Towards End-to-End Encrypted Calendars

Fernando Virdia
University of Surrey

Joint work with Tomás Bertoli (UBA), Benjamin Dowling (KCL) and Simone Colombo (KCL)

Cryptographic Applications Workshop 2026

Hot takes 🌶️🔥

Hot takes 🌶️🔥

- Cloud service providers want to collect the largest possible amount of information from their users in order to monetise them

Hot takes 🌶️🔥

- Cloud service providers want to collect the largest possible amount of information from their users in order to monetise them
- Data leaks have proven inevitable, resulting in individuals and companies being regularly victimised online

Hot takes 🌶️🔥

- Cloud service providers want to collect the largest possible amount of information from their users in order to monetise them
- Data leaks have proven inevitable, resulting in individuals and companies being regularly victimised online
- Remote data storage is just too convenient

Hot takes 🌶️🔥

- Cloud service providers want to collect the largest possible amount of information from their users in order to monetise them
- Data leaks have proven inevitable, resulting in individuals and companies being regularly victimised online
- Remote data storage is just too convenient

- A mitigation? End-to-end encryption (E2EE)

Hot takes 🌶️🔥

- Cloud service providers want to collect the largest possible amount of information from their users in order to monetise them
- Data leaks have proven inevitable, resulting in individuals and companies being regularly victimised online
- Remote data storage is just too convenient

- A mitigation? End-to-end encryption (E2EE)
- We've seen successful deployments of secure in-transit encryption and E2EE messaging

Hot takes 🌶️🔥

- Cloud service providers want to collect the largest possible amount of information from their users in order to monetise them
- Data leaks have proven inevitable, resulting in individuals and companies being regularly victimised online
- Remote data storage is just too convenient

- A mitigation? End-to-end encryption (E2EE)
- We've seen successful deployments of secure in-transit encryption and E2EE messaging
- E2EE storage has been a mixed bag, only recently a provably secure proposal, CSS

Hot takes 🌶️🔥

- Cloud service providers want to collect the largest possible amount of information from their users in order to monetise them
- Data leaks have proven inevitable, resulting in individuals and companies being regularly victimised online
- Remote data storage is just too convenient

- A mitigation? End-to-end encryption (E2EE)
- We've seen successful deployments of secure in-transit encryption and E2EE messaging
- E2EE storage has been a mixed bag, only recently a provably secure proposal, CSS
- Natural question: Is CSS cryptographically usable? Can it improve my daily life?

Hot takes 🌶️🔥

- Cloud service providers want to collect the largest possible amount of information from their users in order to monetise them
- Data leaks have proven inevitable, resulting in individuals and companies being regularly victimised online
- Remote data storage is just too convenient

- A mitigation? End-to-end encryption (E2EE)
- We've seen successful deployments of secure in-transit encryption and E2EE messaging
- E2EE storage has been a mixed bag, only recently a provably secure proposal, CSS
- Natural question: Is CSS cryptographically usable? Can it improve my daily life?
- My favourite activity: ~~doomscrolling~~ checking my work calendar

CalDAV

- Most online calendars support CalDAV (RFC 4791, 6638)

CalDAV

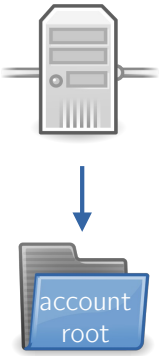
- Most online calendars support CalDAV (RFC 4791, 6638)
- An extension of WebDAV, a storage-over-HTTP protocol (RFC 4918)

CalDAV

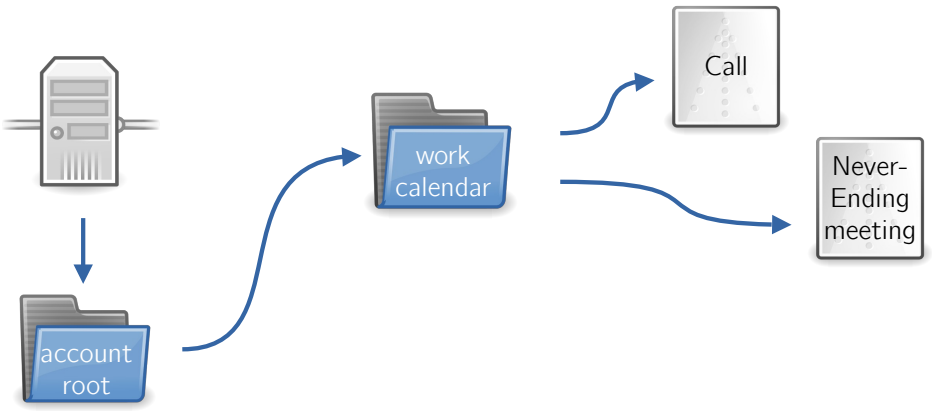
- Most online calendars support CalDAV (RFC 4791, 6638)
- An extension of WebDAV, a storage-over-HTTP protocol (RFC 4918)
- Rich functionality:
  - Multiple calendars per account
  - Text search, range queries, free/busy view of other users, RSVP

CalDAV overview

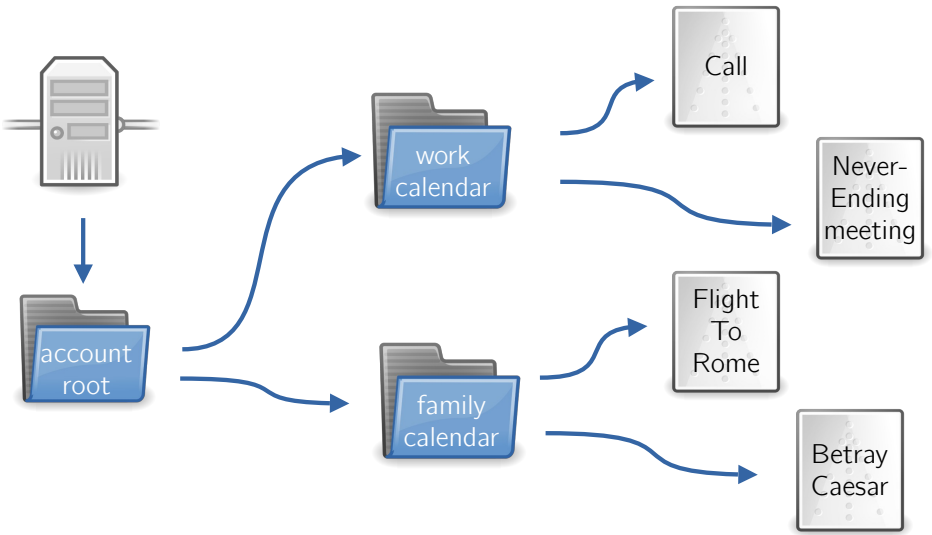
CalDAV overview



CalDAV overview



CalDAV overview



Encrypted calendars in the wild

- Some providers offer(ed) them: Proton, Tuta, Skiff, OurCal

Encrypted calendars in the wild

- Some providers offer(ed) them: Proton, Tuta, Skiff, OurCal
- In 2024/25, Tomás had a closer look at Tuta:
  - Introduced *circa* 2019, open source client
  - Essentially the same key management as CSS

Encrypted calendars in the wild

- Some providers offer(ed) them: Proton, Tuta, Skiff, OurCal
- In 2024/25, Tomás had a closer look at Tuta:
  - Introduced *circa* 2019, open source client
  - Essentially the same key management as CSS

Tuta takeaways

Mostly solid, with expected "in the wild" caveats:

Encrypted calendars in the wild

- Some providers offer(ed) them: Proton, Tuta, Skiff, OurCal
- In 2024/25, Tomás had a closer look at Tuta:
  - Introduced *circa* 2019, open source client
  - Essentially the same key management as CSS

Tuta takeaways

Mostly solid, with expected "in the wild" caveats:

- Claim malicious security, but user public keys are served by the server

Encrypted calendars in the wild

- Some providers offer(ed) them: Proton, Tuta, Skiff, OurCal
- In 2024/25, Tomás had a closer look at Tuta:
  - Introduced *circa* 2019, open source client
  - Essentially the same key management as CSS

Tuta takeaways

Mostly solid, with expected "in the wild" caveats:

- Claim malicious security, but user public keys are served by the server
- Files are JSON objects, values are CBC-encrypted and HMAC only over value, not full object
  $\implies$ Can switch fields around and pass authentication

Encrypted calendars in the wild

- Some providers offer(ed) them: Proton, Tuta, Skiff, OurCal
- In 2024/25, Tomás had a closer look at Tuta:
  - Introduced *circa* 2019, open source client
  - Essentially the same key management as CSS

Tuta takeaways

Mostly solid, with expected "in the wild" caveats:

- Claim malicious security, but user public keys are served by the server
- Files are JSON objects, values are CBC-encrypted and HMAC only over value, not full object
  $\implies$ Can switch fields around and pass authentication
  - WONTFIX, github.com/tutao/tutanota/issues/1470
  - In practice, at most, events disappear, a trivial denial-of-service

Our goals

- Design an E2EE calendar on top of CSS

Our goals

- Design an E2EE calendar on top of CSS
- Figure out multi-user and search functionality, underspecified in CSS

Our goals

- Design an E2EE calendar on top of CSS
- Figure out multi-user and search functionality, underspecified in CSS
- Achieve CalDAV client compatibility (Apple Calendar, Thunderbird, ...) by having a local proxy translate CalDAV requests into CSS queries

Our goals

- Design an E2EE calendar on top of CSS
- Figure out multi-user and search functionality, underspecified in CSS
- Achieve CalDAV client compatibility (Apple Calendar, Thunderbird, ...) by having a local proxy translate CalDAV requests into CSS queries

Where we are

- Identified some blockers in original CSS design

Our goals

- Design an E2EE calendar on top of CSS
- Figure out multi-user and search functionality, underspecified in CSS
- Achieve CalDAV client compatibility (Apple Calendar, Thunderbird, ...) by having a local proxy translate CalDAV requests into CSS queries

Where we are

- Identified some blockers in original CSS design
- Extended CSS security model to account for solutions to blockers

Our goals

- Design an E2EE calendar on top of CSS
- Figure out multi-user and search functionality, underspecified in CSS
- Achieve CalDAV client compatibility (Apple Calendar, Thunderbird, ...) by having a local proxy translate CalDAV requests into CSS queries

Where we are

- Identified some blockers in original CSS design
- Extended CSS security model to account for solutions to blockers
- Outlined strategies for search, attendance, and invitations, without relying on heavy machinery

Our goals

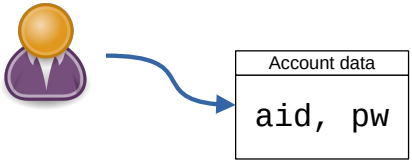
- Design an E2EE calendar on top of CSS
- Figure out multi-user and search functionality, underspecified in CSS
- Achieve CalDAV client compatibility (Apple Calendar, Thunderbird, ...) by having a local proxy translate CalDAV requests into CSS queries

Where we are

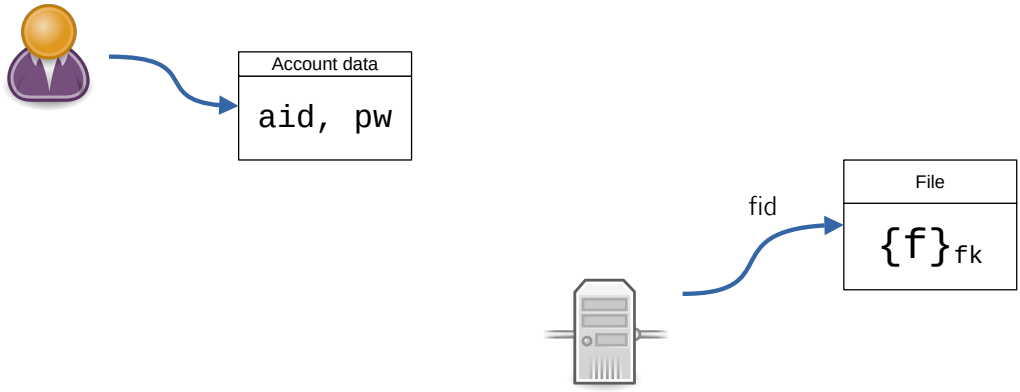
- Identified some blockers in original CSS design
- Extended CSS security model to account for solutions to blockers
- Outlined strategies for search, attendance, and invitations, without relying on heavy machinery
- Working on implementation, paper is WIP

CSS overview

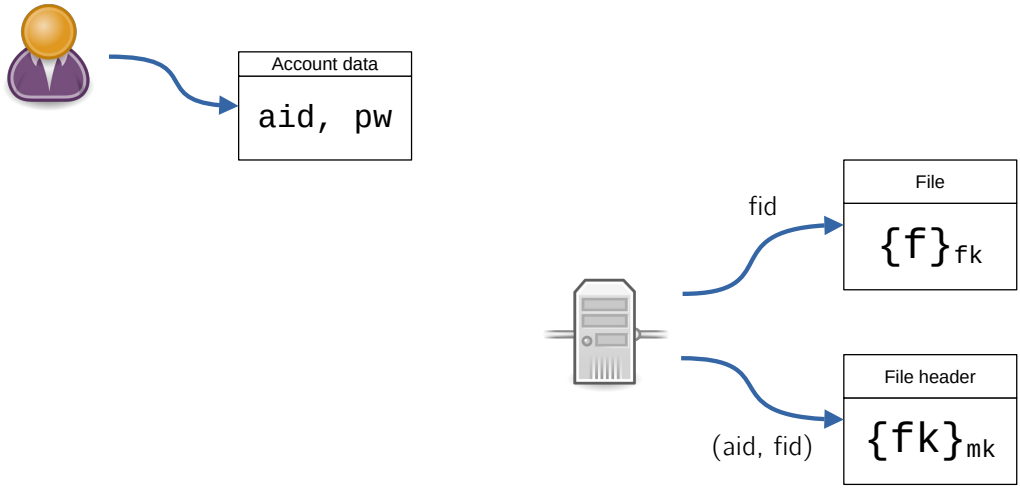
CSS overview



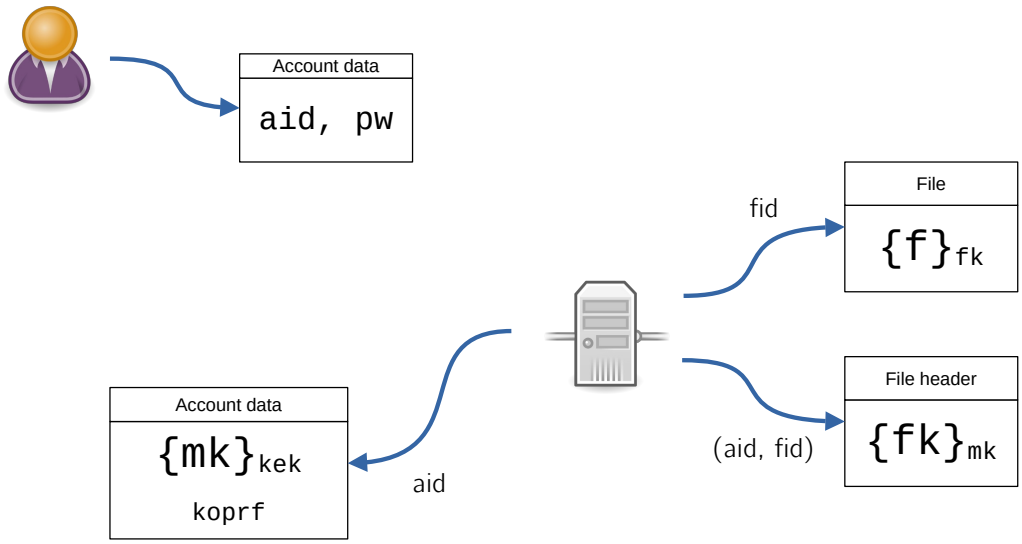
CSS overview



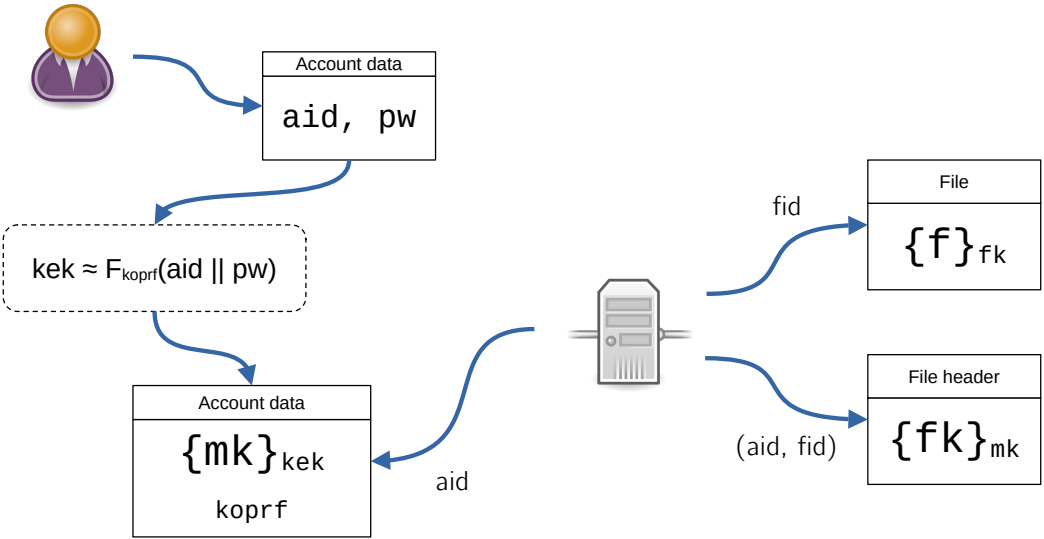
CSS overview



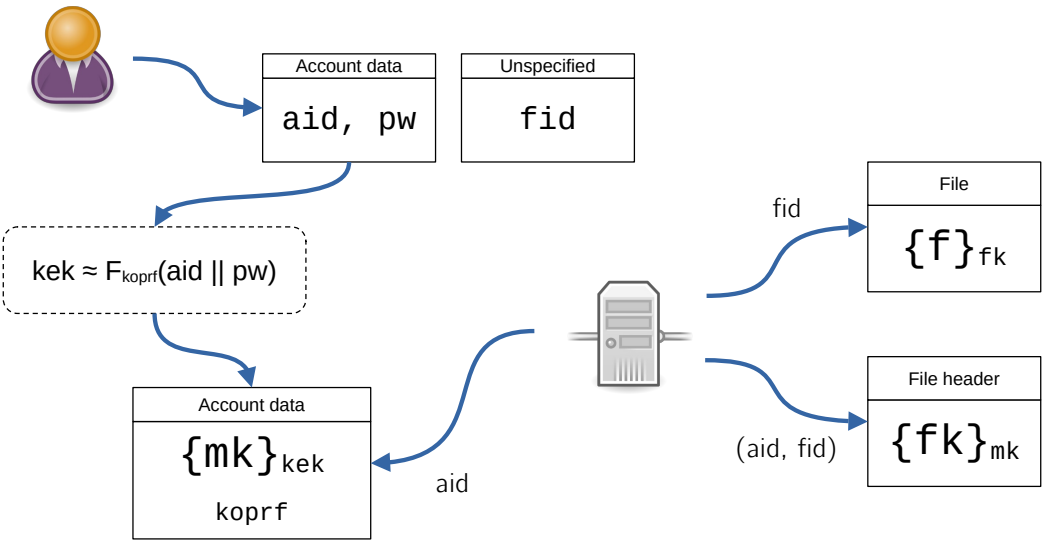
CSS overview



CSS overview



CSS overview



CSS for calendars: what works, what doesn't

Security requirements

- Password updates: ✓
- In-transit encryption: possible

CSS for calendars: what works, what doesn't

Security requirements

- Password updates: ✓
- In-transit encryption: possible

Functionality requirements

- Multi-user support: ✓
- Directory structure: possible
- Push notifications: ✗
- Read-write permissions: ✗
- Range-query search: ✗

CSS for calendars: what works, what doesn't

Security requirements

- Password updates: ✓
- In-transit encryption: possible

Functionality requirements

- Multi-user support: ✓
- Directory structure: possible
- Push notifications: ✗
- Read-write permissions: ✗
- Range-query search: ✗

We propose minor changes to CSS to add some of this functionality: "CSS++"

Security requirements

In-transit encryption

Security requirements

In-transit encryption

- The original CSS paper focuses on the malicious server model, abstracting away client-server channels

Security requirements

In-transit encryption

- The original CSS paper focuses on the malicious server model, abstracting away client-server channels
- User authentication is password-based, from a provable-security angle, hard to compose CSS with TLS without changing ACCE/AKE security model

Security requirements

In-transit encryption

- The original CSS paper focuses on the malicious server model, abstracting away client-server channels
- User authentication is password-based, from a provable-security angle, hard to compose CSS with TLS without changing ACCE/AKE security model
- To shortcircuit the complexity, we build a simplified PAKE and embed it into our construction

Security requirements

In-transit encryption

- The original CSS paper focuses on the malicious server model, abstracting away client-server channels
- User authentication is password-based, from a provable-security angle, hard to compose CSS with TLS without changing ACCE/AKE security model
- To shortcircuit the complexity, we build a simplified PAKE and embed it into our construction

Password updates

Security requirements

In-transit encryption

- The original CSS paper focuses on the malicious server model, abstracting away client-server channels
- User authentication is password-based, from a provable-security angle, hard to compose CSS with TLS without changing ACCE/AKE security model
- To shortcircuit the complexity, we build a simplified PAKE and embed it into our construction

Password updates

- We make the password update procedure explicit

Security requirements

In-transit encryption

- The original CSS paper focuses on the malicious server model, abstracting away client-server channels
- User authentication is password-based, from a provable-security angle, hard to compose CSS with TLS without changing ACCE/AKE security model
- To shortcircuit the complexity, we build a simplified PAKE and embed it into our construction

Password updates

- We make the password update procedure explicit
- This allows us to include it in the security proof

Functionality requirements

- For each functionality specific to CalDAV, many possible implementation approaches

Functionality requirements

- For each functionality specific to CalDAV, many possible implementation approaches

Calendar as a shallow wrapper on storage

Functionality requirements

- For each functionality specific to CalDAV, many possible implementation approaches

Calendar as a shallow wrapper on storage

- Ideally, cryptographic operations only happen inside CSS code

Functionality requirements

- For each functionality specific to CalDAV, many possible implementation approaches

Calendar as a shallow wrapper on storage

- Ideally, cryptographic operations only happen inside CSS code
- Simpler to reason about data confidentiality

Functionality requirements

- For each functionality specific to CalDAV, many possible implementation approaches

Calendar as a shallow wrapper on storage

- Ideally, cryptographic operations only happen inside CSS code
- Simpler to reason about data confidentiality
- We rely heavily on multiple shared files

Functionality requirements

- For each functionality specific to CalDAV, many possible implementation approaches

Calendar as a shallow wrapper on storage

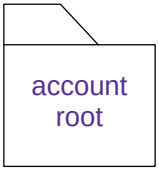
- Ideally, cryptographic operations only happen inside CSS code
- Simpler to reason about data confidentiality
- We rely heavily on multiple shared files

Let's look at

- Directory trees
- Event search
- Event sharing
- RSVP via read/write permissions

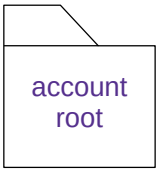
Directory trees in CSS

Directory trees in CSS



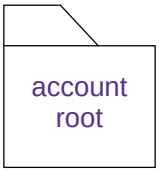

```
{
    fid: “me@a.com-root”,
    children: [
        “me@a.com-work”,
        ...
    ]
}
```

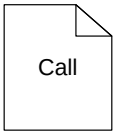
Directory trees in CSS




```
{
    fid: “me@a.com-root”,
    children: [
        “me@a.com-work”,
        ...
    ]
}
{
    fid: “me@a.com-work”,
    children: [
        “19183c3d-99f2-4831-bc70-8e90b914f116”,
        ...
    ]
}
```

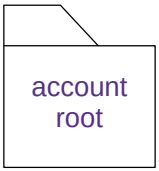
Directory trees in CSS

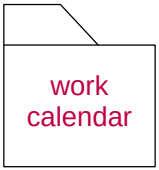


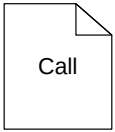



```
{
    fid: “me@a.com-root”,
    children: [
        “me@a.com-work”,
        ...
    ]
}
{
    fid: “me@a.com-work”,
    children: [
        “19183c3d-99f2-4831-bc70-8e90b914f116”,
        ...
    ]
}
{
    fid: “19183c3d-99f2-4831-bc70-8e90b914f116”,
    desc: “Call”,
    time: “...”,
    invitees: “...”
}
```

Directory trees in CSS



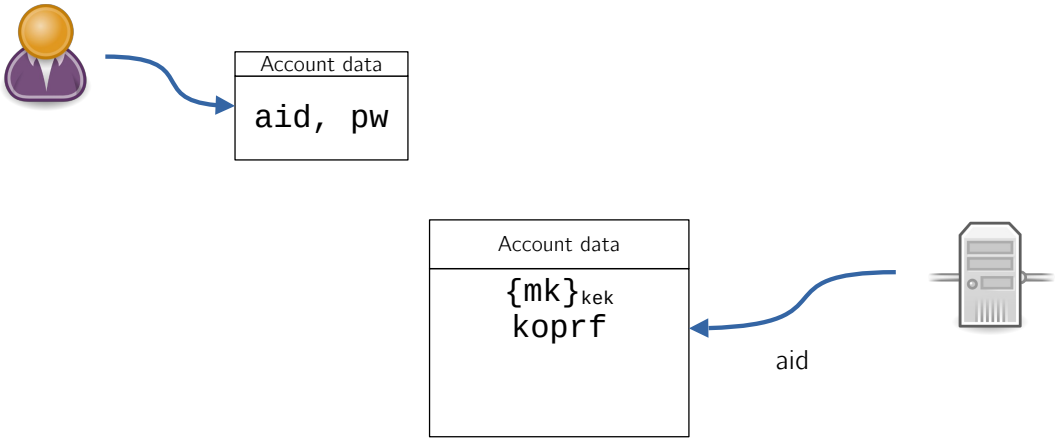




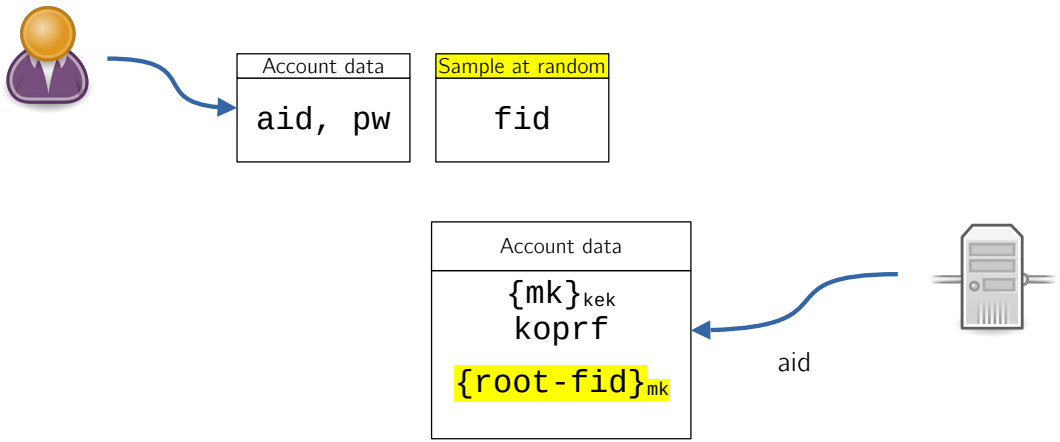

```
{
    fid: “me@a.com-root”,
    children: [
        “me@a.com-work”,
        ...
    ]
}
{
    fid: “me@a.com-work”,
    children: [
        “19183c3d-99f2-4831-bc70-8e90b914f116”,
        ...
    ]
}
{
    fid: “19183c3d-99f2-4831-bc70-8e90b914f116”,
    desc: “Call”,
    time: “...”,
    invitees: “...”
}
```

No guarantee that root fid is not taken

Directory trees in CSS

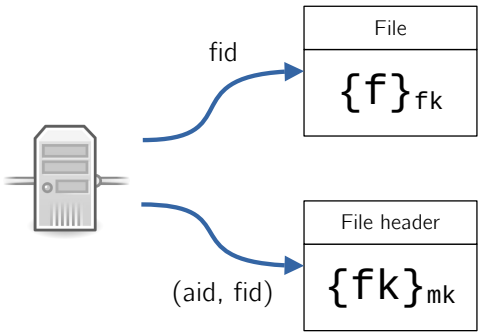


Directory trees in CSS

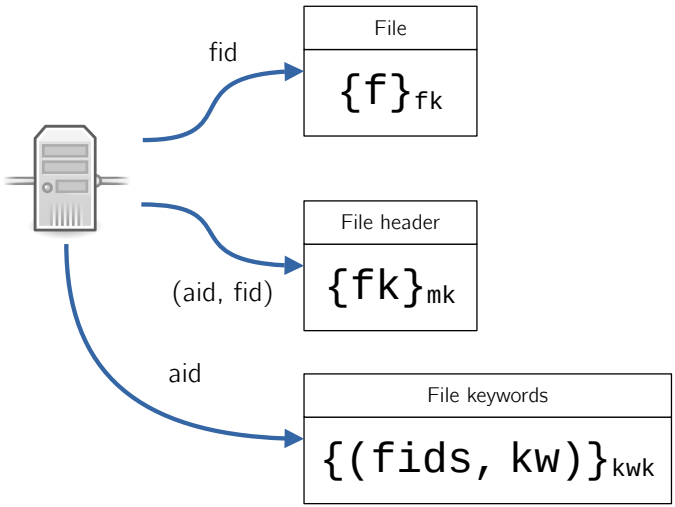


Range/search queries

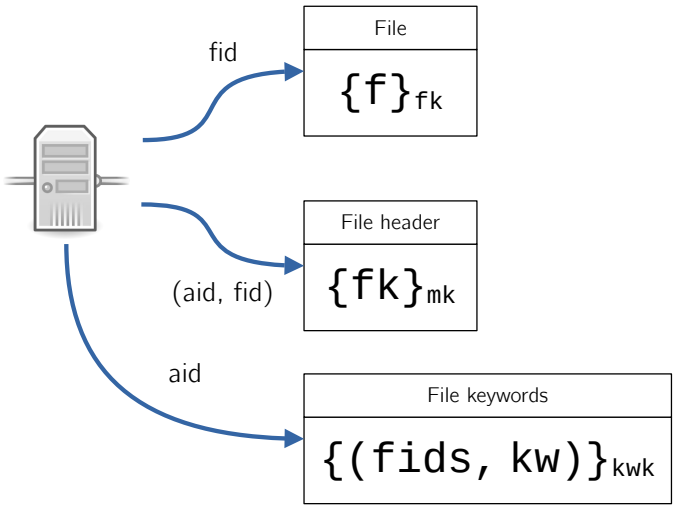
Range/search queries

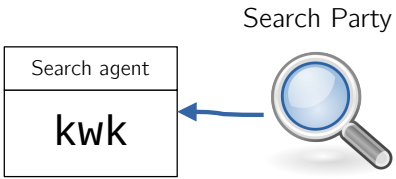


Range/search queries



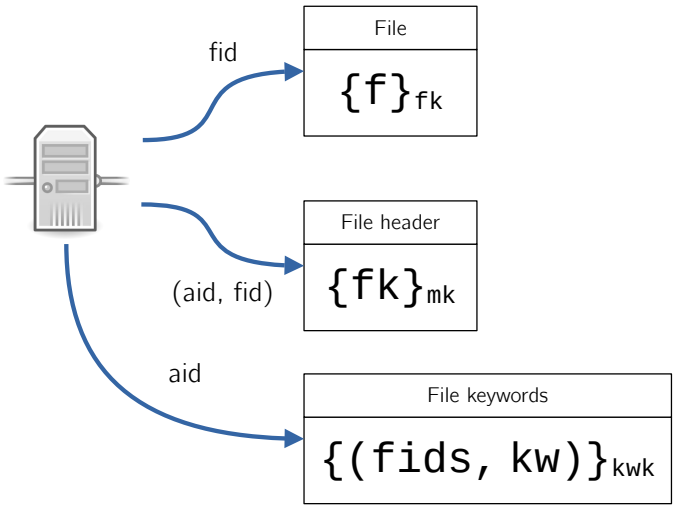
Range/search queries

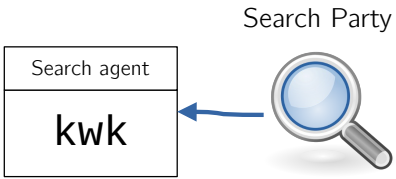




- Leak search metadata to chosen Search Party

Range/search queries





- Leak search metadata to chosen Search Party
- Could implement this as a simple encrypted file shared with the Search Party

Search agent deployment models

Search Party

Search agent deployment models

Search Party

Cloud provider as Search Agent

- Leaks all metadata to server, simple implementation

Search agent deployment models

Search Party

Cloud provider as Search Agent

- Leaks all metadata to server, simple implementation

Trusted third party as Search Agent (eg, home server!)

- Think self-hosting, but if you get hacked only metadata is leaked
- Harder to configure

Search agent deployment models

Search Party

Cloud provider as Search Agent

- Leaks all metadata to server, simple implementation

Trusted third party as Search Agent (eg, home server!)

- Think self-hosting, but if you get hacked only metadata is leaked
- Harder to configure

Oneself as Search Agent

- No leakage, simple to set
- Requires keeping metadata on disk at all times

Search agent deployment models

Search Party

Cloud provider as Search Agent

- Leaks all metadata to server, simple implementation

Trusted third party as Search Agent (eg, home server!)

- Think self-hosting, but if you get hacked only metadata is leaked
- Harder to configure

Oneself as Search Agent

- No leakage, simple to set
- Requires keeping metadata on disk at all times

If you like controversy

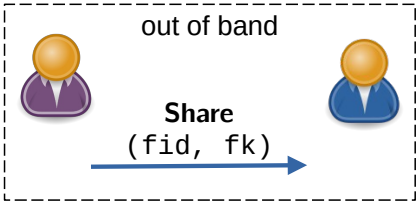
- Add an *extra* layer of leaky searchable encryption (SE)
- Store $\texttt{\{SE(fids, kw)\}}_{\texttt{kwk}}$ in place of $\texttt{\{(fids, kw)\}}_{\texttt{kwk}}$

CSS file sharing vs event sharing, and R/W access

CSS file sharing

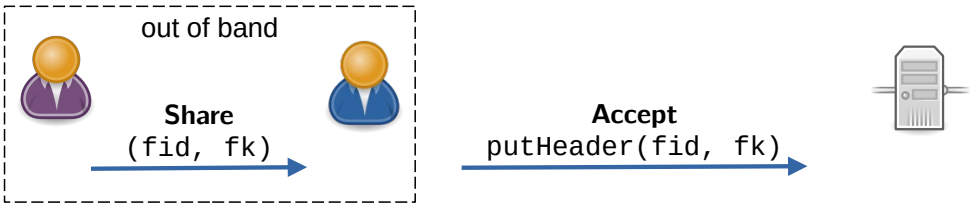
CSS file sharing vs event sharing, and R/W access

CSS file sharing



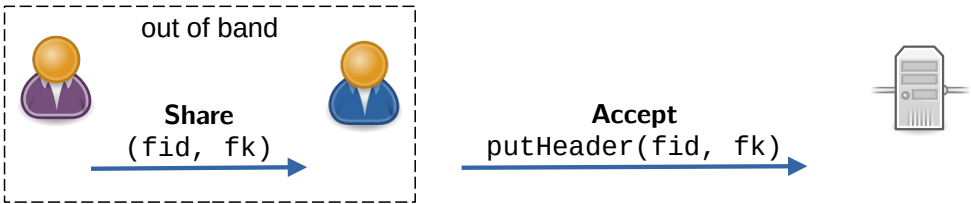
CSS file sharing vs event sharing, and R/W access

CSS file sharing



CSS file sharing vs event sharing, and R/W access

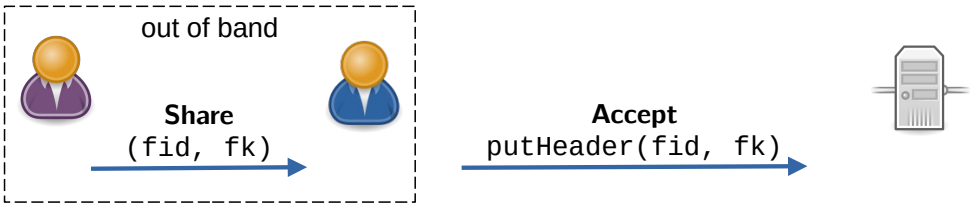
CSS file sharing



- Requires communicating with receiver every time a file is shared

CSS file sharing vs event sharing, and R/W access

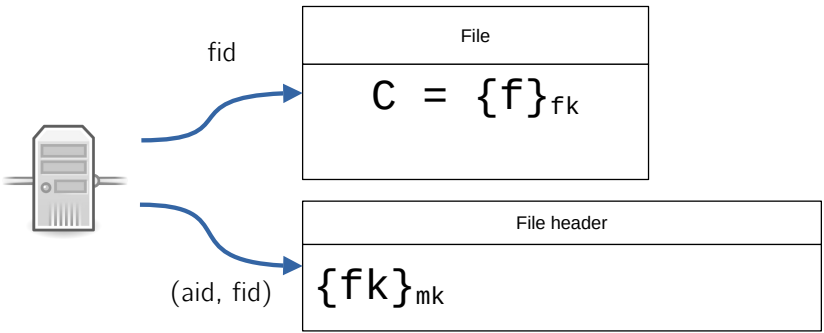
CSS file sharing



- Requires communicating with receiver every time a file is shared
- Receiver becomes equal owner of the file

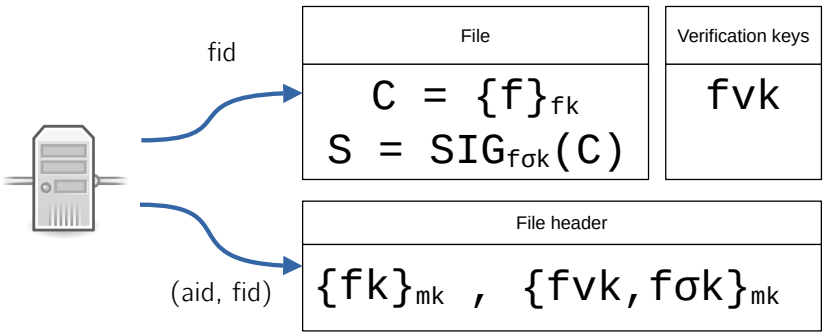
CSS file sharing vs event sharing, and R/W access

Adding read/write permissions to file



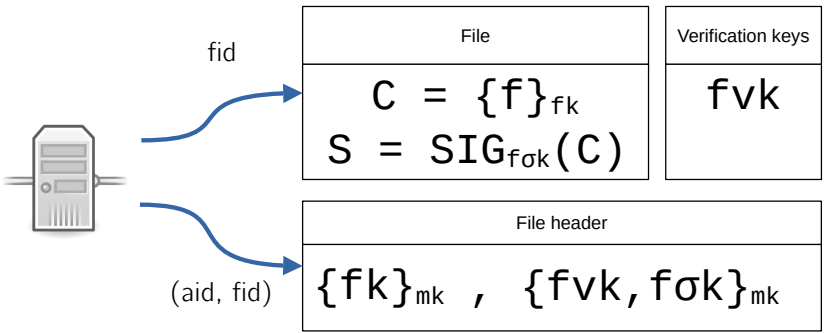
CSS file sharing vs event sharing, and R/W access

Adding read/write permissions to file



CSS file sharing vs event sharing, and R/W access

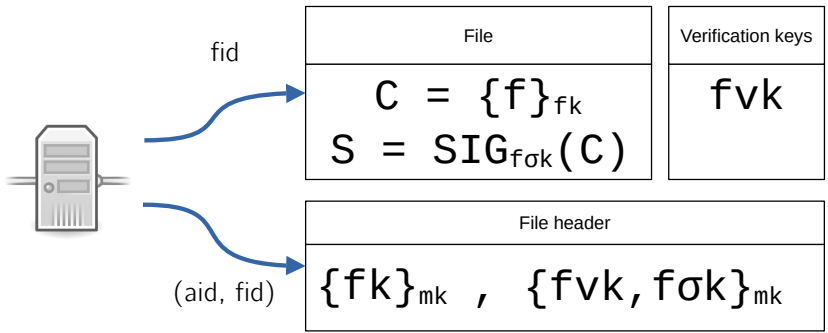
Adding read/write permissions to file



- At time of file creation, add `(fvk, fσk) = SIG.Kg()` to file header and sign `C`

CSS file sharing vs event sharing, and R/W access

Adding read/write permissions to file



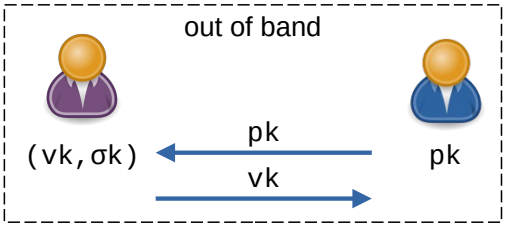
- At time of file creation, add `(fvk, fσk) = SIG.Kg()` to file header and sign `C`
- With every file change, server/other users can verify update

CSS file sharing vs event sharing, and R/W access

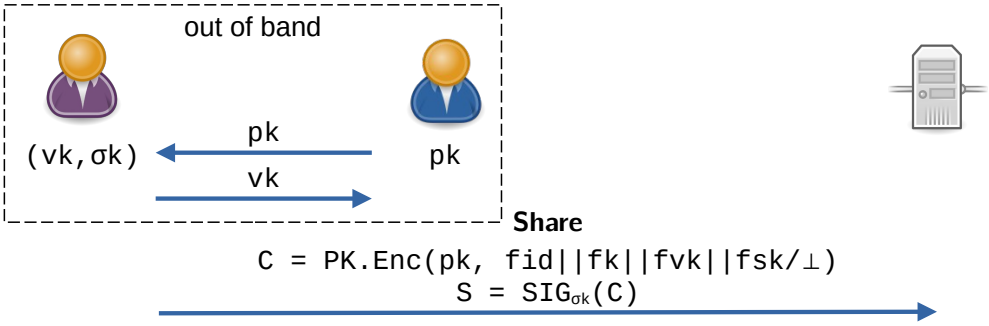
$(vk, \sigma k)$

pk

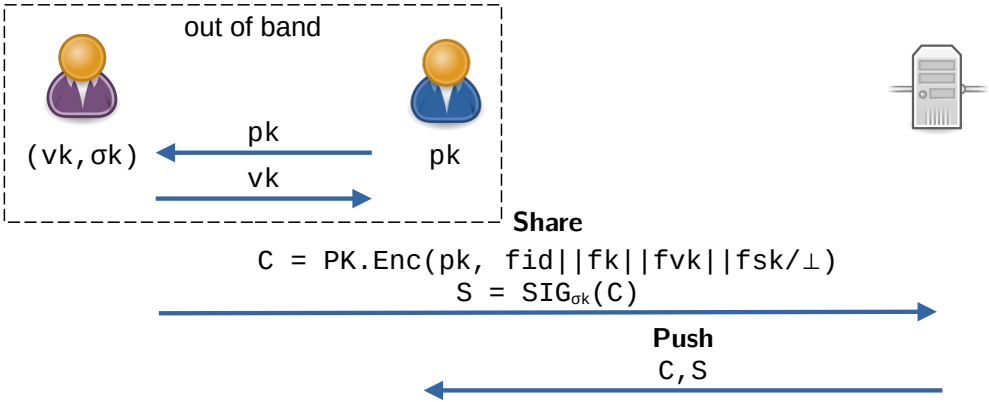
CSS file sharing vs event sharing, and R/W access



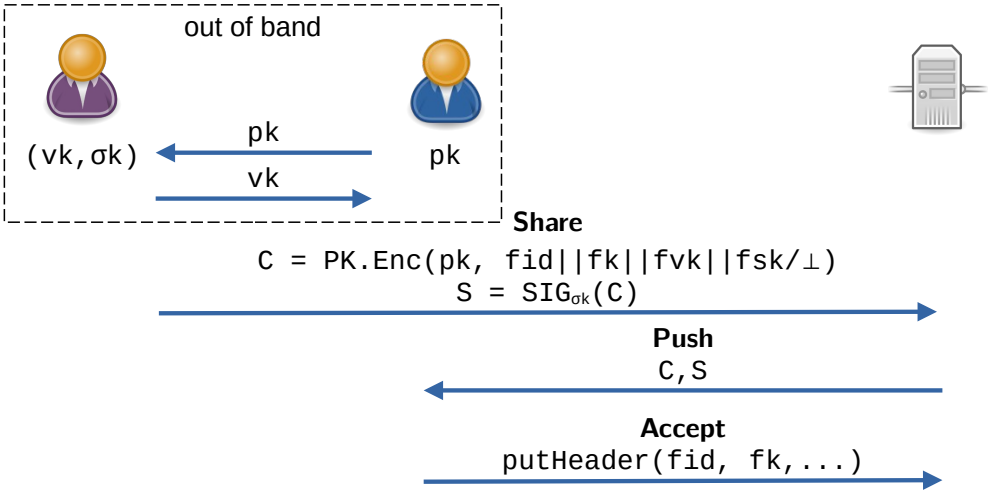
CSS file sharing vs event sharing, and R/W access



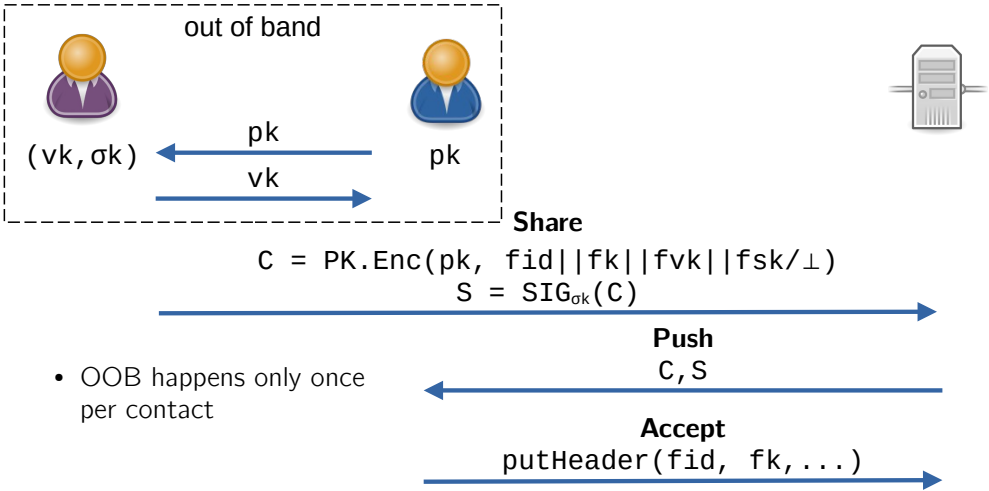
CSS file sharing vs event sharing, and R/W access



CSS file sharing vs event sharing, and R/W access

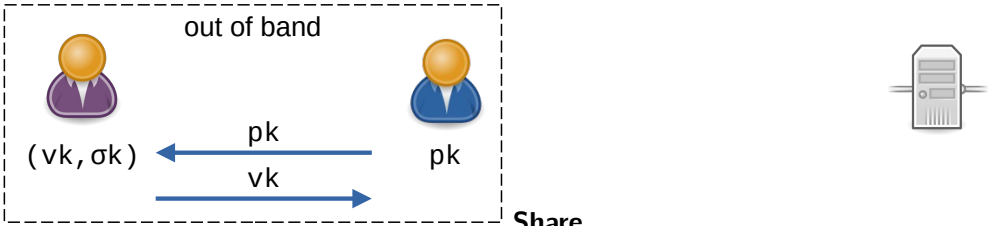


CSS file sharing vs event sharing, and R/W access



- OOB happens only once per contact

CSS file sharing vs event sharing, and R/W access



- OOB happens only once per contact
- If `fsk` is not shared, receiver has read-only access

Event invitations with RSVP support

With server-mediated file sharing and R/W permissions, we can construct event invitations:

Event invitations with RSVP support

With server-mediated file sharing and R/W permissions, we can construct event invitations:

1. Event organizer O identifies invitees A, B, C to a new event

Event invitations with RSVP support

With server-mediated file sharing and R/W permissions, we can construct event invitations:

1. Event organizer O identifies invitees A, B, C to a new event
2. For each guest $G \in \{A, B, C\}$, O creates an RSVP file R_G

Event invitations with RSVP support

With server-mediated file sharing and R/W permissions, we can construct event invitations:

1. Event organizer O identifies invitees A, B, C to a new event
2. For each guest $G \in \{A, B, C\}$, O creates an RSVP file R_G
3. O creates an event file E that includes pointers to the `fid` of each RSVP file R_G

Event invitations with RSVP support

With server-mediated file sharing and R/W permissions, we can construct event invitations:

1. Event organizer O identifies invitees A, B, C to a new event
2. For each guest $G \in \{A, B, C\}$, O creates an RSVP file R_G
3. O creates an event file E that includes pointers to the `fid` of each RSVP file R_G
4. Files E, R_A, R_B, R_C are shared to each participant as read-only, with the exception of R_G being writable by party G

Event invitations with RSVP support

With server-mediated file sharing and R/W permissions, we can construct event invitations:

1. Event organizer O identifies invitees A, B, C to a new event
2. For each guest $G \in \{A, B, C\}$, O creates an RSVP file R_G
3. O creates an event file E that includes pointers to the `fid` of each RSVP file R_G
4. Files E, R_A, R_B, R_C are shared to each participant as read-only, with the exception of R_G being writable by party G

If party G decides to RSVP, they simply edit R_G accordingly.

Conclusion

Conclusion

- Online calendars host sensitive information while requiring non-trivial file-sharing functionality

Conclusion

- Online calendars host sensitive information while requiring non-trivial file-sharing functionality
- Natural to attempt instantiating E2EE calendars from CSS

Conclusion

- Online calendars host sensitive information while requiring non-trivial file-sharing functionality
- Natural to attempt instantiating E2EE calendars from CSS
- We propose small tweaks to CSS to improve applicability to calendars, likely useful elsewhere
  - We've shown a non-exhaustive list!

Conclusion

- Online calendars host sensitive information while requiring non-trivial file-sharing functionality
- Natural to attempt instantiating E2EE calendars from CSS
- We propose small tweaks to CSS to improve applicability to calendars, likely useful elsewhere
  - We've shown a non-exhaustive list!

Thank you