Public append-only logs

Linus Nordberg, NORDUnet
linus@nordu.net 0x23291265

FRISC winter school, Finse, Norway

Introduction

Detection

- Protection is hard
- . . .let's detect more
- Both for protecting the next
- . . .and for deterrence

Detection

- Protection is hard
- . . .let's detect more
- Both for protecting the next
- . . .and for deterrence

Detection

- Protection is hard
- . . .let's detect more
- Both for protecting the next
- . . .and for deterrence

Detection

- Protection is hard
- . . .let's detect more
- Both for protecting the next
- . . .and for deterrence

An introduction to public, append-only, untrusted logs.

A log is a database with log entries.

Public

- Anybody can retrieve a log entry
- In some systems anybody can submit log entries
- . . .in which case spam countermeasures is a good idea

Public

- Anybody can retrieve a log entry
- In some systems anybody can submit log entries
- . . .in which case spam countermeasures is a good idea

Public

- Anybody can retrieve a log entry
- In some systems anybody can submit log entries
- . . .in which case spam countermeasures is a good idea

Append-only

- Entries can be added but never changed or removed
- A pretty specialised database
- Poses some operational challenges

Append-only

- Entries can be added but never changed or removed
- A pretty specialised database
- Poses some operational challenges

Append-only

- Entries can be added but never changed or removed
- A pretty specialised database
- Poses some operational challenges

Untrusted

- Important, to avoid “just another key”
- A lying or compromised log will get caught

Untrusted

- Important, to avoid “just another key”
- A lying or compromised log will get caught

A solution

- Merkle tree
- Signed tree heads
- The need for a quick proof of inclusion – signed timestamps
- Auditing is crucial

A solution

- Merkle tree
- Signed tree heads
- The need for a quick proof of inclusion – signed timestamps
- Auditing is crucial

A solution

- Merkle tree
- Signed tree heads
- The need for a quick proof of inclusion – signed timestamps
- Auditing is crucial

A solution

- Merkle tree
- Signed tree heads
- The need for a quick proof of inclusion – signed timestamps
- Auditing is crucial

What append-only logs can be used for.

X509 certificates, a.k.a. CT

- Certificate Transparency, RFC 6962
- Anybody can (and CA:s should) submit certs to logs
- Browsers require certs to be logged
- Monitors keep logs honest
- Signed Certificate Timestamps for quicker “proof”
- Enabling site owners, who are the ones who really know about issuance

X509 certificates, a.k.a. CT

- Certificate Transparency, RFC 6962
- Anybody can (and CA:s should) submit certs to logs
- Browsers require certs to be logged
- Monitors keep logs honest
- Signed Certificate Timestamps for quicker “proof”
- Enabling site owners, who are the ones who really know about issuance

X509 certificates, a.k.a. CT

- Certificate Transparency, RFC 6962
- Anybody can (and CA:s should) submit certs to logs
- Browsers require certs to be logged
- Monitors keep logs honest
- Signed Certificate Timestamps for quicker “proof”
- Enabling site owners, who are the ones who really know about issuance

X509 certificates, a.k.a. CT

- Certificate Transparency, RFC 6962
- Anybody can (and CA:s should) submit certs to logs
- Browsers require certs to be logged
- Monitors keep logs honest
- Signed Certificate Timestamps for quicker “proof”
- Enabling site owners, who are the ones who really know about issuance

X509 certificates, a.k.a. CT

- Certificate Transparency, RFC 6962
- Anybody can (and CA:s should) submit certs to logs
- Browsers require certs to be logged
- Monitors keep logs honest
- Signed Certificate Timestamps for quicker “proof”
- Enabling site owners, who are the ones who really know about issuance

X509 certificates, a.k.a. CT

- Certificate Transparency, RFC 6962
- Anybody can (and CA:s should) submit certs to logs
- Browsers require certs to be logged
- Monitors keep logs honest
- Signed Certificate Timestamps for quicker “proof”
- Enabling site owners, who are the ones who really know about issuance

Binary Transparency

- Am I running the same code as everybody else?
- Any binary, including typically closed firmware
- . . .but also code like JavaScript
- Goes especially well with reproducible builds

Binary Transparency

- Am I running the same code as everybody else?
- Any binary, including typically closed firmware
- . . .but also code like JavaScript
- Goes especially well with reproducible builds

Binary Transparency

- Am I running the same code as everybody else?
- Any binary, including typically closed firmware
- . . .but also code like JavaScript
- Goes especially well with reproducible builds

Binary Transparency

- Am I running the same code as everybody else?
- Any binary, including typically closed firmware
- . . .but also code like JavaScript
- Goes especially well with reproducible builds

DNSSEC Transparency

- Lots of keys in DNSSEC
- Especially the DS records should be watched, to detect misissuance by parent zone
- See draft-zhang-trans-ct-dnssec

DNSSEC Transparency

- Lots of keys in DNSSEC
- Especially the DS records should be watched, to detect misissuance by parent zone
- See draft-zhang-trans-ct-dnssec

DNSSEC Transparency

- Lots of keys in DNSSEC
- Especially the DS records should be watched, to detect misissuance by parent zone
- See draft-zhang-trans-ct-dnssec

Tor Consensus Transparency

- Tor consensus documents are trusted if they're signed by five keys
- Logging consensus documents would increase the chances of detecting an attack against a subset of the users
- Directory caches could act as log auditors with help from clients
- See "Tor Consensus Transparency proposal"

Tor Consensus Transparency

- Tor consensus documents are trusted if they're signed by five keys
- Logging consensus documents would increase the chances of detecting an attack against a subset of the users
- Directory caches could act as log auditors with help from clients
- See "Tor Consensus Transparency proposal"

Tor Consensus Transparency

- Tor consensus documents are trusted if they're signed by five keys
- Logging consensus documents would increase the chances of detecting an attack against a subset of the users
- Directory caches could act as log auditors with help from clients
- See "Tor Consensus Transparency proposal"

Tor Consensus Transparency

- Tor consensus documents are trusted if they're signed by five keys
- Logging consensus documents would increase the chances of detecting an attack against a subset of the users
- Directory caches could act as log auditors with help from clients
- See "Tor Consensus Transparency proposal"

System logs

- Also known as syslog
- Probably not public
- . . .but append-only and untrusted
- Useful for forensics
- See Crosby and Wallach 2009

System logs

- Also known as syslog
- Probably not public
- . . .but append-only and untrusted
- Useful for forensics
- See Crosby and Wallach 2009

System logs

- Also known as syslog
- Probably not public
- . . .but append-only and untrusted
- Useful for forensics
- See Crosby and Wallach 2009

System logs

- Also known as syslog
- Probably not public
- . . .but append-only and untrusted
- Useful for forensics
- See Crosby and Wallach 2009

System logs

- Also known as syslog
- Probably not public
- . . .but append-only and untrusted
- Useful for forensics
- See Crosby and Wallach 2009

Notary services

- Diplomas from online courses
- Timestamp service
- Tax office has seen a receipt
- Tracking legal documents like consent receipts

Notary services

- Diplomas from online courses
- Timestamp service
- Tax office has seen a receipt
- Tracking legal documents like consent receipts

Notary services

- Diplomas from online courses
- Timestamp service
- Tax office has seen a receipt
- Tracking legal documents like consent receipts

Notary services

- Diplomas from online courses
- Timestamp service
- Tax office has seen a receipt
- Tracking legal documents like consent receipts

Monitoring, introduction

- Log auditing ensures that the log behaves correctly, i.e.
  - does include submitted entries in the log, on time
  - doesn't change or remove any entries
- Log monitoring cares also about the contents of the log

Monitoring, introduction

- Log auditing ensures that the log behaves correctly, i.e.
  - does include submitted entries in the log, on time
  - doesn't change or remove any entries
- Log monitoring cares also about the contents of the log

Monitoring, introduction

- Log auditing ensures that the log behaves correctly, i.e.
  - does include submitted entries in the log, on time
  - doesn't change or remove any entries
- Log monitoring cares also about the contents of the log

Monitoring, introduction

- Log auditing ensures that the log behaves correctly, i.e.
  - does include submitted entries in the log, on time
  - doesn't change or remove any entries
- Log monitoring cares also about the contents of the log

Temporal

Verifying log consistency over time

- An inclusion proof shows that a given entry is indeed part of a given tree
- Auditors send a log index I and a signed tree head STH and
- . . .receive the nodes needed to calculate STH given entry number I

Temporal

Verifying log consistency over time

- An inclusion proof shows that a given entry is indeed part of a given tree
- Auditors send a log index I and a signed tree head STH and
- . . .receive the nodes needed to calculate STH given entry number I

Temporal

Verifying log consistency over time

- An inclusion proof shows that a given entry is indeed part of a given tree
- Auditors send a log index I and a signed tree head STH and
- . . .receive the nodes needed to calculate STH given entry number I

Temporal (cont.)

- A consistency proof shows that a given signed tree head STH_1 is a subset of another given signed tree head STH_2
- Auditors send two tree sizes, I_1 and I_2, representing STH_1 and STH_2 and
- . . .receive the nodes required to verify that the first I_1 entries are equal in both trees
- Consistency proofs also makes it possible to discard old tree heads once they've been verified

Temporal (cont.)

- A consistency proof shows that a given signed tree head STH_1 is a subset of another given signed tree head STH_2
- Auditors send two tree sizes, l_1 and l_2, representing STH_1 and STH_2 and
- . . .receive the nodes required to verify that the first l_1 entries are equal in both trees
- Consistency proofs also makes it possible to discard old tree heads once they've been verified

Temporal (cont.)

- A consistency proof shows that a given signed tree head STH_1 is a subset of another given signed tree head STH_2
- Auditors send two tree sizes, I_1 and I_2, representing STH_1 and STH_2 and
- . . .receive the nodes required to verify that the first I_1 entries are equal in both trees
- Consistency proofs also makes it possible to discard old tree heads once they've been verified

Temporal (cont.)

- A consistency proof shows that a given signed tree head STH_1 is a subset of another given signed tree head STH_2
- Auditors send two tree sizes, l_1 and l_2, representing STH_1 and STH_2 and
- . . .receive the nodes required to verify that the first l_1 entries are equal in both trees
- Consistency proofs also makes it possible to discard old tree heads once they've been verified

Spatial

Verifying log consistency in space, i.e. the same log being shown to all parties

- A log could fork the tree and serve certain clients a different view
- This is detected by clients gossiping about their view of the log
- Not specified how this should be done yet, but see draft-linus-trans-gossip-ct for a suggestion
- Except it lacks gossiping of STH's

Spatial

Verifying log consistency in space, i.e. the same log being shown to all parties

- A log could fork the tree and serve certain clients a different view
- This is detected by clients gossiping about their view of the log
- Not specified how this should be done yet, but see draft-linus-trans-gossip-ct for a suggestion
- Except it lacks gossiping of STH's

Spatial

Verifying log consistency in space, i.e. the same log being shown to all parties

- A log could fork the tree and serve certain clients a different view
- This is detected by clients gossiping about their view of the log
- Not specified how this should be done yet, but see draft-linus-trans-gossip-ct for a suggestion
- Except it lacks gossiping of STH's

Spatial

Verifying log consistency in space, i.e. the same log being shown to all parties

- A log could fork the tree and serve certain clients a different view
- This is detected by clients gossiping about their view of the log
- Not specified how this should be done yet, but see draft-linus-trans-gossip-ct for a suggestion
- Except it lacks gossiping of STH's

catlfish design

- NORDUnet is developing a CT log as part of a GÉANT project
- Free software, written in Erlang
- Modular – build other transparency systems
- Distributed – scalability and participation across organisational boundaries
- Expecting moderate write load (submit, 0.1 qps) and very high read load (queries, 7-20k qps)
- HSM support

catlfish design

- NORDUnet is developing a CT log as part of a GÉANT project
- Free software, written in Erlang
- Modular – build other transparency systems
- Distributed – scalability and participation across organisational boundaries
- Expecting moderate write load (submit, 0.1 qps) and very high read load (queries, 7-20k qps)
- HSM support

catlfish design

- NORDUnet is developing a CT log as part of a GÉANT project
- Free software, written in Erlang
- Modular – build other transparency systems
- Distributed – scalability and participation across organisational boundaries
- Expecting moderate write load (submit, 0.1 qps) and very high read load (queries, 7-20k qps)
- HSM support

catlfish design

- NORDUnet is developing a CT log as part of a GÉANT project
- Free software, written in Erlang
- Modular – build other transparency systems
- Distributed – scalability and participation across organisational boundaries
- Expecting moderate write load (submit, 0.1 qps) and very high read load (queries, 7-20k qps)
- HSM support

catlfish design

- NORDUnet is developing a CT log as part of a GÉANT project
- Free software, written in Erlang
- Modular – build other transparency systems
- Distributed – scalability and participation across organisational boundaries
- Expecting moderate write load (submit, 0.1 qps) and very high read load (queries, 7-20k qps)
- HSM support

catlfish design

- NORDUnet is developing a CT log as part of a GÉANT project
- Free software, written in Erlang
- Modular – build other transparency systems
- Distributed – scalability and participation across organisational boundaries
- Expecting moderate write load (submit, 0.1 qps) and very high read load (queries, 7-20k qps)
- HSM support

Sources and credit

- RFC 6962, IETF
- Tor Consensus Transparency proposal

  https://gitweb.torproject.org/user/linus/torspec.git/tree/proposals/ideas/xxx-tor-consensus-transparency.txt?h=tct
- Crosby, S. and D. Wallach, "Efficient Data Structures for Tamper-Evident Logging", Proceedings of the 18th USENIX Security Symposium, Montreal, August 2009

  http://static.usenix.org/event/sec09/tech/full_papers/crosby.pdf

Questions and discussion