

# Fake Picassos, Tampered History, and Digital Forgery: Protecting the Genealogy of Bits with Secure Provenance



**Ragib Hasan**

Joint work with **Radu Sion** (Stony Brook) and **Marianne Winslett** (UIUC)  
Dept. of Computer Science  
Johns Hopkins University

CS 690 SBU/UIUC  
October 28, 2009

# Let's play a game



Real, worth **\$101.8** million



**Fake**, listed at eBay,  
worth nothing

Can you spot the fake **Picasso**?

# So, how do art buyers authenticate art?

Among other things, they look at **Provenance records**



**Provenance:** from Latin *provenire* ‘come from’, defined as

*“(i) the fact of coming from some particular source or quarter; origin, derivation.*

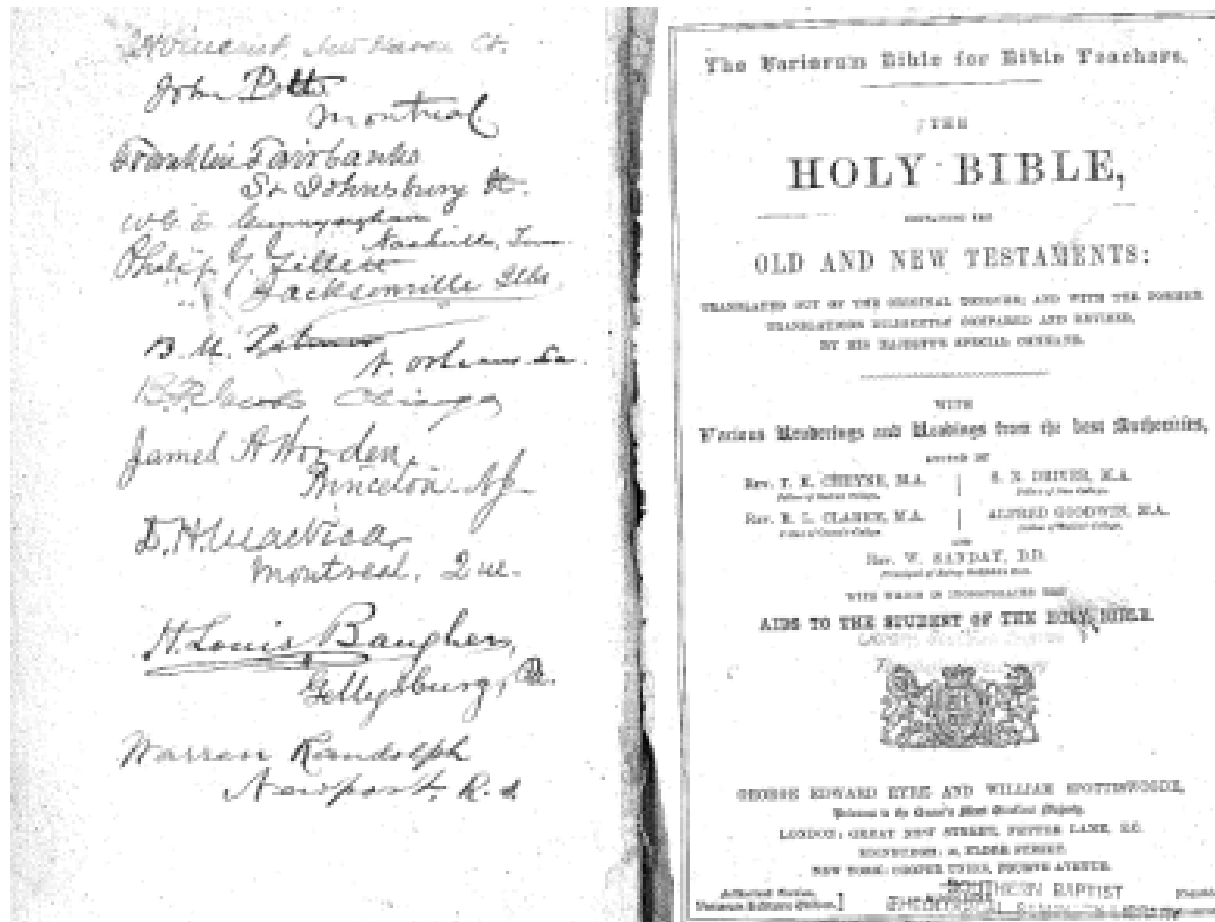
*“(ii) the history or pedigree of a work of art, manuscript, rare book, etc.; a record of the ultimate derivation and passage of an item through its various owners” (Oxford English Dictionary)*

In other words, **Who owned it, what was done to it, how was it transferred ...**

Widely used in arts, archives, and archeology, called the Fundamental Principle of Archival

L'artiste et son modèle (1928), at Museum of Modern Art

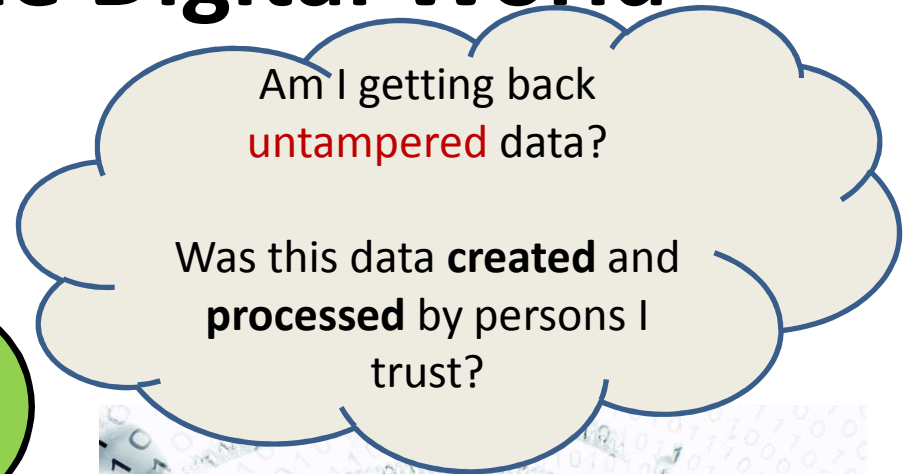
<http://moma.org/collection/provenance/items/644.67.html>



## Provenance of a book

Owner names recorded in sequential order on left side

# Let's consider the Digital World



Unlike data processing in the past, digital data is generated, processed, and stored in a digital format. To trust data we receive from a server or retrieve from digital storage we need to look into the integrity of both: the **present state** and the **past history** of data.

# What exactly is Data Provenance?

- Definition\*
  - Description of the **origins** of data and the **process** by which it arrived at the database. [Buneman et al.]
  - Information describing materials and **transformations** applied to derive the data. [Lanter]
  - Metadata recording the **process of experiment workflows**, annotations, and notes about experiments. [Greenwood]
  - Information that helps determine the **derivation history** of a data product, starting from its original sources. [Simmhan et al.]

\*Simmhan et al. A Survey of Provenance in E-Science. SIGMOD Record, 2005.

# Usage of Provenance

- **Data Quality** – depends on input data and actions taken on data
- **Replication** – a recipe for recreating data
- **Attribution** – proper attribution of copyright
- **Informational** – learn how data was derived
- **Audit Trails** – proof of due diligence; investigation of problem sources, system intrusions, proof of prior work in patents

# Example provenance systems

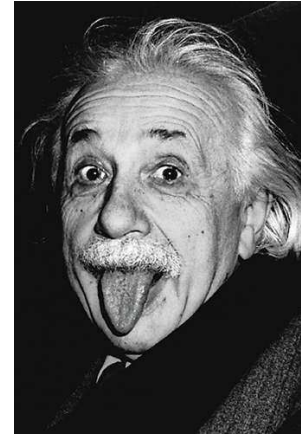
	Lanter, D. P. (LIP)	Chimera	MyGRID	CMCS	PASOA	ESSW	Tioga	Buneman, P.	Cui, Y., Widom, J. (Trio)
Applied Domain	GIS	Physics, Astronomy	Biology	Chemical Sciences	Biology	Earth Sciences	Atmospheric Science	Generic (Scientific databases)	Generic
Data Processing Framework	Command Processing	Service Oriented	Service Oriented	Service Oriented	Service Oriented	Script Based	Relational Database	Relational/Semi Structured Database	Relational Database
Application of Provenance	Informational; update stale, regenerate & compare data	Informational; Audit; Data Regeneration; Planning	Contextual Information; Re-enactment	Informational; Data Update	Informational; Re-enactment	Informational	Informational; Track errors	Annotation propagation; View Update	Information; update propagation
Data/Process Oriented	Data	Process	Process	Data	Process	Both	Data	Data	Data
Granularity	Spatial layers	Abstract datasets (Currently files)	Abstract resources having LSID	Files	Abstract parameters to Workflow	Files	Attributes in Database	Attributes & Tuples in Databases	Tuples in Database
Representation Scheme	Commands & Frames as Annotations	Virtual Data Language Annotations	XML/RDF Annotations	Dublin Core XML Annotations	Annotations	XML/RDF Annotations	Inverse Functions	Inverse Queries	Inverse queries
Semantic Information	No	No	Yes	Limited	No	No, Proposed	No	No	No
Storage Repository/ Backend	MetaDatabase	Virtual Data Catalog/ Relational DB	mIR repository/ Relational DB	SAM over WebDAV/ Relational DB	PreServ/ Relational DB, File System	Lineage Server/ Relational DB	Relational DB	N/A	Relational DB
Provenance Collection Overhead	Store User commands; solicit metadata	User defines derivations; automated WF trace	User defines service semantics; Automated WF Trace	Manual; Apps use DAV APIs, Users use portal	Manual; Actors use PReP API	Libraries assist user to generate, store provenance	User registers inverse functions	N/A	Inverse queries automatically generated
Addressed Scalability	No	Yes	No	No	No (Proposed)	No (Proposed)	Yes	N/A	No
Provenance Dissemination	Queries	Queries	Semantic browser; Lineage graph	Browser; Queries; GXL/RDF	Queries	Browser	Queries; box-and-arrows visualization	N/A	SQL/TriQL Queries

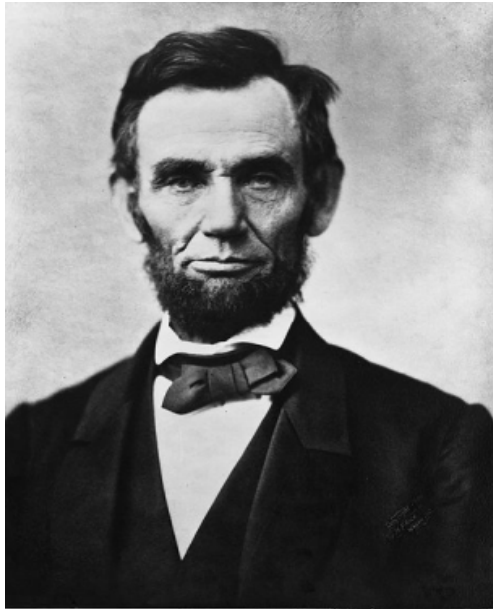
Simmhan et al., 2005

# What was the common theme of all those systems?

- They were all scientific computing systems
- And scientists trust people (more or less)
- Previous research covers provenance collection, annotation, querying, and workflow, but **security** issues are **not** handled
- For provenance in untrusted environments, we need **integrity, confidentiality** and **privacy** guarantees

So, we need **provenance of provenance**, i.e. a model for **Secure Provenance**





“History is not history unless it is  
the truth”.

Abraham Lincoln, 1856

# Secure provenance means preventing “Undetectable history rewriting”

- **Prevent undetectable modification** of history, such that,
  - Adversaries cannot insert fake events, remove genuine events from a document’s provenance
  - No one can deny history of own actions
- **Allow** fine grained preservation of privacy and confidentiality of actions
  - Users can choose which auditors can see details of their work
  - Attributes can be selectively disclosed or hidden without harming integrity check

# Usage and threat model



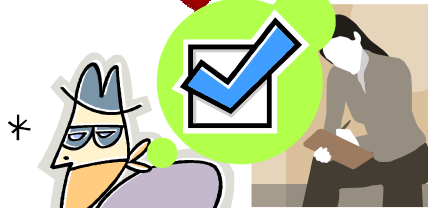
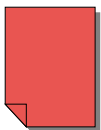
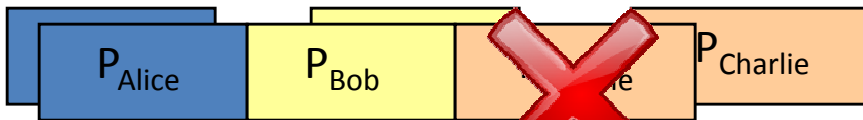
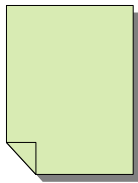
Alice



Bob



Charlie



Marvin



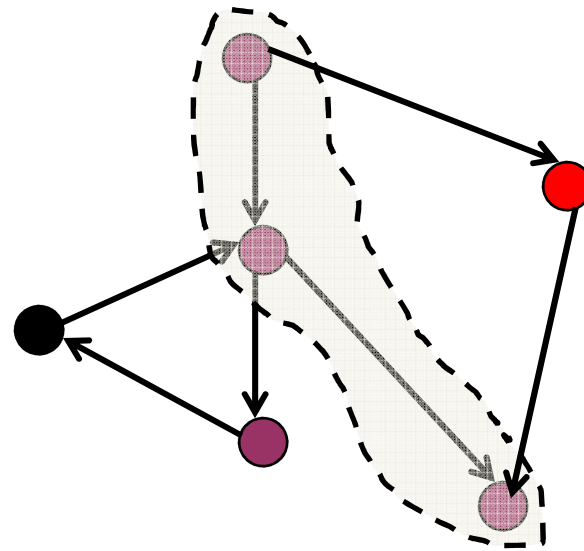
Audrey

- **Users:** Edit **documents** on their machines
- **Auditors:** semi-trusted principals
  - All auditors can verify chain
- **Adversaries:** insiders or outsiders who
  - Add or remove history entries
  - Collude with others to add/remove entries
  - Claim a chain belongs to another document
  - Repudiate an entry

Ragib Hasan, Radu Sion, and Marianne Winslett, "Introducing Secure Provenance: Problems and Challenges", ACM StorageSS 2007

# Plausible history is something that **could have happened**

**Plausible history:** The version history that will result if a document were created and subsequently edited and transferred from user to user, with provenance information **correctly** and **indelibly** recorded all along the way.



A document can have multiple plausible histories

# Our Goal: Ensure Verification of Plausible History



Given a document and a provenance chain, we want to verify if this **could have happened**

**Forger's goal:** To create a plausible history involving honest users



# Analogy: The Forger and the Prada



Fake Prada,  
street price \$50



Real Prada, price  
\$3000

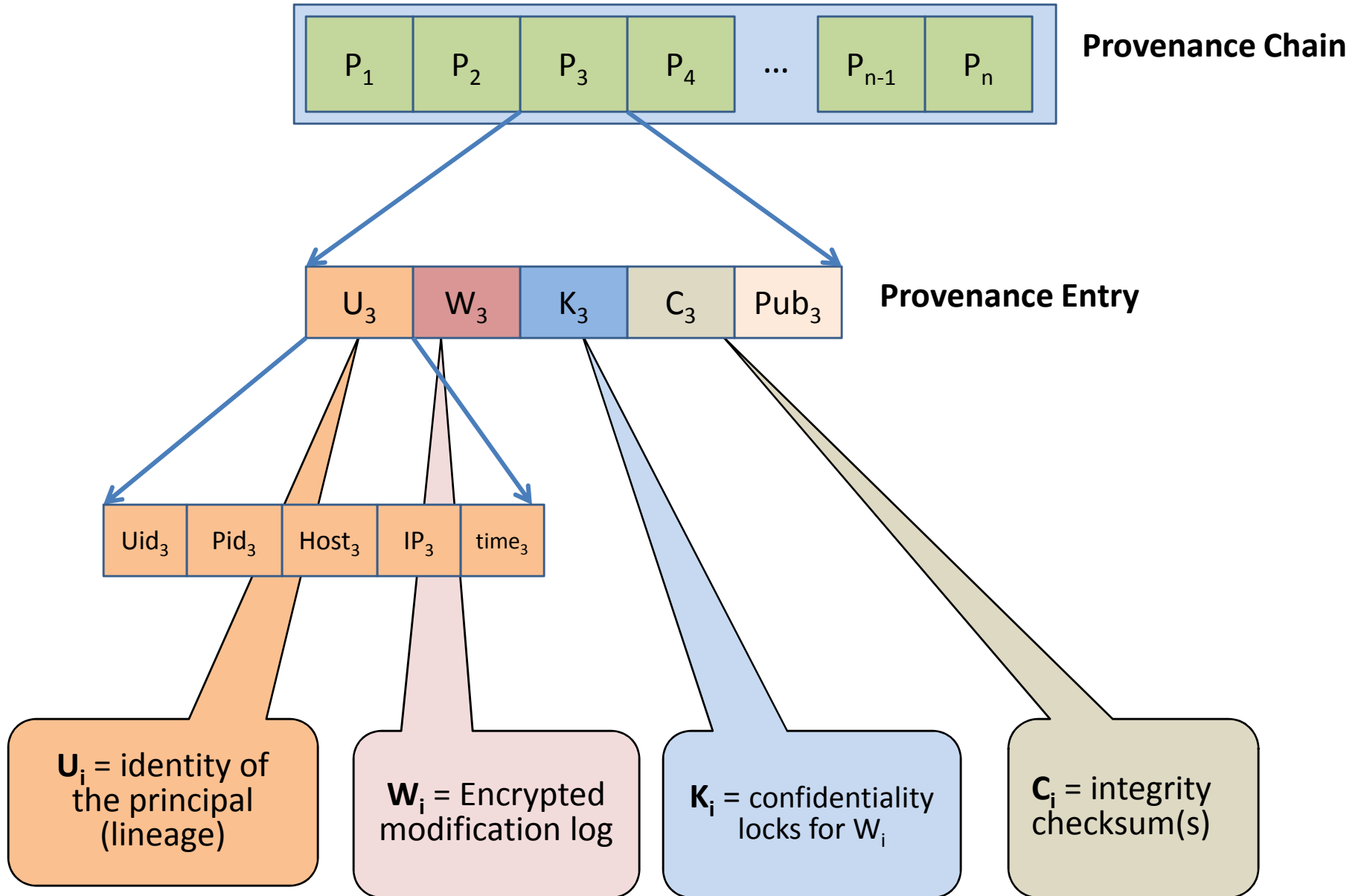
Forger's goal is to create a **plausible history** for a **fake** Prada bag (i.e. make it appear to come from real Prada distributors via real supply chain)

Forger's goal is **NOT** to replace a real Prada bag's plausible history, to show it was made elsewhere.

# Previous work on integrity assurances

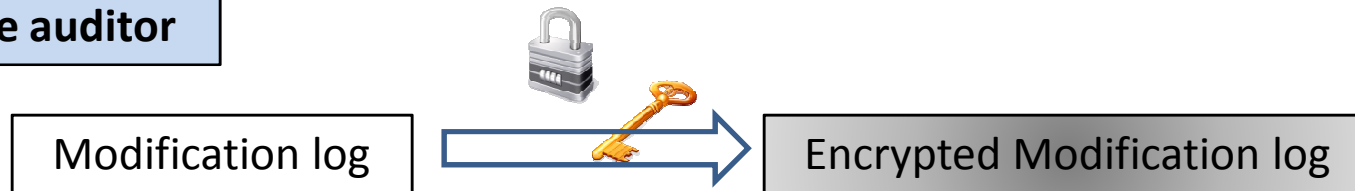
- (Logically) centralized repository (CVS, Subversion, GIT)
  - Changes to files recorded
  - Not applicable to mobile documents
- File systems with integrity assurances (SUNDR, PASIS, TCFS)
  - Provide local integrity checking
  - Do not apply to data that traverses systems
- System state entanglement (Baker 02)
  - Entangle one system's state with another, so others can serve as witness to a system's state
  - Not applicable to mobile data
- Secure audit logs / trails (Schneier and Kelsey 99), LogCrypt (Holt 2004), (Peterson et al. 2006)
  - Trusted notary certifies logs, or trusted third party given hash chain seed

# Our solution: Overview



# Our solution: Confidentiality

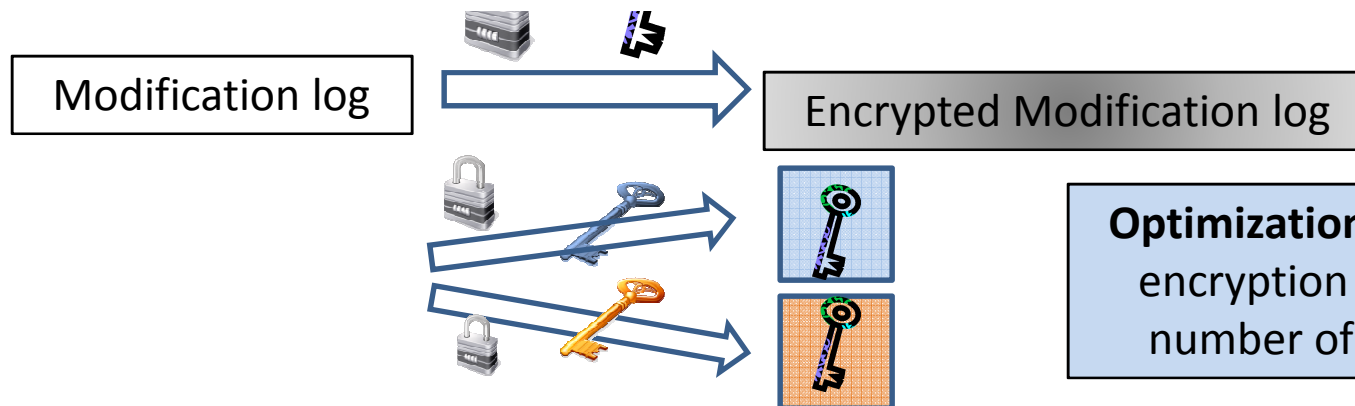
A single auditor



Multi

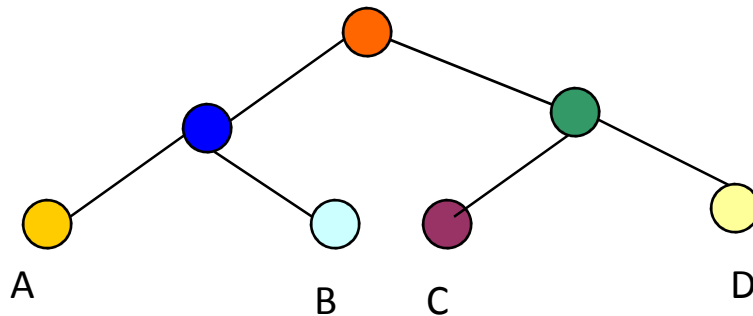
## Issues

- Each user trusts a **subset** of the auditors
- Only the auditor(s) **trusted** by the user can see the user's actions on the document



**Optimization:** Use broadcast encryption tree to reduce number of required keys

# Sidenote: Broadcast Encryption

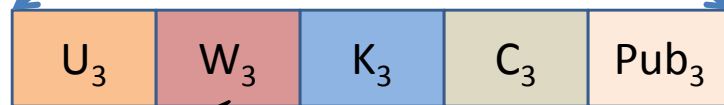
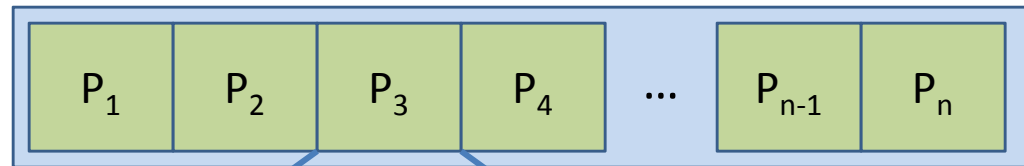


Organize the keys as a tree,  
with auditors at leaves

Each principal knows all the  
keys from leaf to root

- To trust everyone, use the root key
- To trust only D
- To trust B and C
- To trust A and C and D

# Our solution: Confidentiality



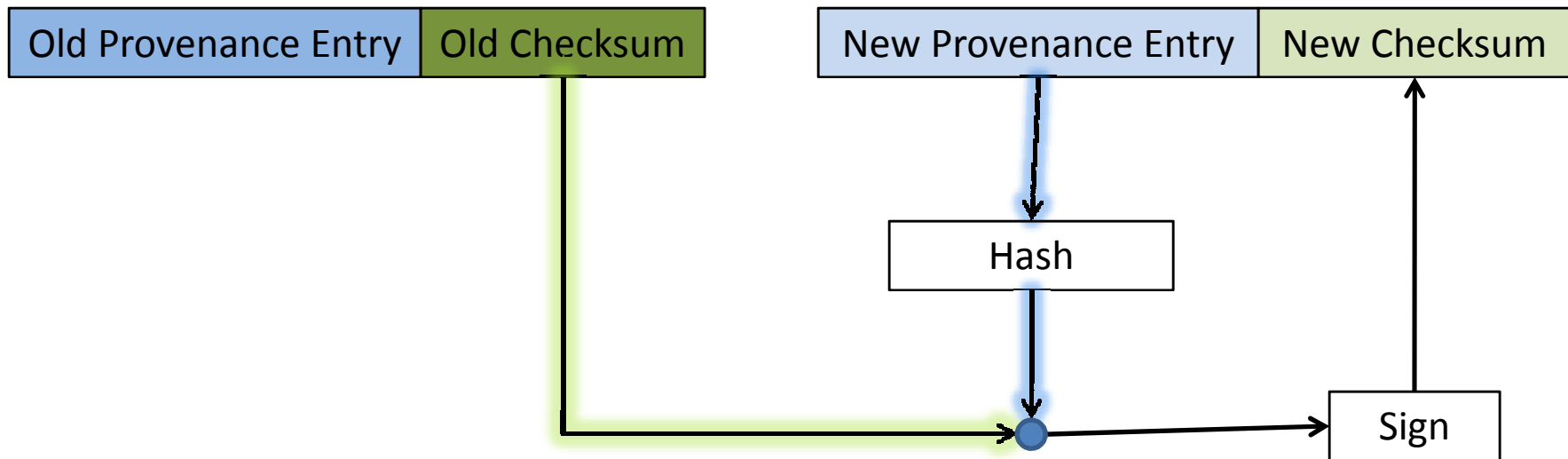
$$W_i = E_{k_i} (w_i) \parallel \text{hash}(D)$$

$$K_i = \{E_{k_a} (k_i)\}$$

- $k_i$  is a secret key that authorized auditors can retrieve from the field  $K_i$
- $w_i$  is either the diff or the set of actions taken on the file

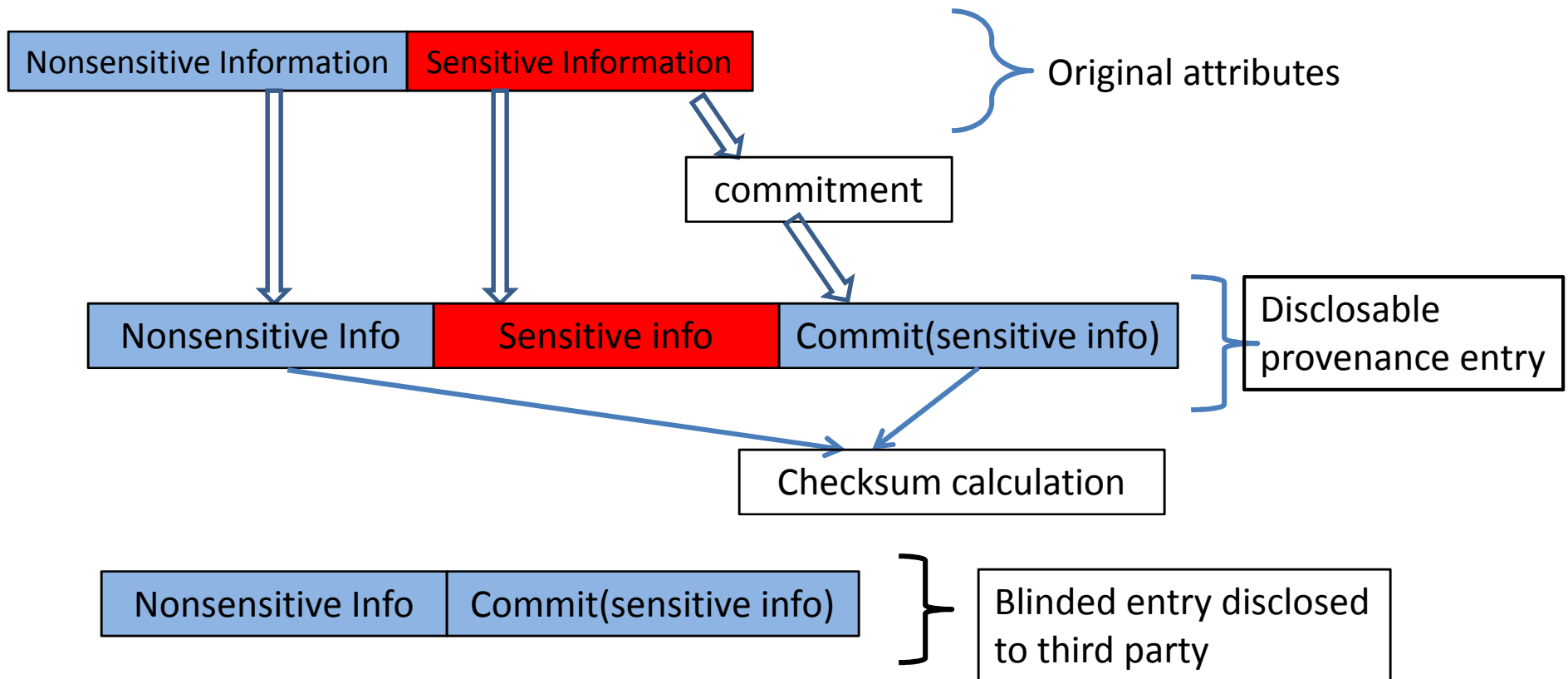
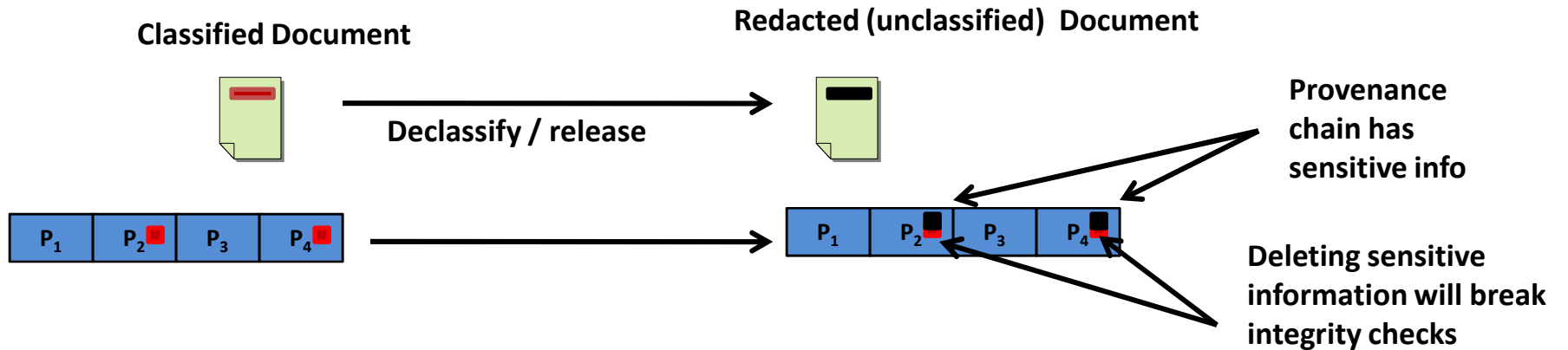
- $k_a$  is the key of a trusted auditor

# Our solution: Integrity

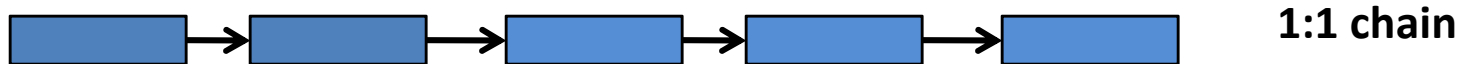


$$C_i = S_{\text{private}_i}(\text{hash}(U_i, W_i, K_i) | C_{i-1})$$

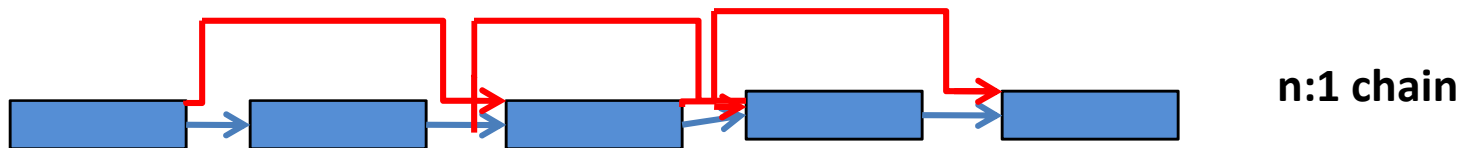
# Fine grained control over confidentiality



# We can summarize provenance chains to save space, make audits fast



Each entry has **1** checksum, calculated from **1** previous checksum



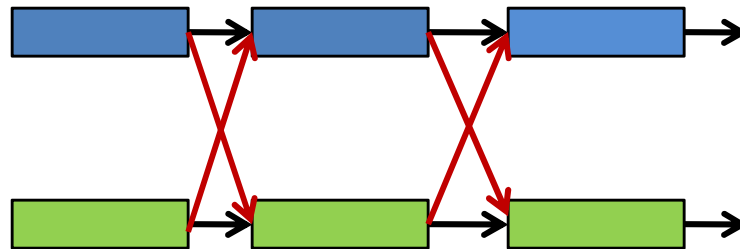
Each entry has **n** checksums, each of them calculated from **1** previous checksum

We can systematically remove entries from the chain while still being able to prove integrity of chain

# Co-Provenance provides stronger guarantees

**Problem:** We can only tell if a given provenance chain shows what could have happened, but not exactly what really happened

**Solution:** Entangle chains of **related** documents



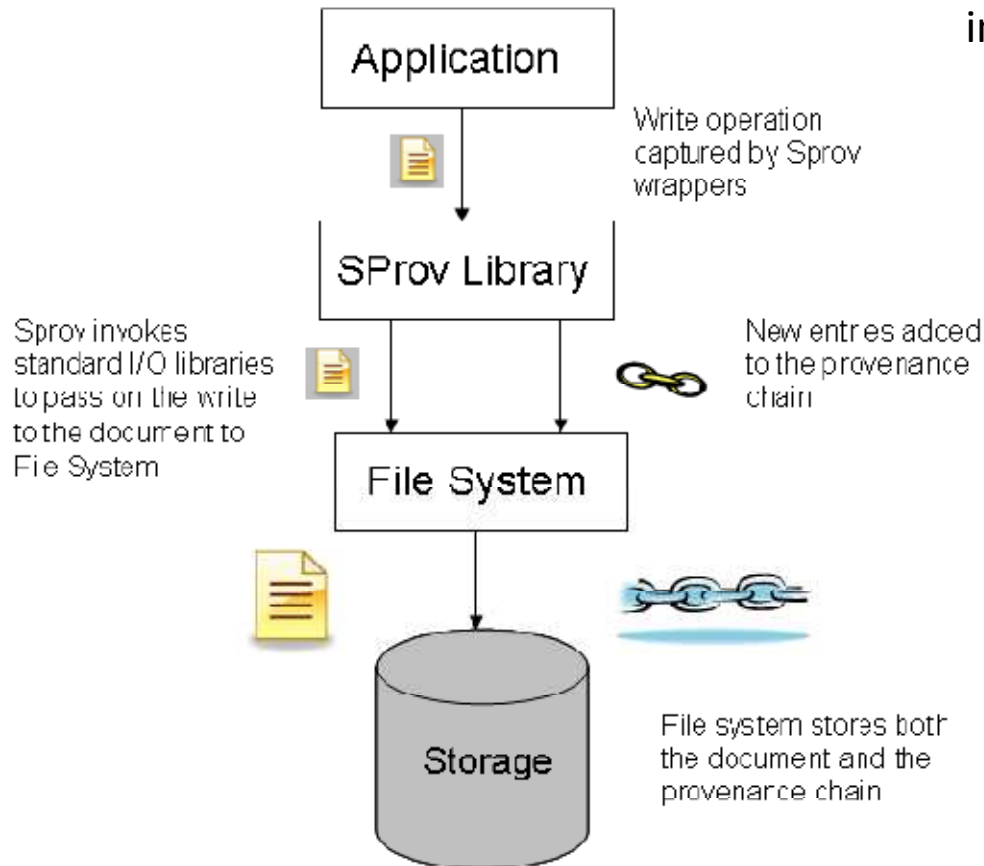
Each entry now contains **n checksums**, each computed using the checksum of the last entry of the provenance chain of a related document

# Implementation Options

- Secure provenance management can be implemented in
  - **Kernel layer**
    - Transparent to user apps
    - But less portable
  - **File System layer**
    - Also transparent to user apps
    - Less portable
  - **Application layer**
    - Needs (slight) modification of apps
    - Highly portable

# Implementation: SPROV library

- Automated capture of writes to files
- Record all write data, and related context information in the provenance entry



SPROV is an application layer library in C

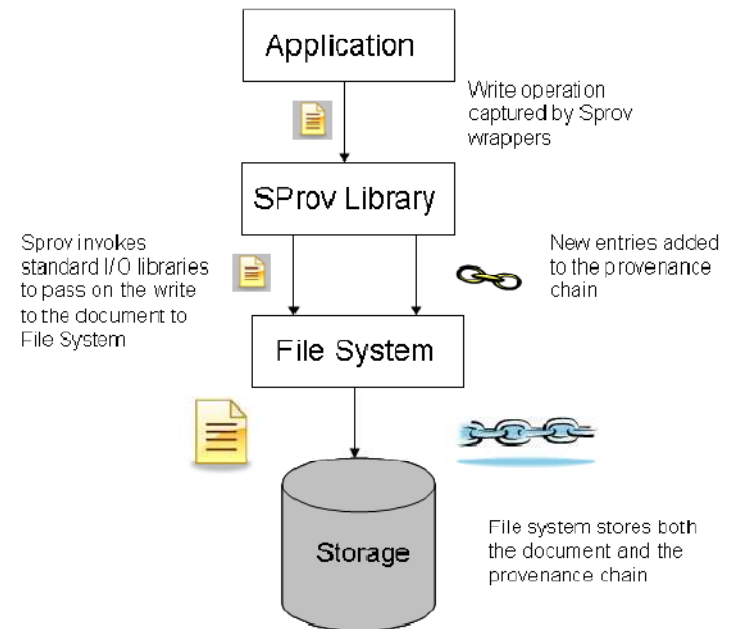
Can be added to existing applications with almost no change to code

Provides the file system APIs from `stdio.h`

To add secure provenance, simply relink applications with `sprov` library instead of `stdio.h`

# Implementation: How Sprov works

- Modified **fopen** / **fwrite** / **fclose** calls
  - File I/O calls trigger provenance chain handling mechanism
  - Calls to fwrite log the modifications
  - Calls to fclose writes a new provenance entry, computes new checksum



# Experiment goals

- Measure run time overhead of secure provenance
  - Run the benchmarks with and without secure provenance
  - Calculate the % overhead
- Observe the effects of different types of workloads

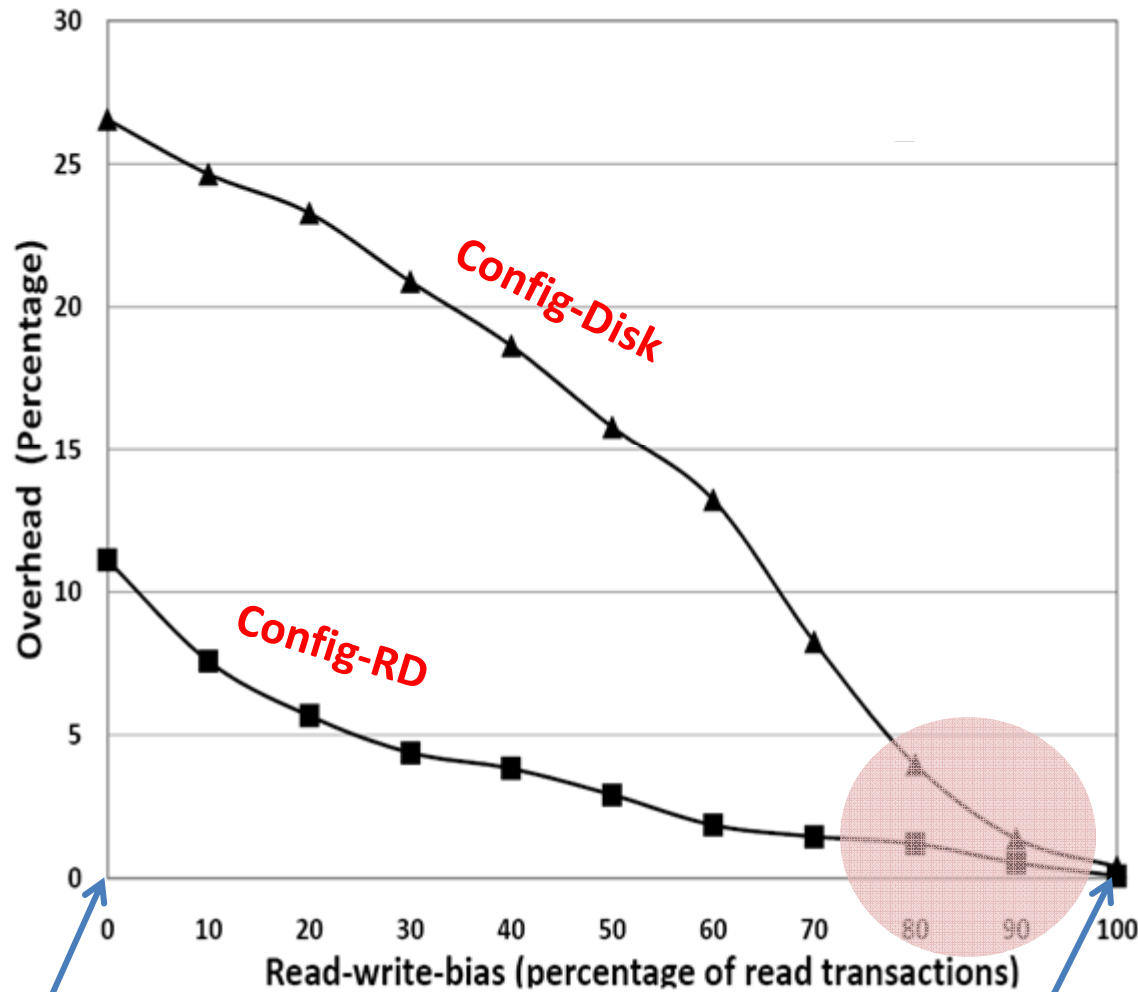
# Experimental settings

- **Crypto settings**
  - 1024 bit DSA signatures
  - 128 bit AES encryption
  - SHA-1 for hashes
- **Experiments**
  - **Postmark benchmark** : performance of writes on small files
  - **Hybrid workload benchmark** : performance with real life file system distribution and workloads

# Experimental settings

- **Chain storage:**
  - Chains stored as XML formatted meta-files
- **Modes**
  - **Config-Disk** : Provenance chains stored on Disk
  - **Config-RD**: Provenance chains stored on RAM Disk buffer, and periodically saved to disk

# Postmark small file benchmark: Overhead < 5% for realistic workloads



100% writes,  
0% reads

0% writes,  
100% reads

- **20,000** small files (8KB-64KB) subjected to 100% to 0% write load with the Postmark benchmark
- At 100% write load, execution time overhead of using secure provenance over the no-provenance case is approx. 27% (12% with RD)
- At 50% write load, overheads go down to 16% (3% with RD)
- Overheads are less than **5%** with 20% or less write load

# Hybrid workloads: Simulating real file systems

## File system distribution:

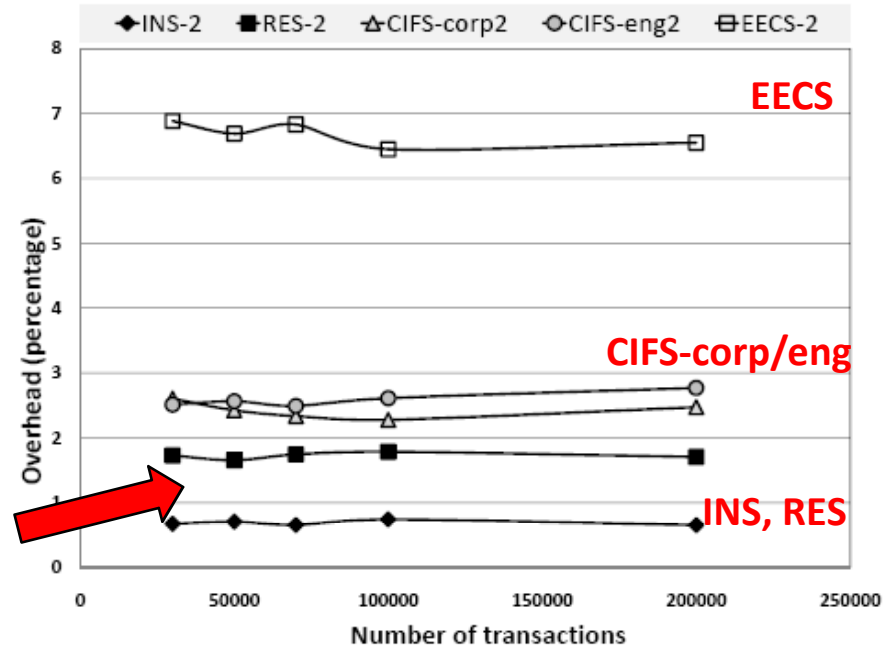
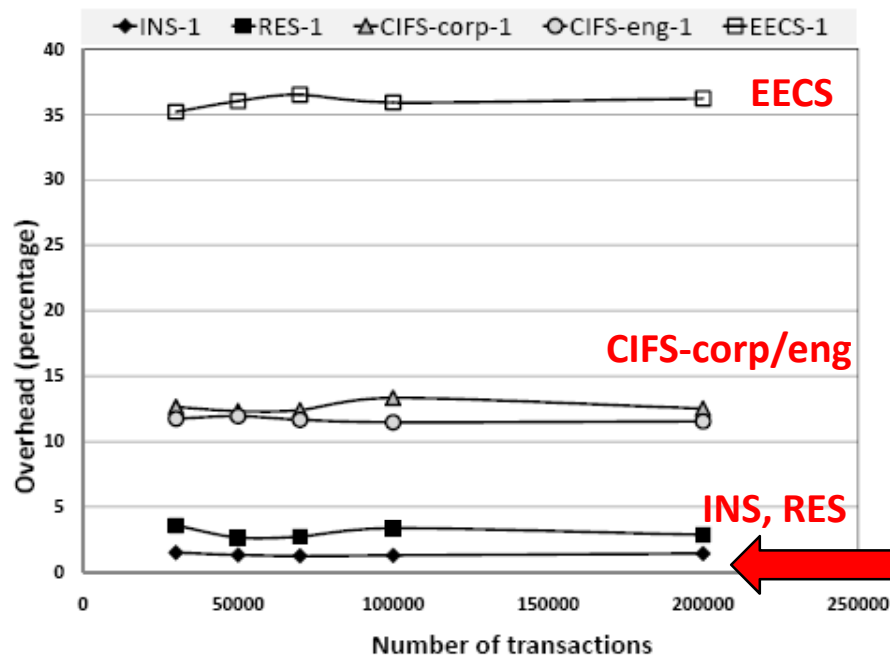
- File size distribution in real file systems follows the **log normal distribution** [Bolosky and Douceur 99]
- We created a file system with **20,000** files, using the lognormal parameters  $\mu = 8.46$ ,  $\sigma = 2.4$
- Median file size = 4KB , mean file size = 80KB
- In addition, we included a few large (1GB+) files

# Hybrid workloads: Using Data from real systems

## Workload

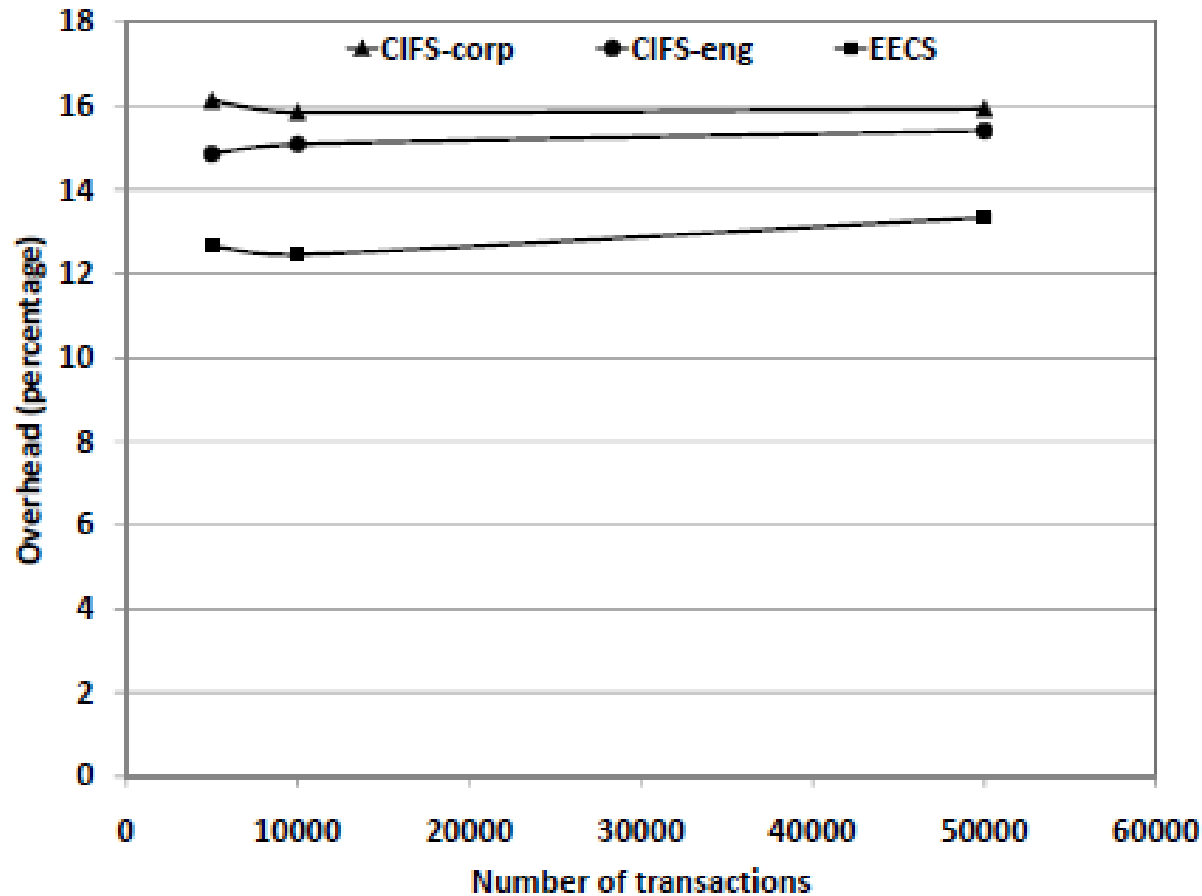
- **INS**: Instructional lab (**1.1%** writes) [Roselli 00]
- **RES**: A Research lab (**2.9%** writes) [Roselli 00]
- **CIFS-Corp**: (**15%** writes) [Leung 08]
- **CIFS-Eng**: (**17%** writes) [Leung 08]
- **EECS**: (**82%** writes) [Ellard 03]

# Typical real life workloads: 1 - 13% overhead



- **INS** and **RES** are read-intensive (80%+ reads), so overheads are very low in both cases.
- **CIFS-corp** and **CIFS-eng** have 2:1 ratio of reads and writes, overheads are still low (range from 12% to 2.5%)
- **EECS** has very high write load (82%+), so the overhead is higher, but still less than 35% for Config-Disk, and less than 7% for Config-RD

# Recording both read and writes costs 12-16% for real-life workloads



# Application-specific history requires different techniques

- **History integrity in databases**
  - Developed a solution that leverages WORM storage to provide history integrity for database tuples
  - Incurs only 1%-3% overhead for TPC-C run-time
  - Reduces window of vulnerability 5-fold
  - Reduced audit times 100-fold over the previous state-of-the-art solution
- **History integrity in cloud computing**

# Beyond Provenance

- Provenance, versions, and snapshots are general cases of the property – **remembrance**
- By enabling different objects to recall their past, we can introduce richer functionality, for example:
  - **1.5<sup>th</sup> factor authentication**: Use “**What you remember**” to strengthen any other authentication factor
  - **Flexible security policies**: Use knowledge of past of attributes/variables when evaluating security policies

Ragib Hasan, Radu Sion, and Marianne Winslett, “**Remembrance: The Unbearable Sentience of Being Digital**”, CIDR 2009

# Papers

- Ragib Hasan, Radu Sion, and Marianne Winslett, “Introducing Secure Provenance: Problems and Challenges”, ACM StorageSS 2007
- Ragib Hasan, Radu Sion, and Marianne Winslett, “**The Case of the Fake Picasso: Preventing History Forgery with Secure Provenance**”, USENIX FAST 2009, ACM TOS (under review)
- Ragib Hasan, Radu Sion, and Marianne Winslett, “Remembrance: The Unbearable Sentience of Being Digital”, CIDR 2009

**Summary: Secure provenance** possible at **low cost**

**Yes, We CAN** achieve secure provenance with integrity and confidentiality assurances with reasonable overheads

- For most real-life workloads, overheads are between 1% and 15% only



More info at <http://tinyurl.com/secprov>