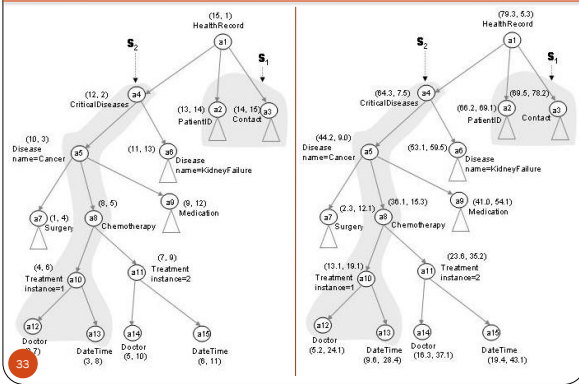
31

Randomized Traversal Numbers

- Technique-1: Sort N random numbers, N = number of nodes
- Technique-3: Order-preserving encryption
 - Rakesh Agarwal et al.
- Technique-2: $R(i) = R(i-1) + w(i)$, $w(i) = \text{random}$
 $R(0) = w(0)$
 - Is it secure?

32

Randomized Traversal Numbers



Notations

- Tree: $T(V, E)$
- $x, y, z \in V$
- p_x : RPON of x
- r_x : RRON of x
- C_x : Content of x
- $|$: concatenation operation
- h : one-way collision-resistant cryptographic hash

Structural Signatures

- Structural Position of a Node 'x':
 - $S_x = (p_x, r_x)$.
- Structural Signature of a Tree 'T':
 - $G_T = h(S_1 | C_1 | S_2 | C_2 | \dots | S_n | C_n)$.
- Structural Signature of a Node 'x':
 - $G_x = h(G_T | S_x | C_x)$.

Signing a Tree

- Signing the tree by Alice:
 - Signs the tree T
 - Certifies the signatures itself or by a trusted CA.
 - Shares the T with distributors (D).

Signing a Tree $T(V, E)$

1. Compute the post-order and pre-order numbers for each node in T .
2. For each node x in V : transform the post-order and pre-order numbers into randomized post-order and pre-order numbers, such that
 - a. for unordered trees, RPON's and RRON's among the siblings do not need to preserve any order;
 - b. for ordered trees, RPON's and RRON's for all nodes, need to preserve the order.
3. Assign (p_x, r_x) to x as its structural position S_x .

37

Signing a Tree $T(V, E)$

4. Compute the structural signature of the tree T , G_T
 - $G_T = h(S_1 | C_1 | S_2 | C_2 | \dots | S_n | C_n | \dots | S_n | C_n)$.
5. For each node x in V : compute the signature, G_x
 - $G_x = h(G_T, S_x, C_x)$.

38

Sharing the Tree

- D sends a subtree S to Bob
 - Two strategies for sending S
 - Complete structure of S
 - Only nodes of S

39

Sharing the Tree

- D sends a subtree S to Bob
 - Two strategies for sending S
 - Complete structure of S
 - Only nodes of S
 - Signature of tree T : G_T
 - Signatures of nodes in S
 - Digest $_s = h(G_1 | G_2 | \dots | G_{|s|})$.

40

Sharing the Tree

- D sends a subtree S to Bob
 - Two strategies for sending S
 - Complete structure of S
 - Only nodes of S
 - Signature of tree T : G_T
 - Signatures of nodes in S
 - $\text{Digest}_S = h(G_1 | G_2 | \dots | G_{|S|})$.
- Bob verifies integrity of S .

41

Integrity

Sharing Complete Structure of S

Bob receives: $R(V_r, E_r)$.

1. Verifies the certificate of each node in V_r
2. Verifies content integrity
3. Verifies structural integrity
4. If all nodes are valid, Bob computes Digest_r
 - a. If $(\text{Digest}_r = \text{Digest}_S)$ then no node is dropped.

42

Integrity

Sharing Complete Structure of S

Bob receives: $R(V_r, E_r)$.

1. Verifies the certificate of each node in V_r
2. Verifies content integrity
3. Verifies structural integrity
4. If all nodes are valid, Bob computes Digest_r
 - a. If $(\text{Digest}_r = \text{Digest}_S)$ then no node is dropped.

Integrity Verification for Content

1. For each node y in V_r
 - a. $\text{temp} = h(G_T | S_y | C_y)$
2. If temp is same as G_y , then the integrity of the node y is verified.

44

Integrity

Sharing Complete Structure of S

Bob receives: $R(V_r, E_r)$.

1. Verifies the certificate of each node in V_r
2. Verifies content integrity
3. Verifies structural integrity
4. If all nodes are valid, Bob computes Digest_r
 - a. If $(\text{Digest}_r = \text{Digest}_S)$ then no node is dropped.

Integrity Verification for Content

1. For each node y in V_r
 - a. $\text{temp} = h(G_T | S_y | C_y)$
2. If temp is same as G_y , then the integrity of the node y is verified.

45

Structural
Signature of y

Structural Integrity

Integrity Verification for Structural Relations

1. Carry out a pre-order traversal on R .
2. Let x be the parent of z ;
 - a. if $((p_x \leq p_z) \text{ or } (r_x \geq r_z))$, then parent-child relationship between x and z is **incorrect**.

46

Structural Integrity

Integrity Verification for Structural Relations

1. Carry out a pre-order traversal on R .
2. Let x be the parent of z ;
 - a. if $((p_x \leq p_z) \text{ or } (r_x \geq r_z))$, then parent-child relationship between x and z is **incorrect**.
3. For ordered trees, let y be the right sibling of z ;
 - a. if $((p_z \geq p_y) \text{ or } (r_z \geq r_y))$, then the left-right order among the siblings y and z is **incorrect**.

47

Sharing only the Nodes of S

Bob receives: V_r

1. Verify the certificate of each node in V_r .
2. Verify the integrity of content of each node.
3. Reconstructs the subtree $R(V_r, E_r)$ using Structural Positions
 - a. Any relation (edge) with an invalid node: invalid.
4. If all nodes are valid, Bob computes Digest_r
 - a. If $(\text{Digest}_r = \text{Digest}_s)$ then no node is dropped.

48

Security Analysis

- **Provably Binding**
 - Structural signatures detect violation(s) to integrity of content and structure.

49

Security Analysis

- **Provably Binding**
 - Structural signatures detect violation(s) to integrity of content and structure.
- **Provably Hiding**
 - Structural signatures do not leak (or leak negligible amount of information about)

50

Complexity Analysis

- **Cost of Signature Generation**
 - Time: $O(|V|)$
 - Storage: $O(|V|)$ ($= O(|S_x| + |G_x| + |G_T|)$)

52

Complexity Analysis

- **Cost of Signature Generation**
 - Time: $O(|V|)$
 - Storage: $O(|V|)$ ($= O(|S_x| + |G_x| + |G_T|)$)
- **Cost of Distribution**
 - **Substructure**
 - Communication: $2|V|-1$, Storage: $2|V|-1$
 - **Only the Nodes: 50% reduction**
 - Communication: $|V|$, Storage: $|V|$

53

Complexity Analysis

- **Cost of Signature Generation**
 - Time: $O(|V|)$
 - Storage: $O(|V|)$ ($= O(|S_x| + |G_x| + |G_T|)$)
- **Cost of Distribution**
 - **Substructure**
 - Communication: $2|V|-1$, Storage: $2|V|-1$
 - **Only the Nodes: 50% reduction**
 - Communication: $|V|$, Storage: $|V|$
- **Cost of Integrity Verification**
 - Time: $O(1)$ for each node, structural relation/order.
 - Time: $O(|V_R|)$, for the received subtree

54

Complexity Analysis

Time: identical to MHT
Storage: constant factor more than MHT

- Cost of Signature Generation
 - Time: $O(|V|)$, Storage: $O(|V|)$ ($\equiv O(|S_x| + |G_x| + |G_T|)$)
- Cost of Distribution More efficient than MHT
 - Substructure
 - Communication: $2|V|-1$, Storage: $2|V|-1$
 - Only the Nodes: 50% reduction
 - Communication: $|V|$, Storage: $|V|$
- Cost of Integrity Verification More efficient than MHT
 - Time: $O(1)$ for each node, structural relation/order.
 - Time: $O(|V_R|)$, for the received subtree

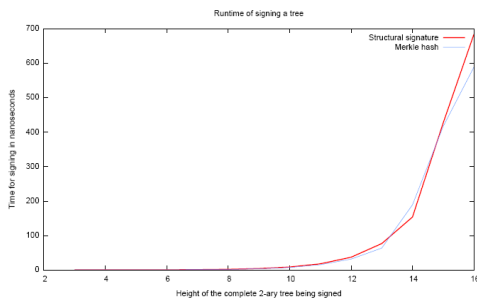
56

Performance: Infrastructure

- Trees
 - 0-65535 nodes, 2-ary ordered complete trees,
 - height 1-16.
- Language
 - Java J2SE5.0
- Machine
 - IBMT42 512MB RAM

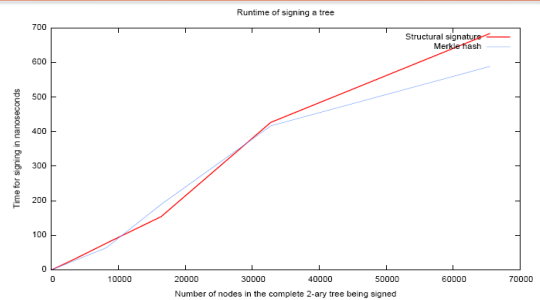
56

Signing: Height Vs. Time (Nano-secs)



57

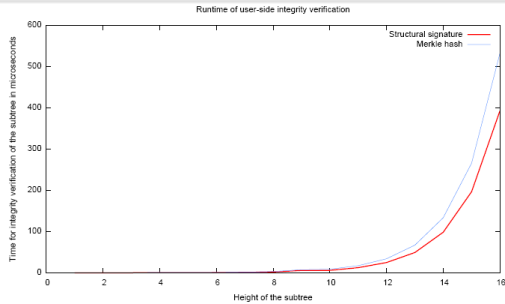
Signing: No. of Nodes Vs. Time (Nano-secs)



- Time taken more than MHT: 0.13 seconds
- Negligible: Signed once, used many times.

58

Integrity: Height Vs. Time (Micro-secs)

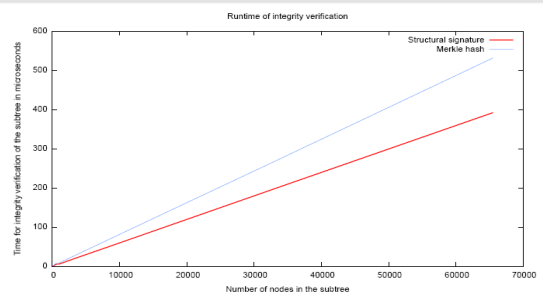


- More efficient than MHT

Verified by many users: so gain is multiplied.

59

Integrity: No. of nodes Vs. Time (Micro-secs)



- More efficient than MHT

Verified by many users: so gain is multiplied.

60

Dynamic Updates

- Node/subtree insertion to T: node signature using G_T
- Deletion: G_T is not updated
- Edge insertion/deletion

61

Dynamic Updates

- Node/subtree insertion: node signature using G_T
 - $\triangleright O(1)$ per node
- Deletion: no change to G_T
 - $\triangleright O(1)$ per node
- Edge insertion/deletion
 - $\triangleright O(1)$ per edge

62

Dynamic Updates

MHT and related techniques:
Updates: $O(\log(n))$ per node/edge

- Node/subtree insertion: node signature using G_T
 - $O(1)$ per node
- Deletion: no change to G_T
 - $O(1)$ per node
- Edge insertion/deletion
 - $O(1)$ per edge

63

Dynamic Updates

- Node/subtree insertion: node signature using G_T
 - $O(1)$ per node
- Deletion: no change to G_T
 - $O(1)$ per node
- Edge insertion/deletion
 - $O(1)$ per edge

Use of G_T (signature of tree T) for the updated tree T' :
Advantage: Prevents leakage related to updates
Drawback: G_T does not reflect updates

64

Summary

- Stronger security guarantees than MHT and related techniques
 - Hiding
 - Binding
- More efficient
 - Distribution
 - Integrity verification
 - Similar cost for signing process
- Simple – an important security criteria
 - To understand and implement

65

Summary

- Stronger security guarantees than MHT and related techniques
 - Hiding
 - Binding
- More efficient
 - Distribution
 - Integrity verification
 - Similar cost for signing process
- Simple – an important security criteria
 - To understand and implement

Pervasive Devices/Applications

66

Some Applications

- Database integrity & confidentiality
- Index integrity & confidentiality
- XML Signatures: Integrity & confidentiality of XML
- Memory integrity
- Directory signatures

67

Related Work: Authentication of Trees

- [Merkle'79] Merkle hash technique (MHT): widely used
 - [Kundu & Bertino'06] **Leaks** content and structural information:
 - $O(\log(n))$, n : no. of vertices
 - [Buldas & Laur'07] **binding but not hiding**;
 - no solution proposed.
- Well-known applications of Merkle hash technique
 - [Pang & Tan'08] Verifying completeness of relational query answers
 - [Martel et al'04] Search DAGS
 - [Bertino et al'04] Selective & authentic third party XML Dissemination
 - [Devanbu et al'01, '03] XML authentication

68

Related Work: Authent

What is leaked?

- [Merkle'79] Merkle hash technique (MHT): Widely used
 - **Leaks** content and structural information: $O(\log(n))$, n : no. of vertices
 - [Buldas & Laur'07] **binding but not hiding**;
 - no solution proposed.
- Well-known applications of Merkle hash technique:
 - [Pang & Tan'08] Verifying completeness of relational query answers
 - [Martel et al'04] Search DAGS
 - [Bertino et al'04] Selective & authentic third party XML Dissemination
 - [Devanbu et al'01, '03] XML authentication

69

Related Work: Information Leakages

- [Wang et al, VLDB'05]
 - Structural relations (edges) in XML: Not accessible by a user
 - Does not address leakages due to integrity verification
- Active area of research: last two decades
 - Privacy-preserving databases
 - [Chatvichienchai et al, DEXA'06] [Rastogi et al, VLDB'07] [Wong et al, VLDB'07] [Zhang and Zhao, VLDB'05]
 - [Irwin & Yu, CCS'05] Automated Trust Negotiation
 - [Dodis & Smith, STOC'05] Error correction
 - [Chatvichienchai et al, DEXA'06] Information leakage in updating XML
 - Leakage from integrity verification of XML – not addressed

70

and questions?

Ashish Kundu
www.cs.purdue.edu/~ashishk

74