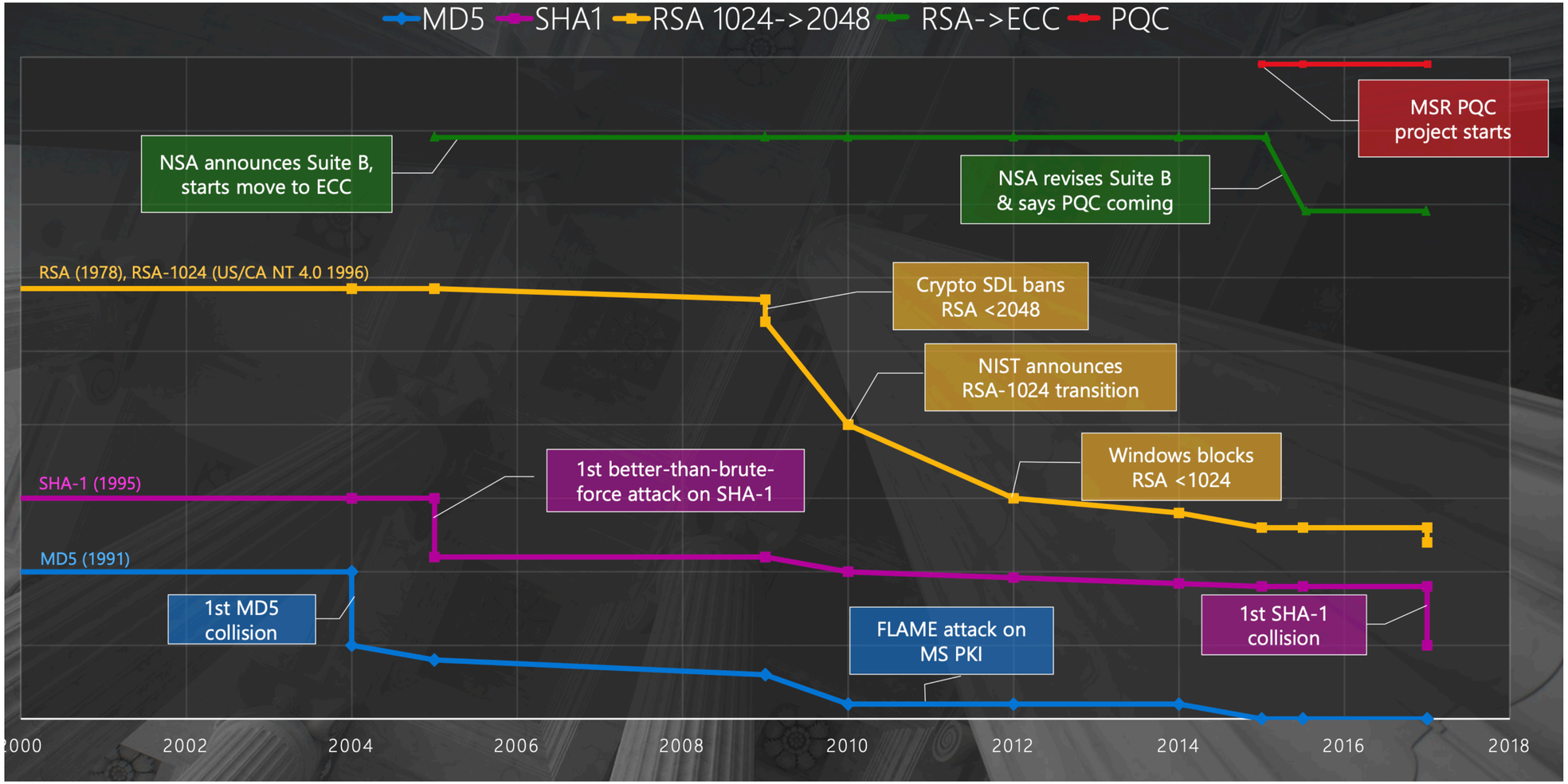
Agile Cryptography: A Universally Composable Approach

Christian Badertscher
Input Output, Switzerland

Michele Ciampi
The Universtiy of Edinburgh, UK

Aggelos Kiayias
The University of Edinburgh &
Input Output, UK

Cryptographic algorithms need updates



Credit to Brian LaMacchia. The Long Road Ahead to Transition to Post-Quantum Cryptography, Invited Opening Keynote, IEEE Secure Development Conference 2022, Atlanta, GA, October 18-20

Cryptographic agility

“The ability to reconfigure an application or system with a different cryptographic algorithm (or implementation)”*

* David Ott, Kenny Paterson, Dennis Moreau. Where Is the Research on Cryptographic Transition and Agility? Communications of the ACM, April 2023, Vol. 66 No. 4, Pages 29-32

Cryptographic agility

“The ability to reconfigure an application or system with a different cryptographic algorithm (or implementation)”*

- Ongoing transition to new algorithms
- Updates or changes to an implementation
- Modifying key configuration parameters
- Retiring deprecated algorithms

* David Ott, Kenny Paterson, Dennis Moreau. Where Is the Research on Cryptographic Transition and Agility? Communications of the ACM, April 2023, Vol. 66 No. 4, Pages 29-32

Cryptographic agility

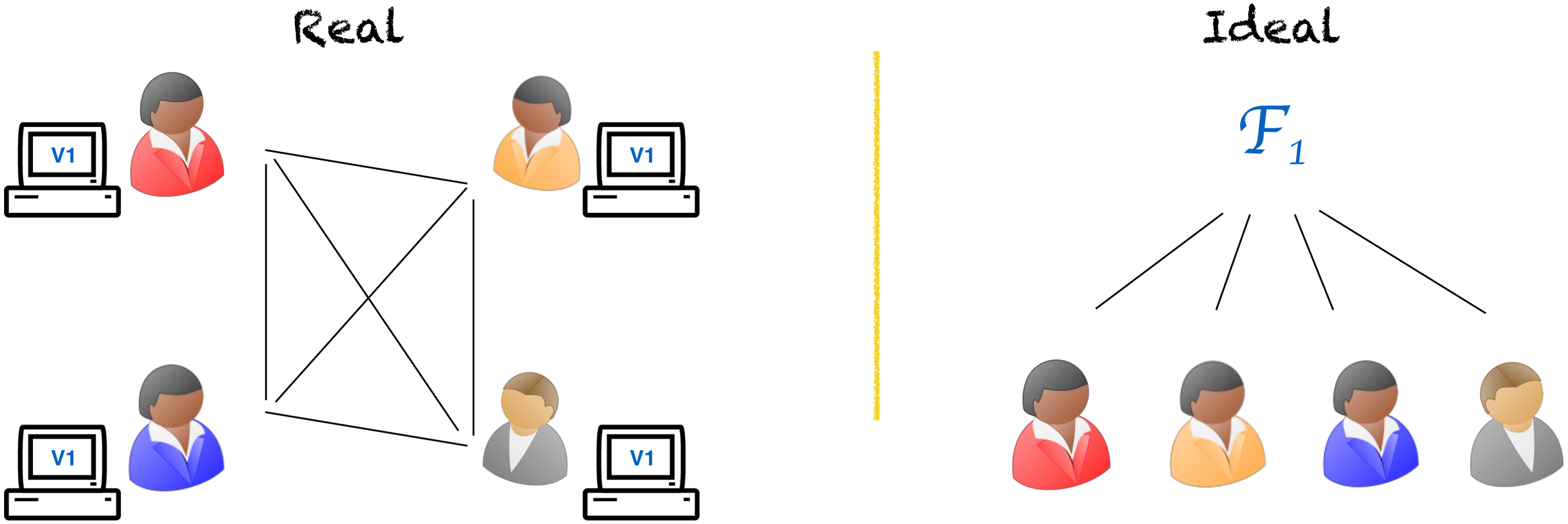
"The ability to reconfigure an application or system with a different cryptographic algorithm (or implementation)"*

Cryptographic protocols need to be updated in practice. Why not in theory as well?

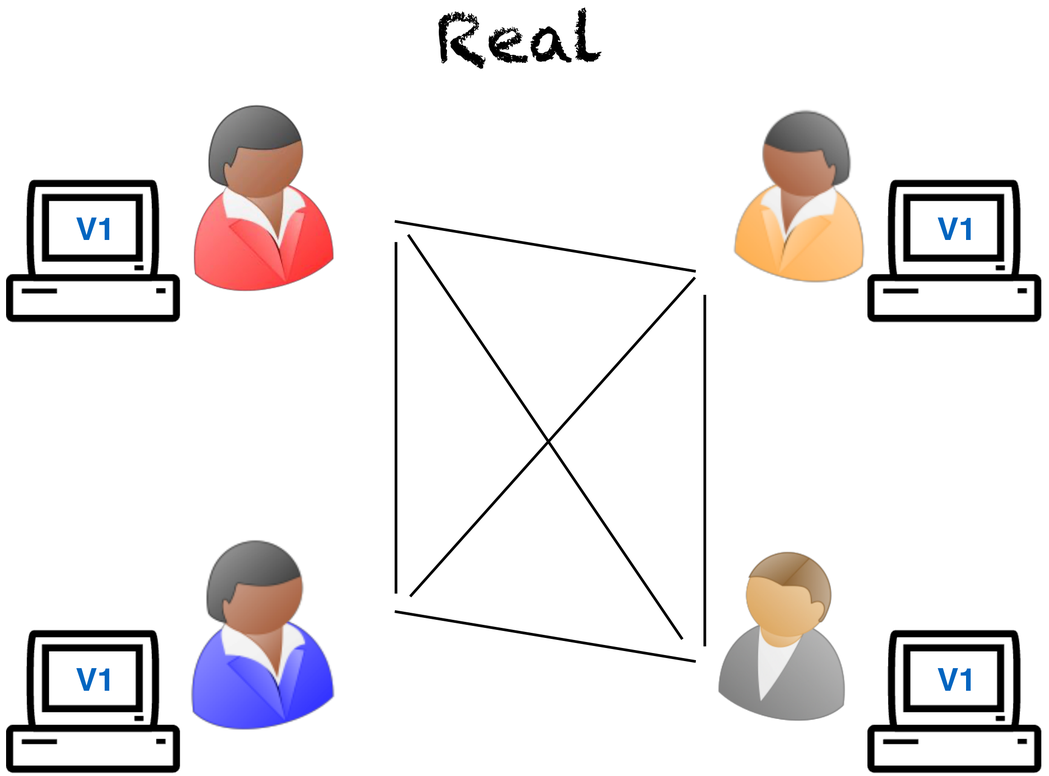
- Ongoing transition to new algorithms
- Updates or changes to an implementation
- Modifying key configuration parameters
- Retiring deprecated algorithms

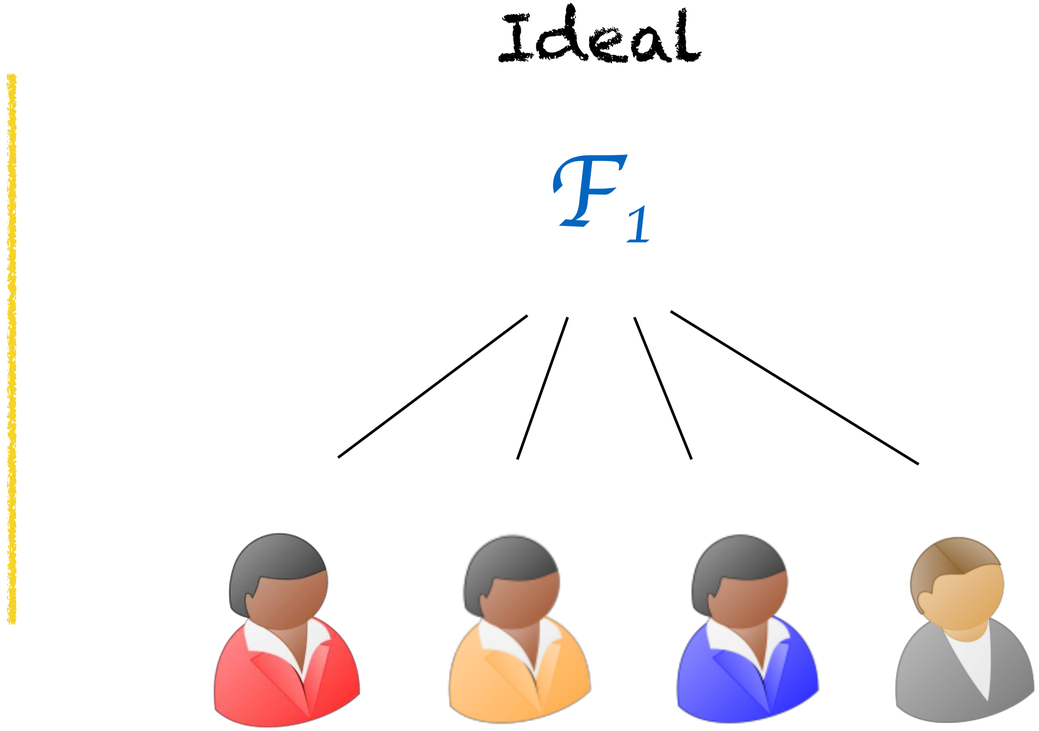
* David Ott, Kenny Paterson, Dennis Moreau. Where Is the Research on Cryptographic Transition and Agility? Communications of the ACM, April 2023, Vol. 66 No. 4, Pages 29-32

Cryptographic agility



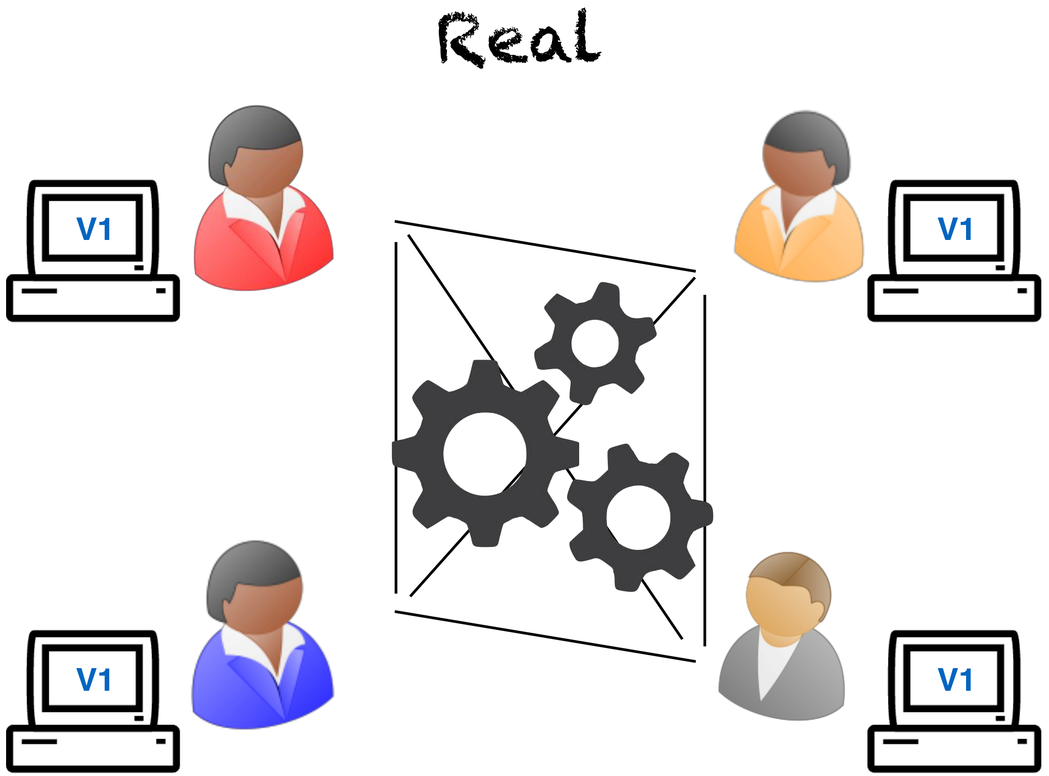
Cryptographic agility

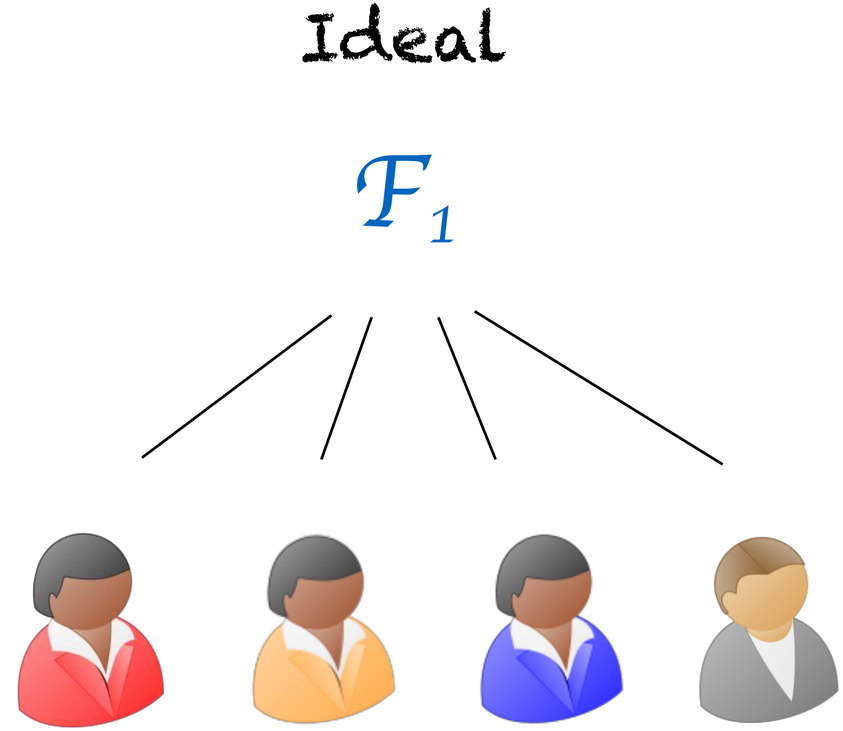




- Recover from broken protocols
- Add additional features

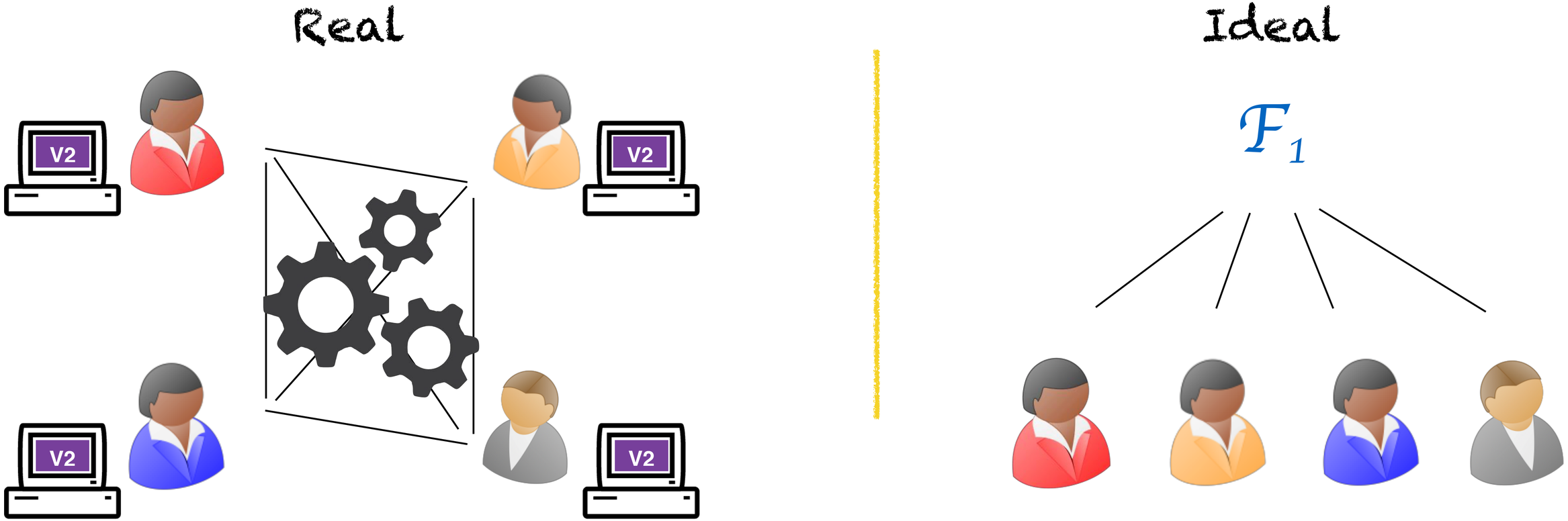
Cryptographic agility





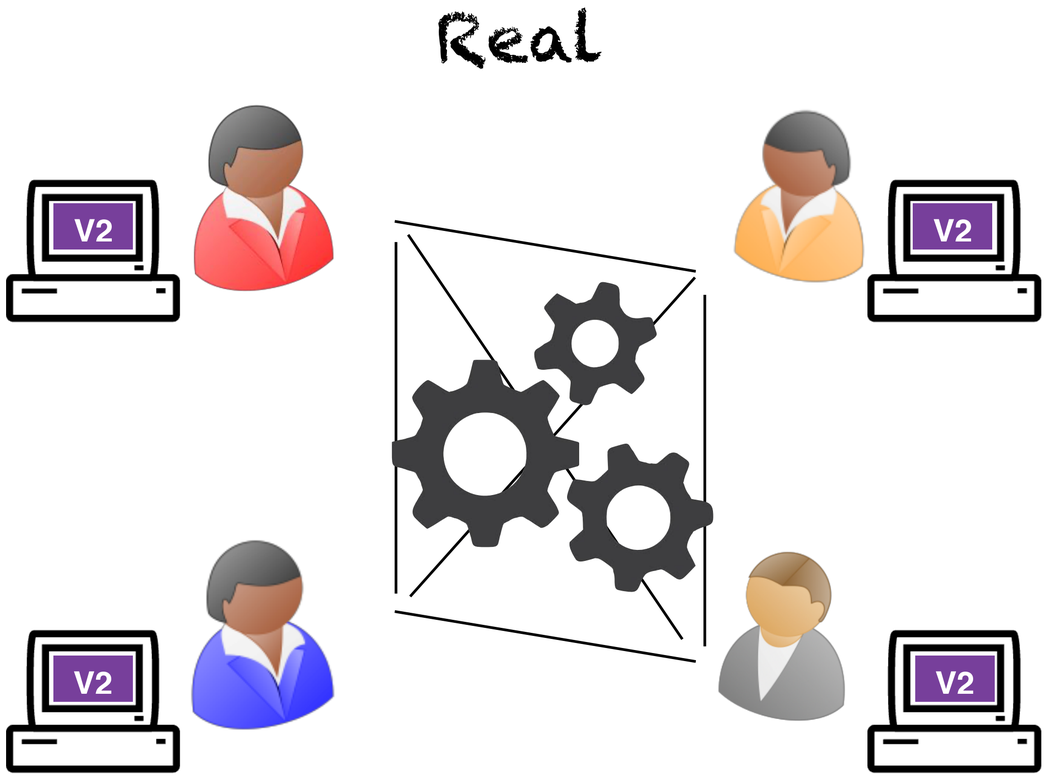
- Recover from broken protocols
- Add additional features

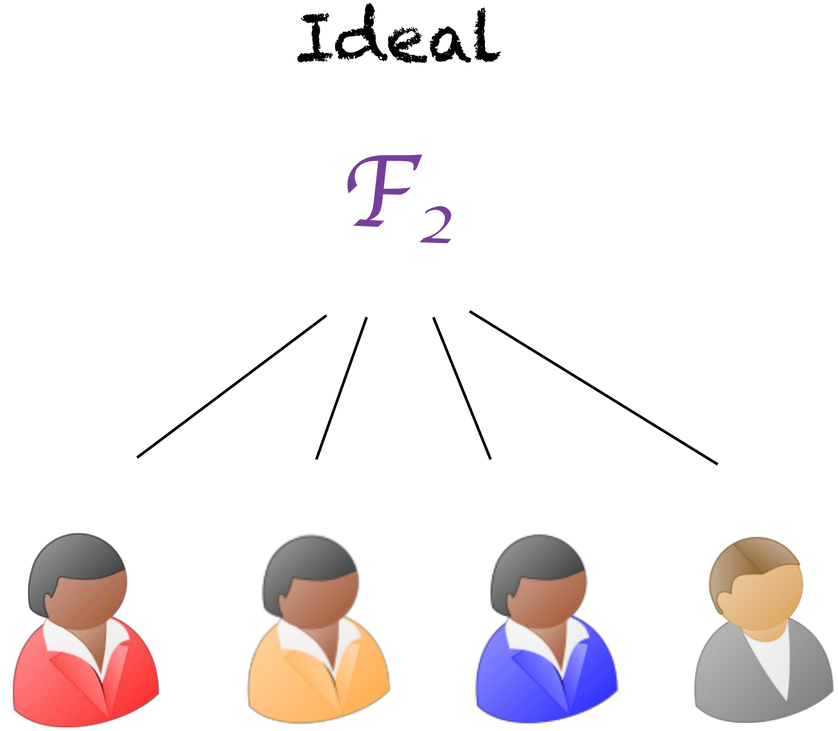
Cryptographic agility



- Recover from broken protocols
- Add additional features

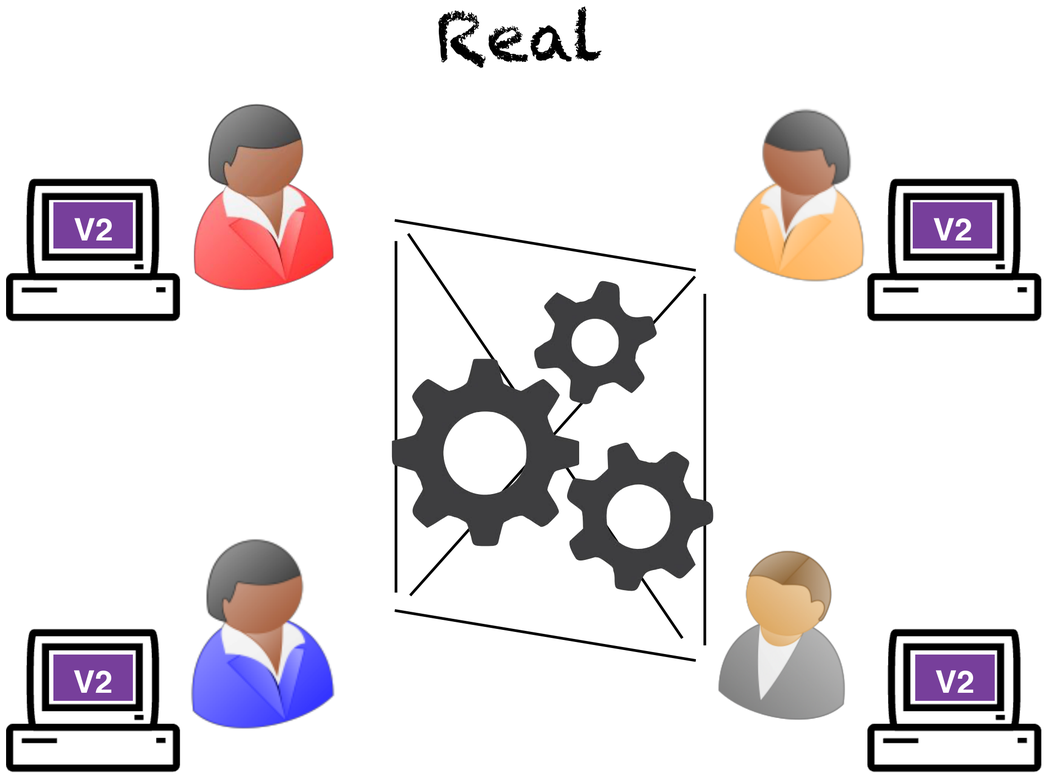
Cryptographic agility

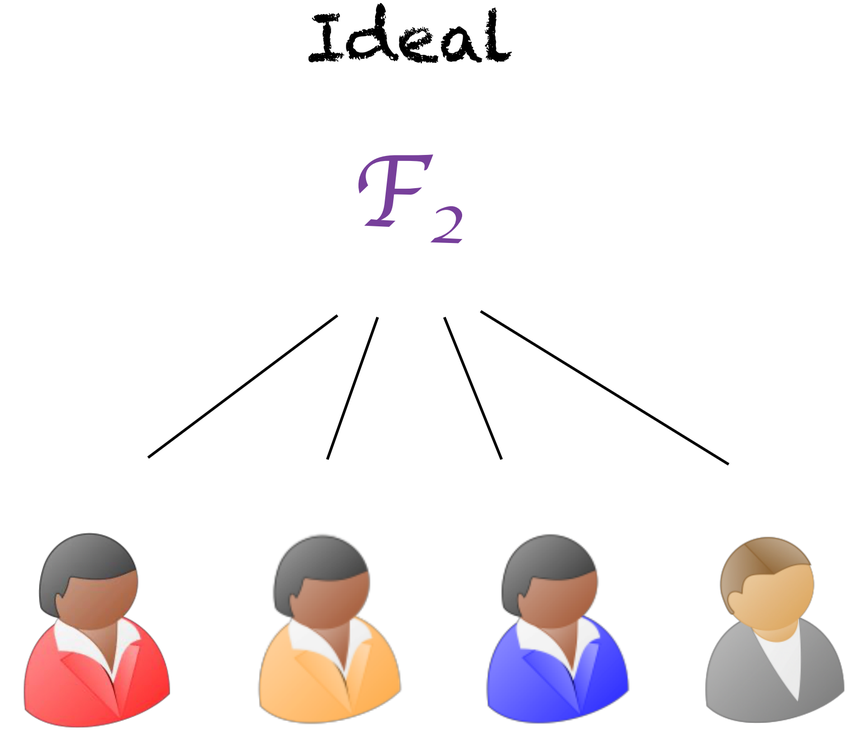




- Recover from broken protocols
- Add additional features

Cryptographic agility

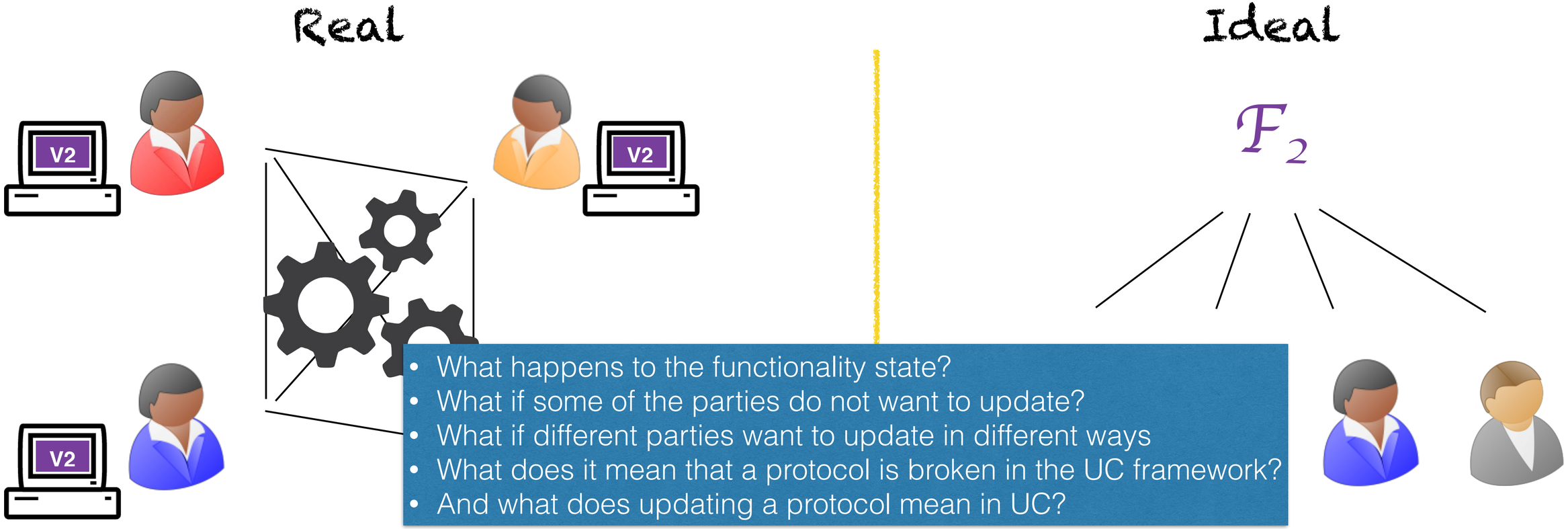




- Recover from broken protocols
- Add additional features

Just corresponds to parties registering to a different functionality, right?

Cryptographic agility

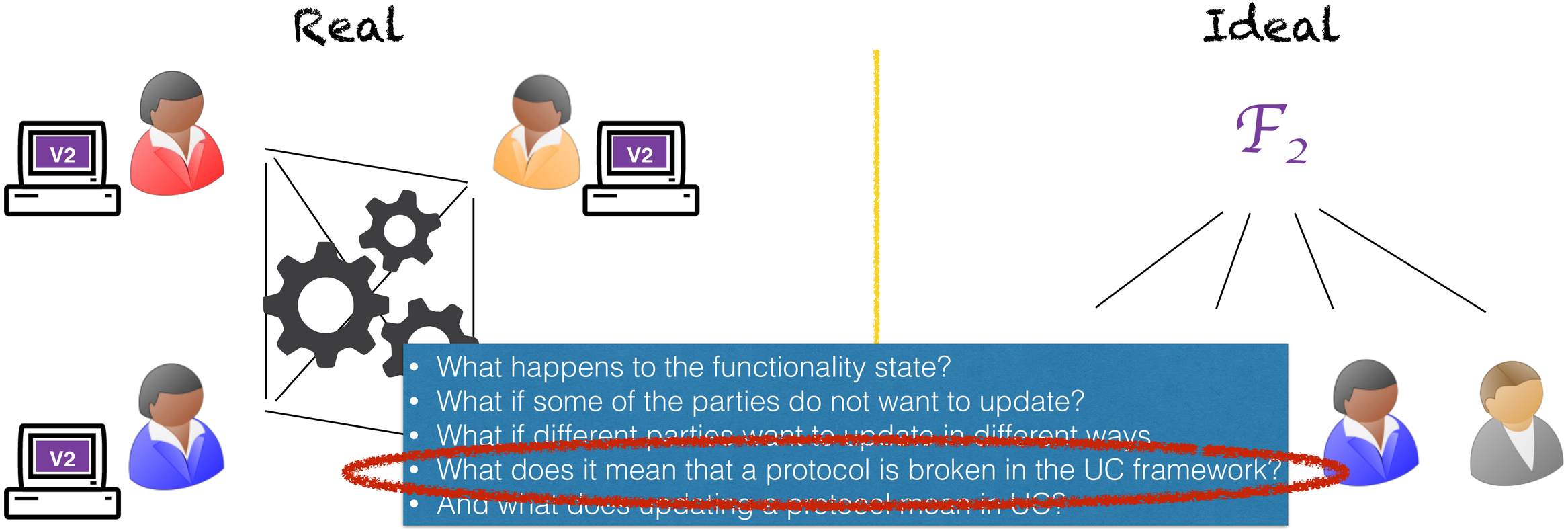


- What happens to the functionality state?
- What if some of the parties do not want to update?
- What if different parties want to update in different ways
- What does it mean that a protocol is broken in the UC framework?
- And what does updating a protocol mean in UC?

- Recover from broken protocols
- Add additional features

Just corresponds to parties registering to a different functionality, right?

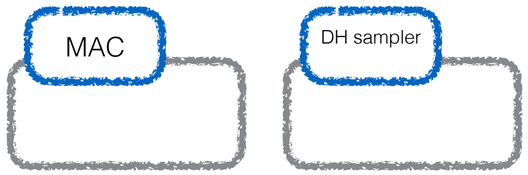
Cryptographic agility

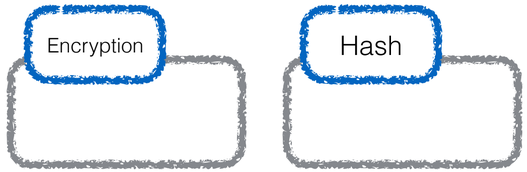


- Recover from broken protocols
- Add additional features

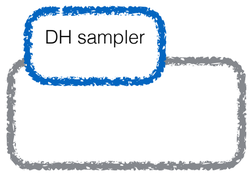
Just corresponds to parties registering to a different functionality, right?

Cryptographic agility





Cryptographic agility

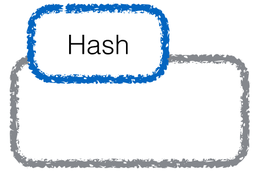


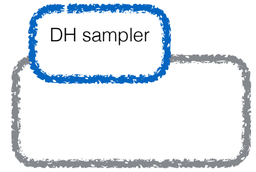
MPC

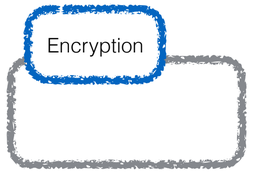
Consensus

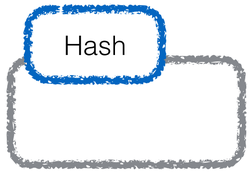
⋮

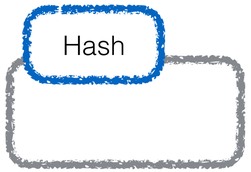
New post-quantum AI-based blockchain for FHE



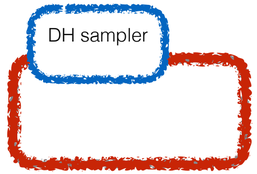








Cryptographic agility

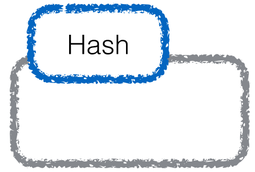


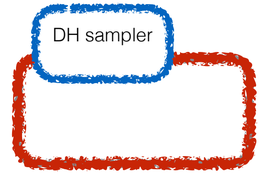
MPC

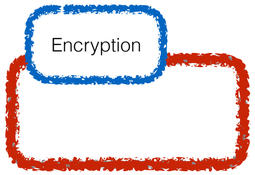
Consensus

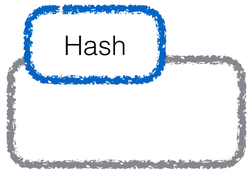
⋮

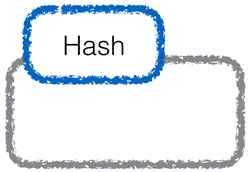
New post-quantum AI-based blockchain for FHE





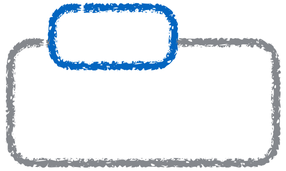






Modelling broken things

Fist contribution

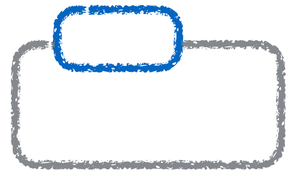


$\mathcal{F}_{comp}$

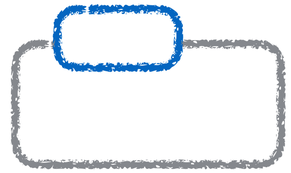
Good mode

Modelling broken things

Fist contribution



(INIT, alg)

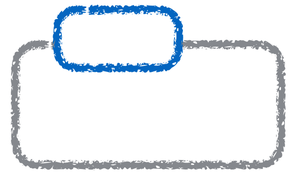


$\mathcal{F}_{comp}$

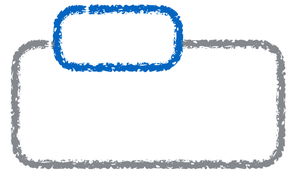
Good mode

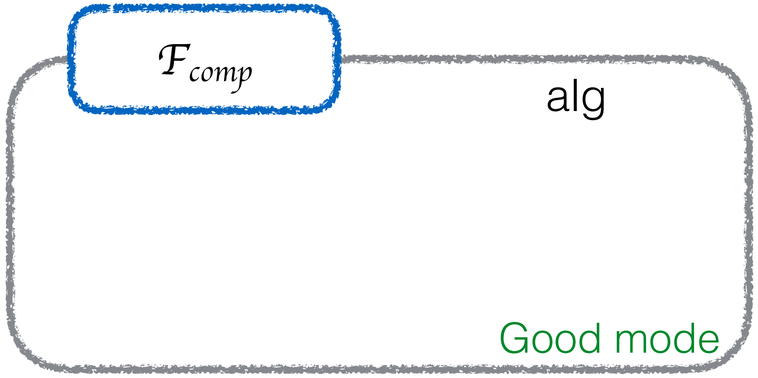
Modelling broken things

Fist contribution



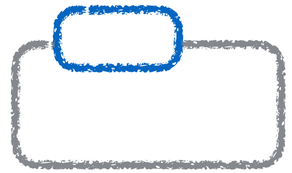
(INIT, alg)

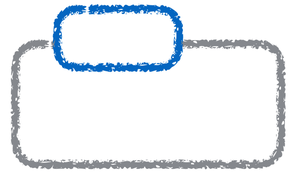




Modelling broken things

Fist contribution





(QUERY, x_1)

$\mathcal{F}_{comp}$

alg

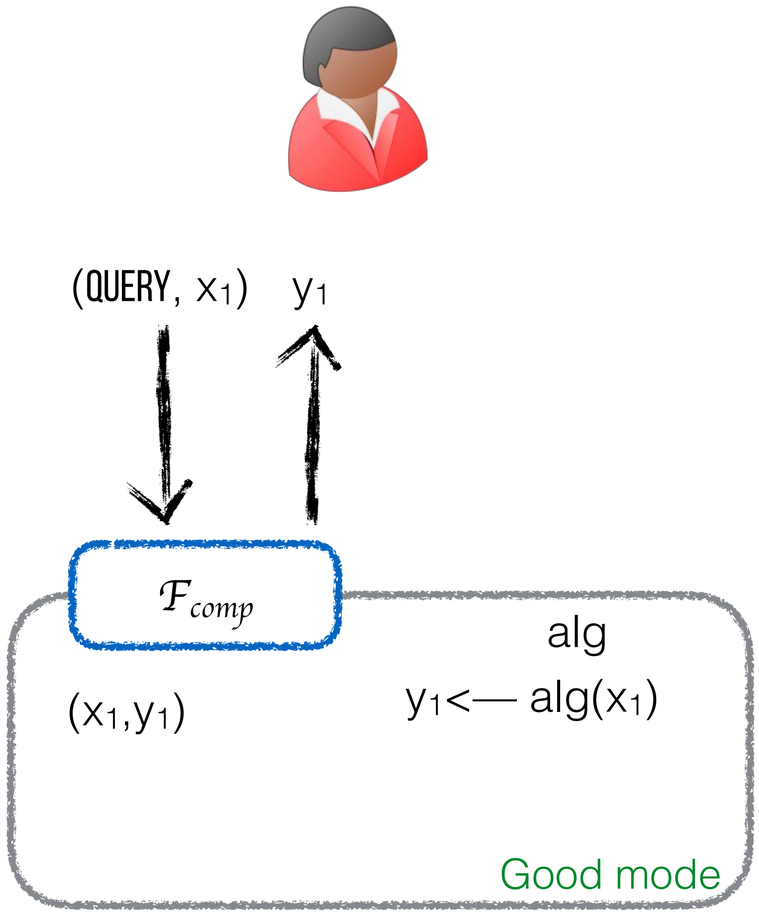
y_1<— alg(x_1)

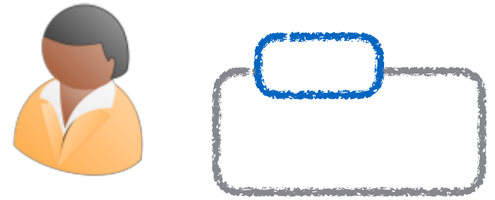
(x_1,y_1)

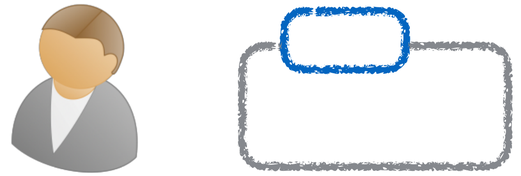
Good mode

Modelling broken things

Fist contribution

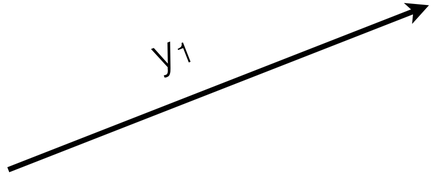


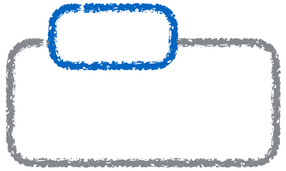


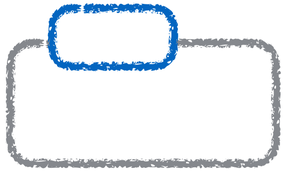


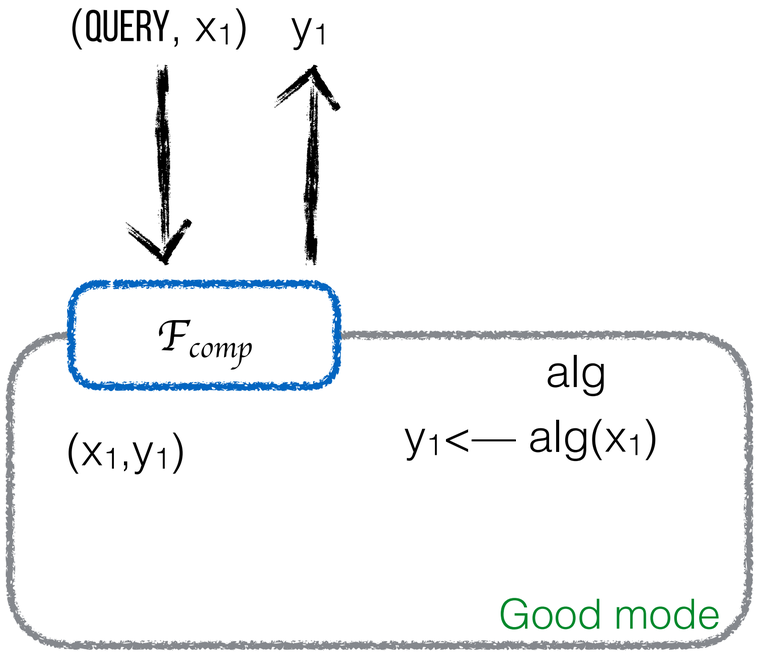
Modelling broken things

Fist contribution



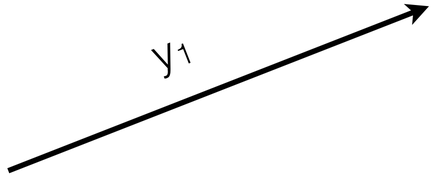


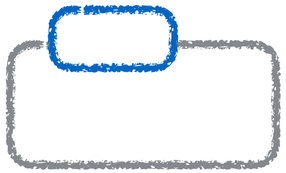


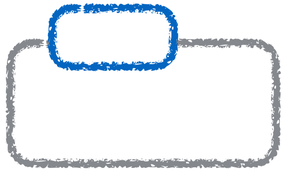


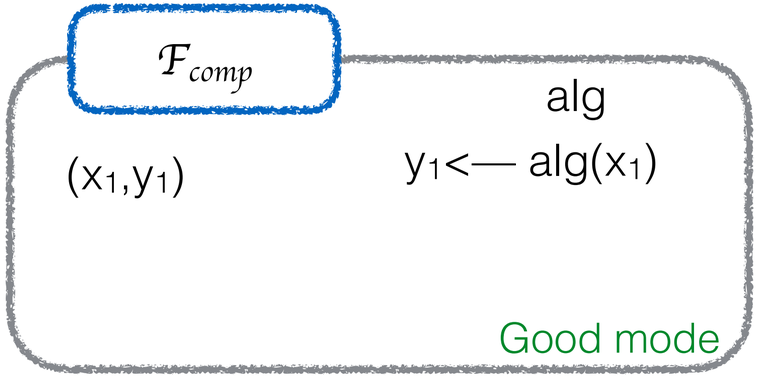
Modelling broken things

Fist contribution



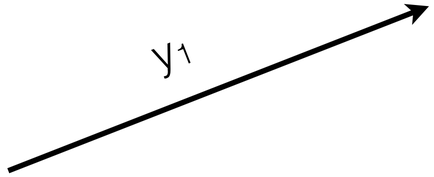


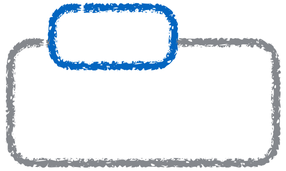


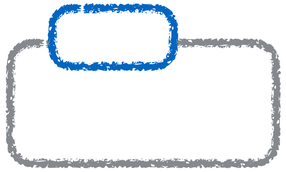


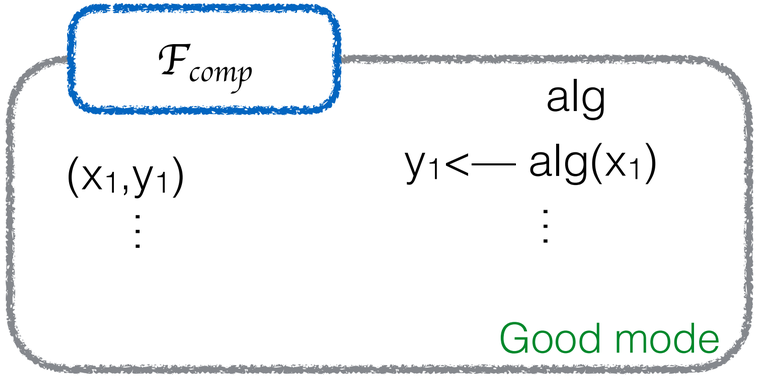
Modelling broken things

Fist contribution



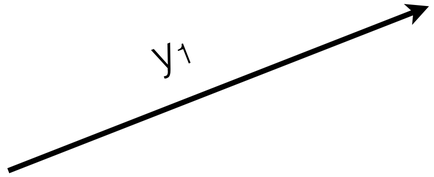


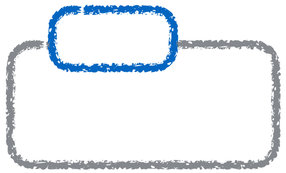


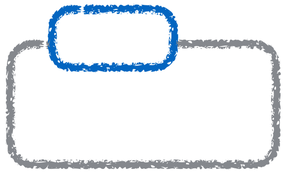


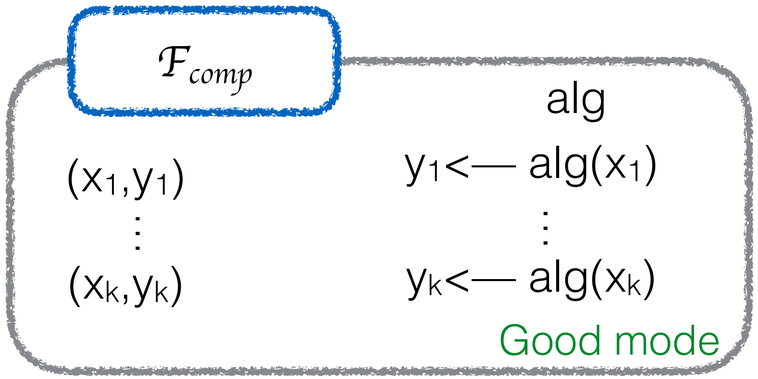
Modelling broken things

Fist contribution



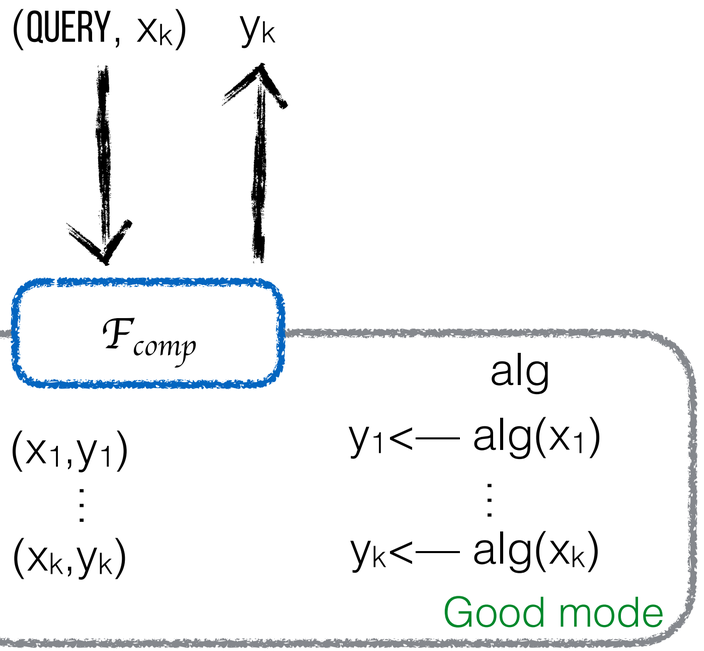


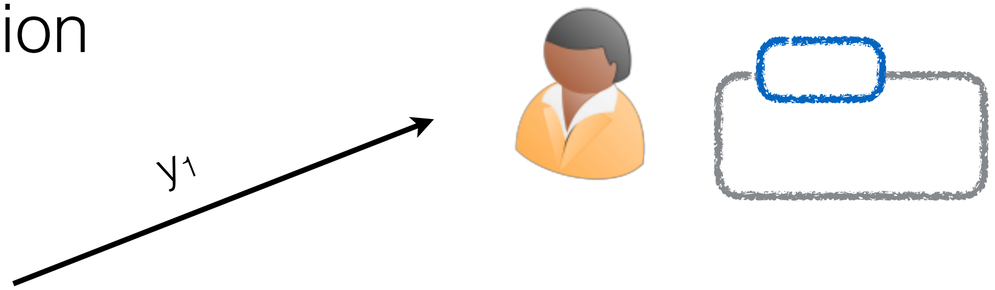


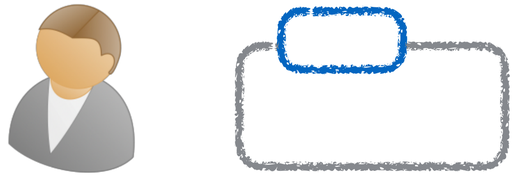


Modelling broken things

Fist contribution

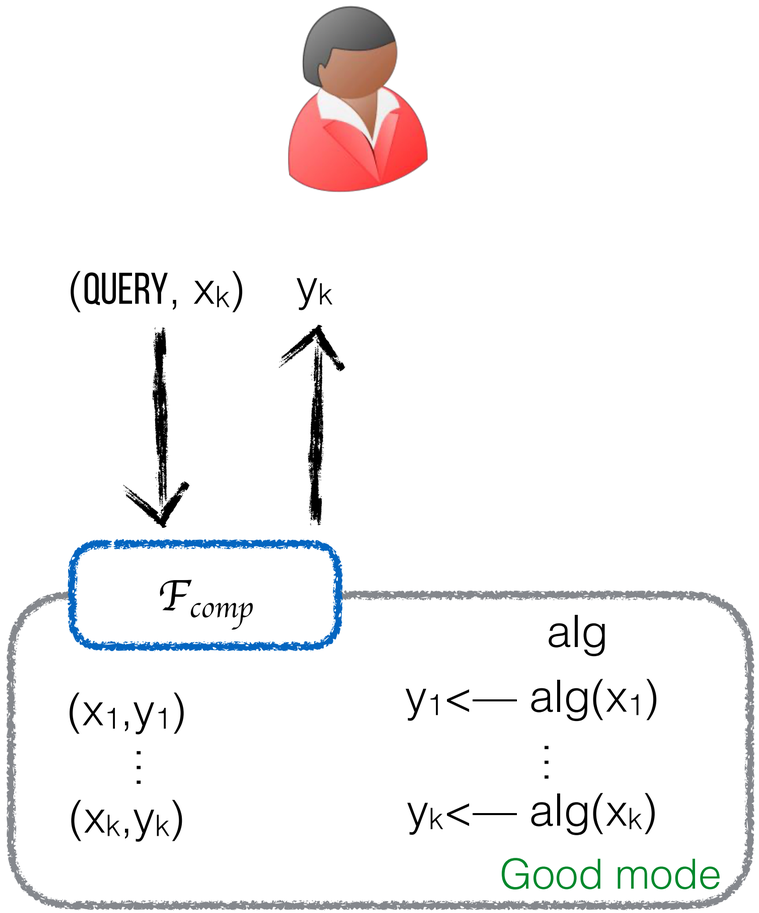


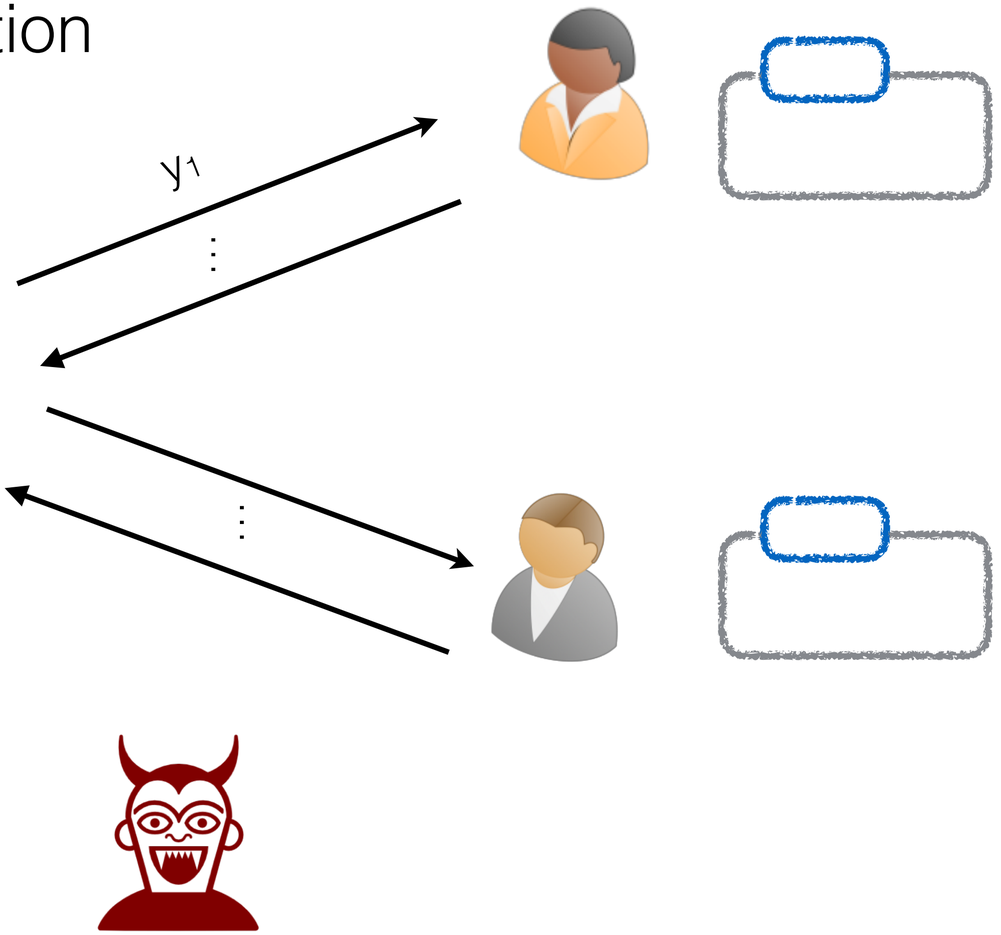




Modelling broken things

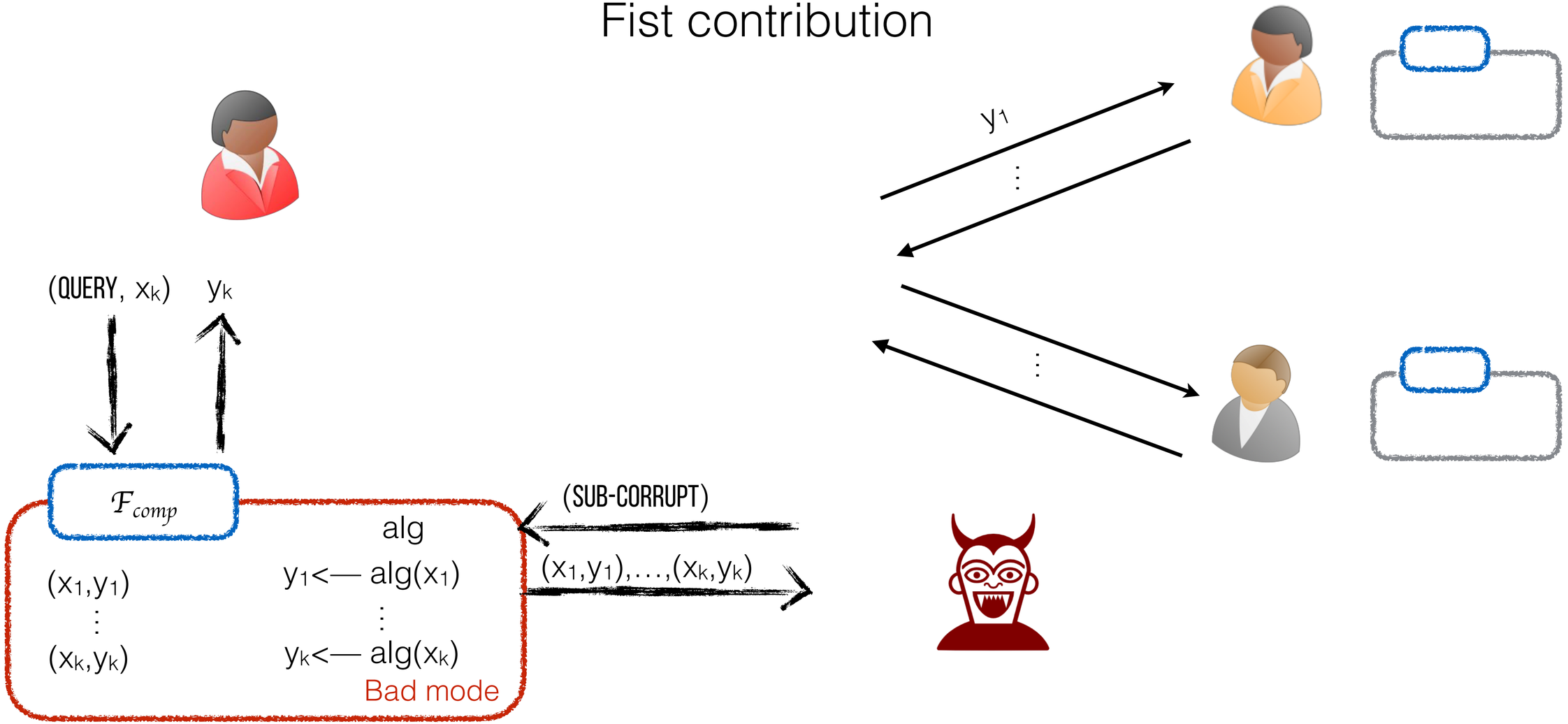
Fist contribution





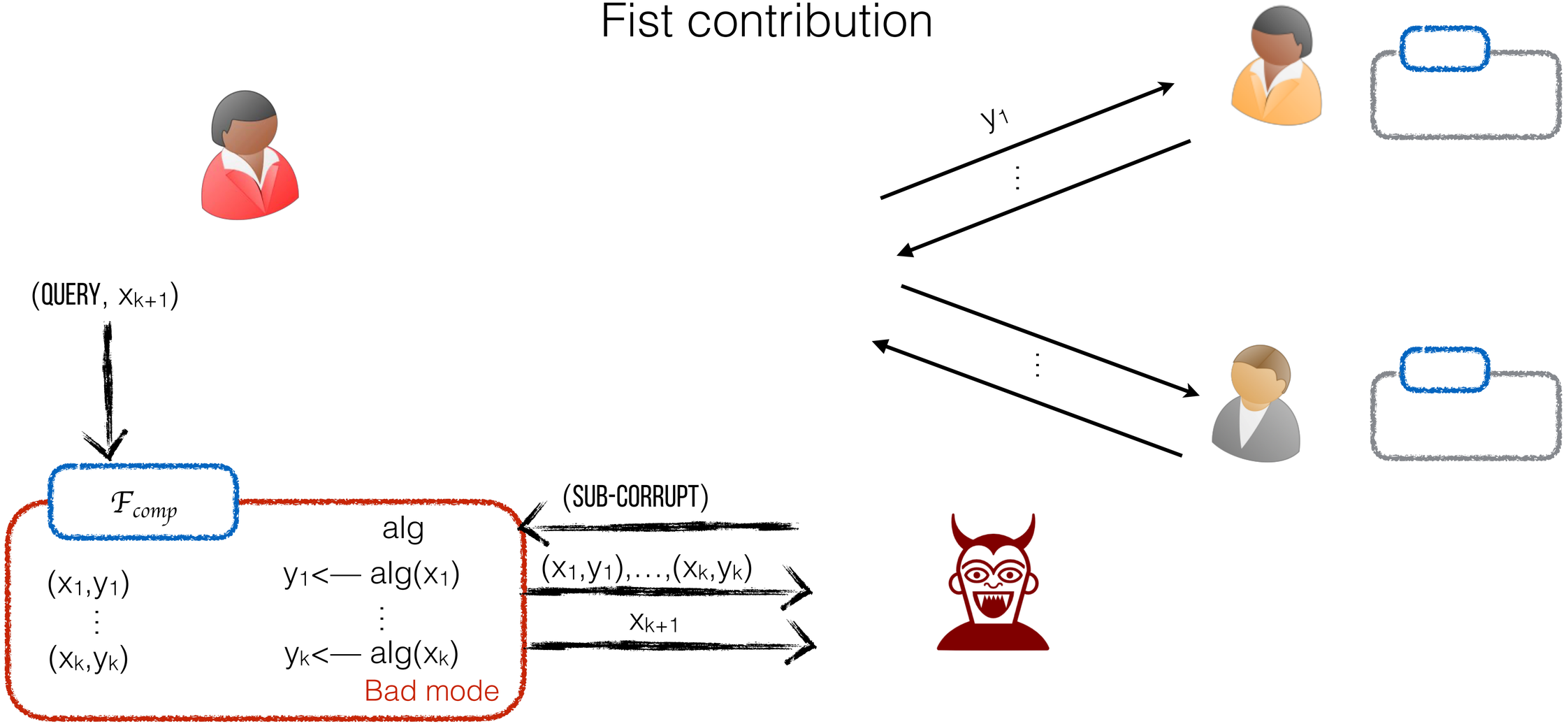
Modelling broken things

Fist contribution



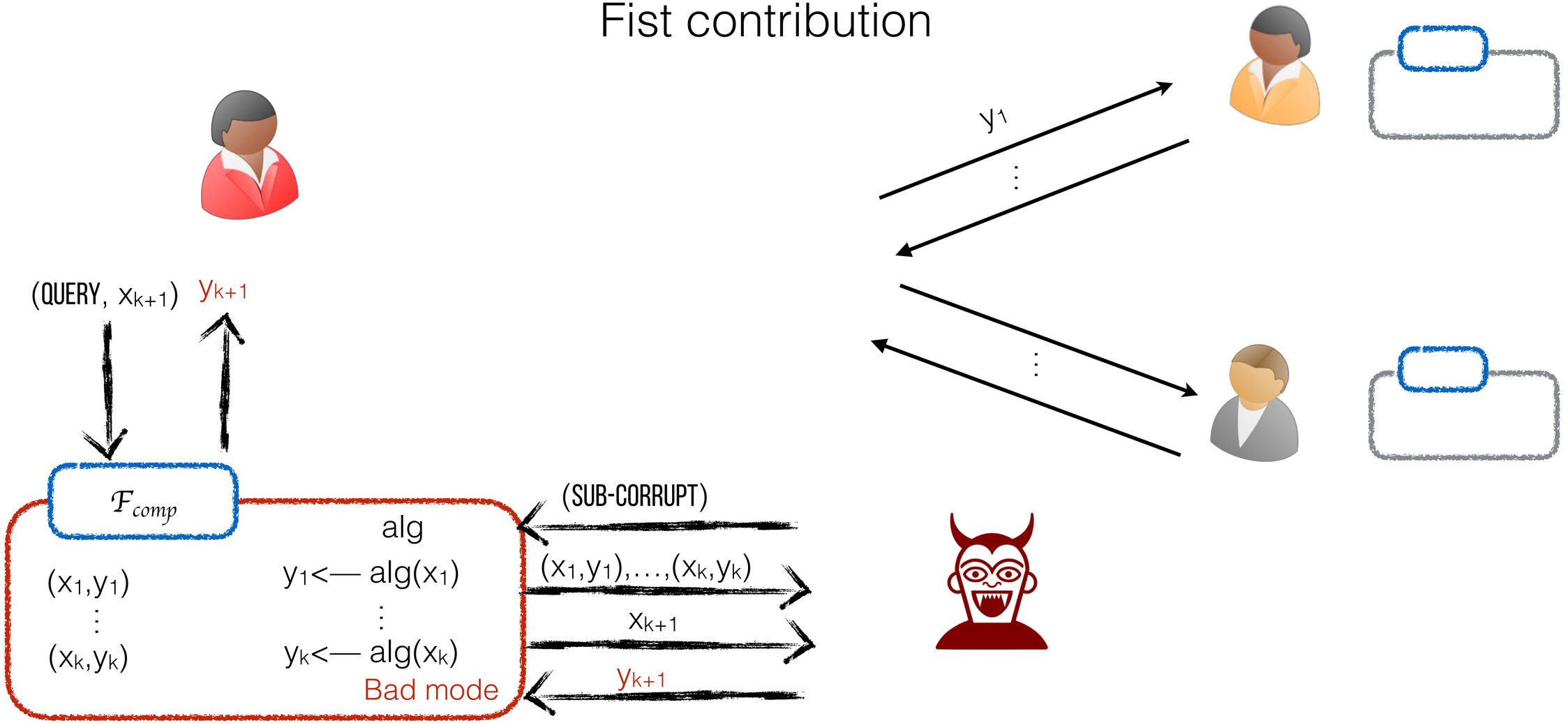
Modelling broken things

Fist contribution



Modelling broken things

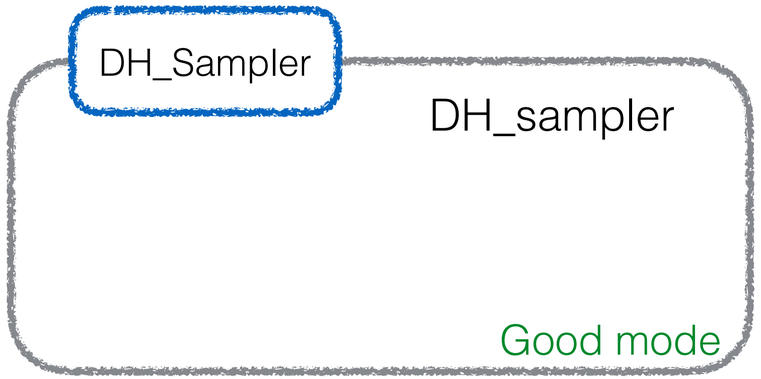
Fist contribution

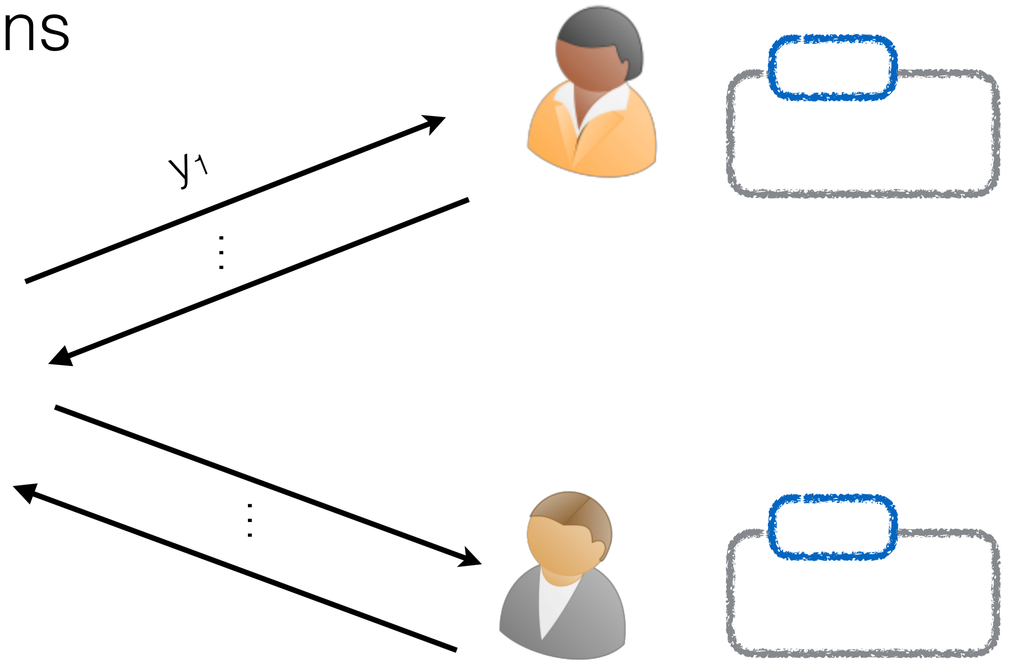


Modelling broken things

Assumptions

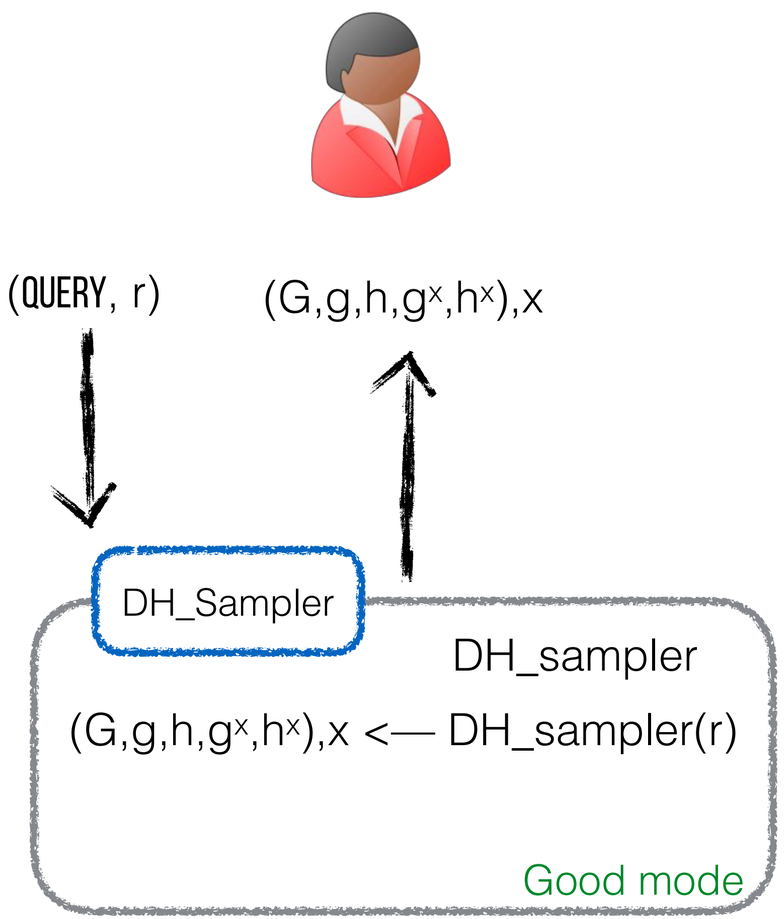
(INIT, DH_sampler)

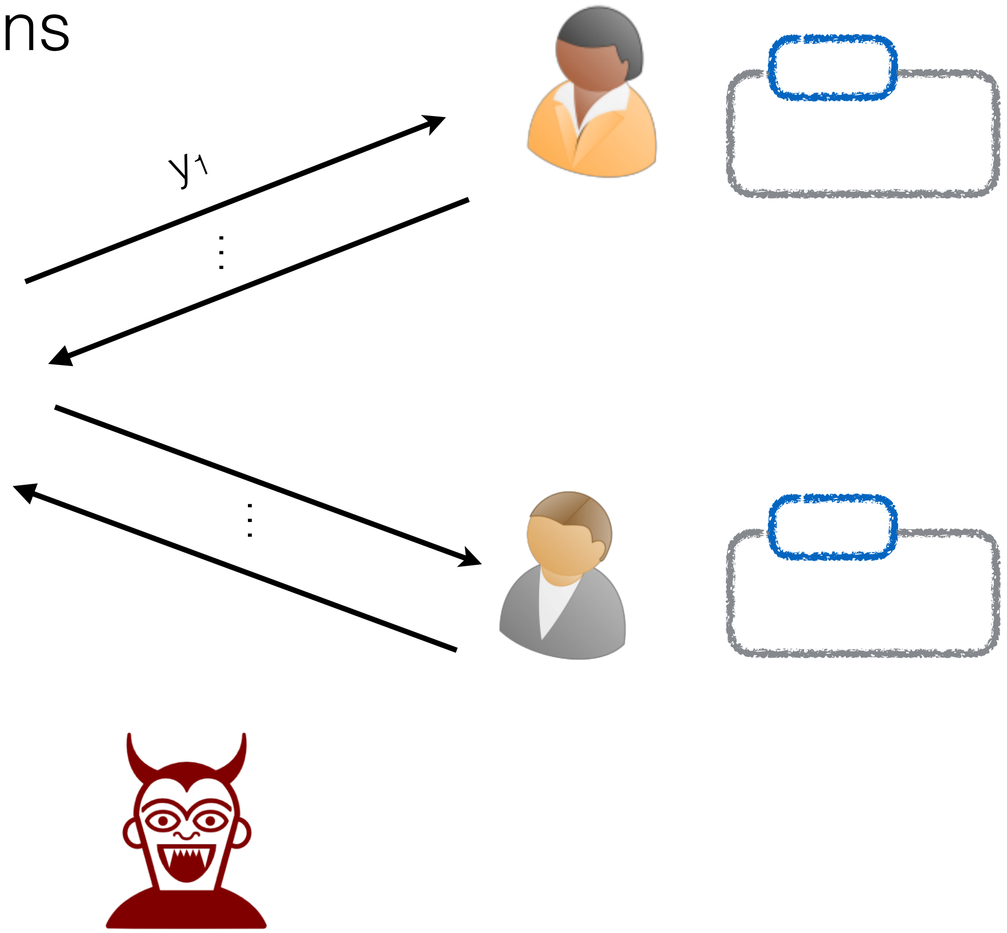




Modelling broken things

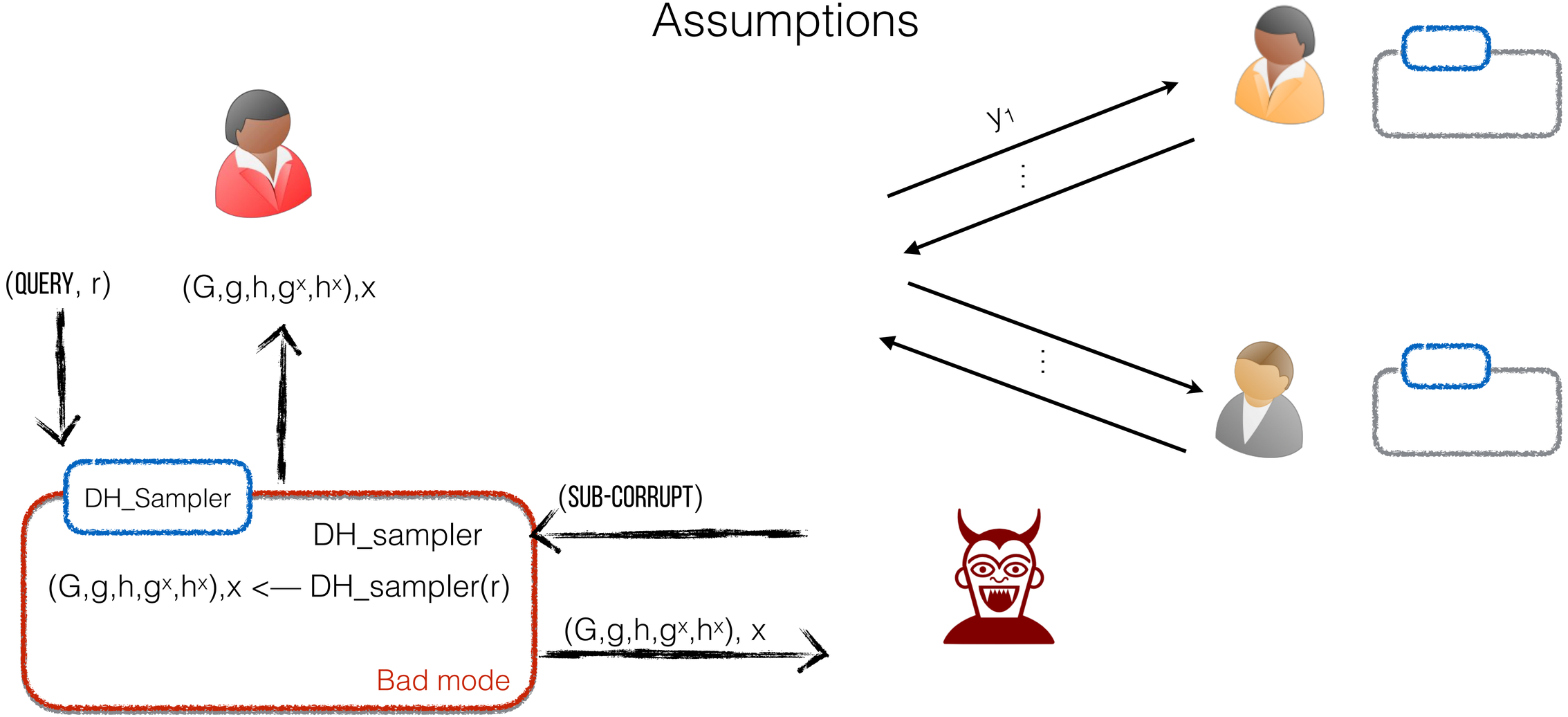
Assumptions



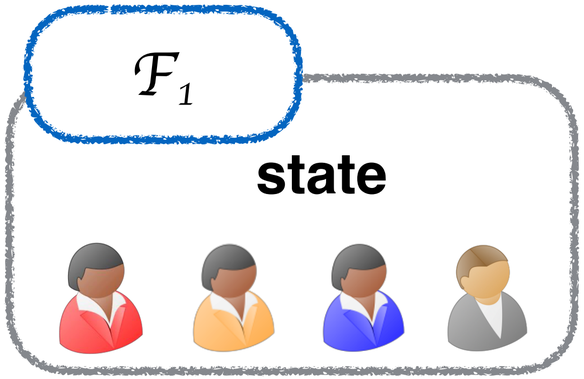


Modelling broken things

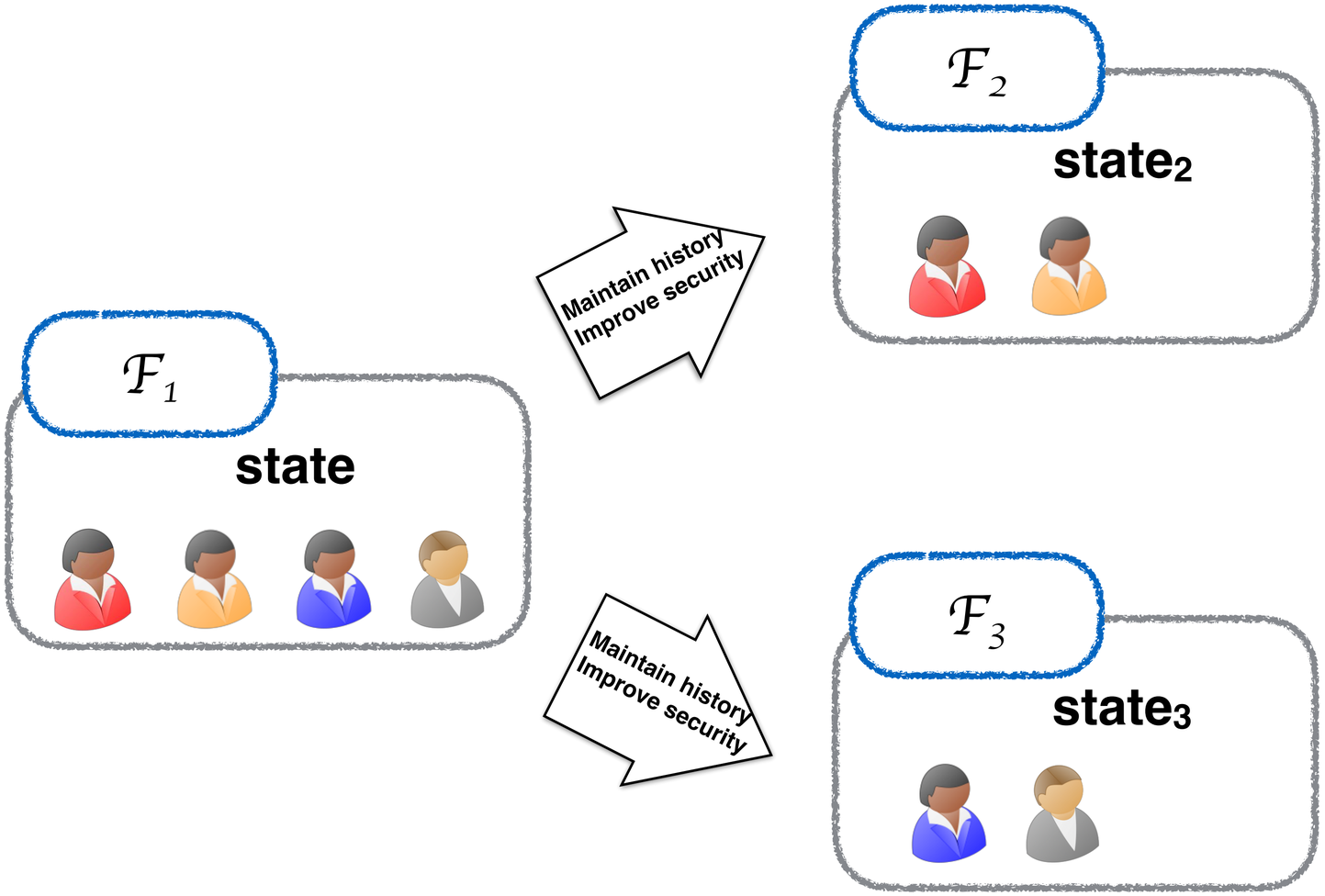
Assumptions



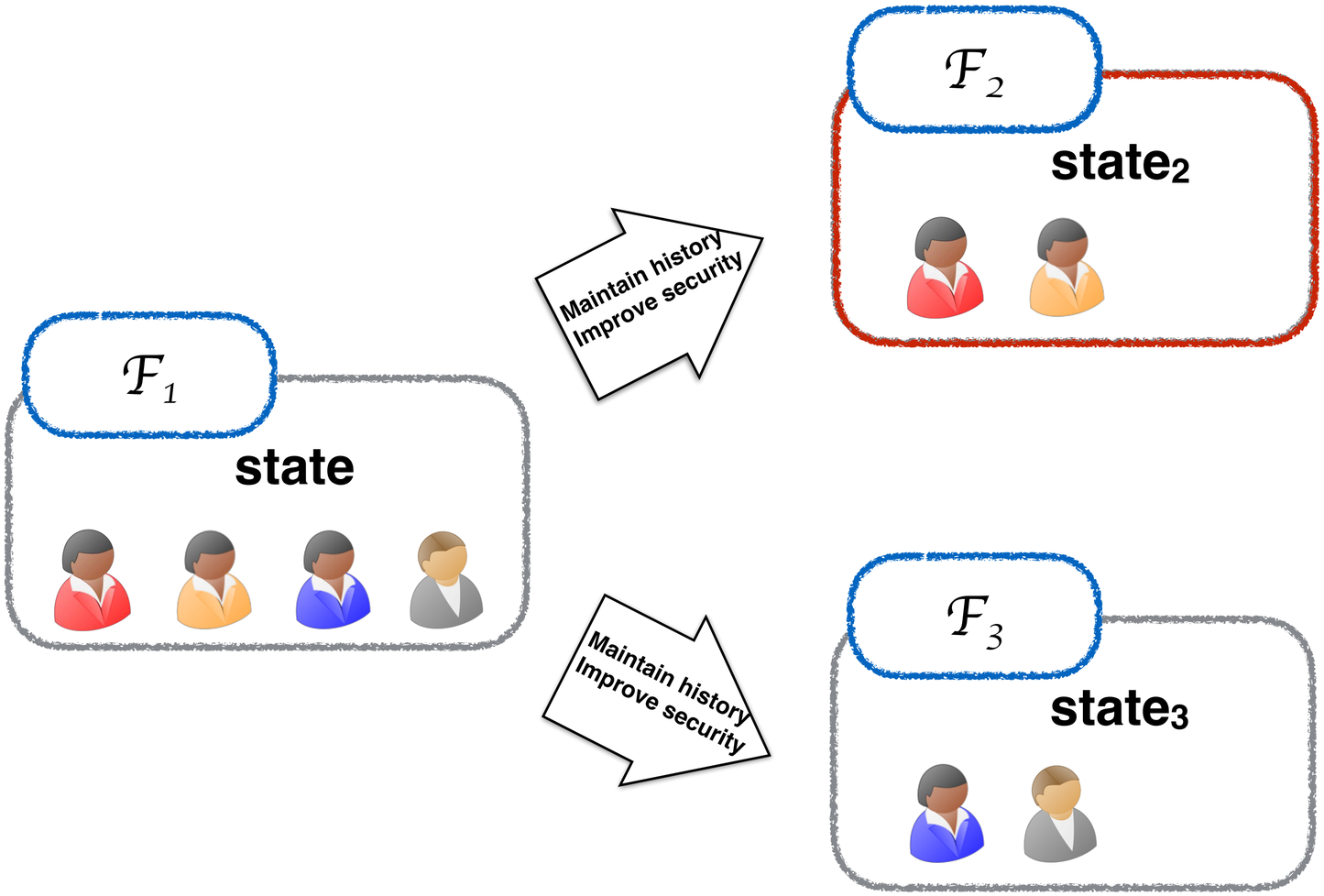
Cryptographic agility



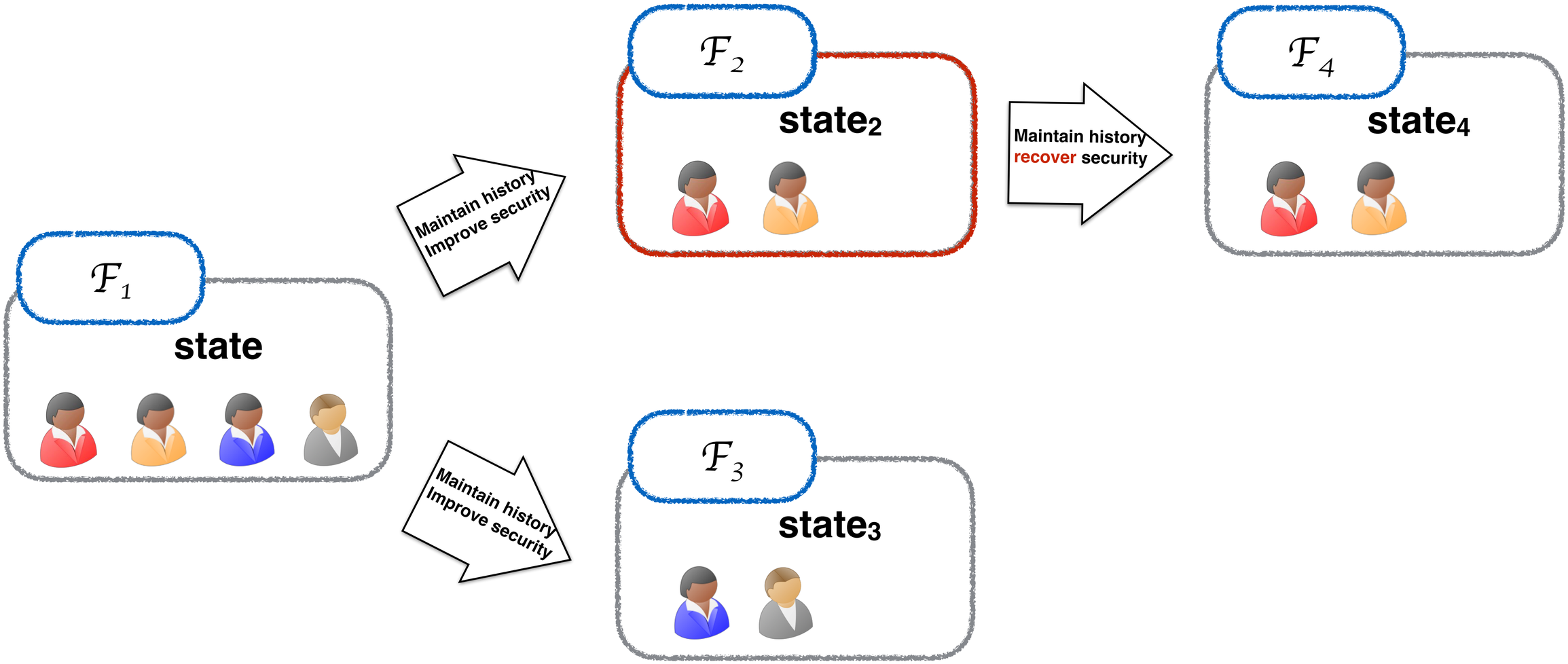
Cryptographic agility



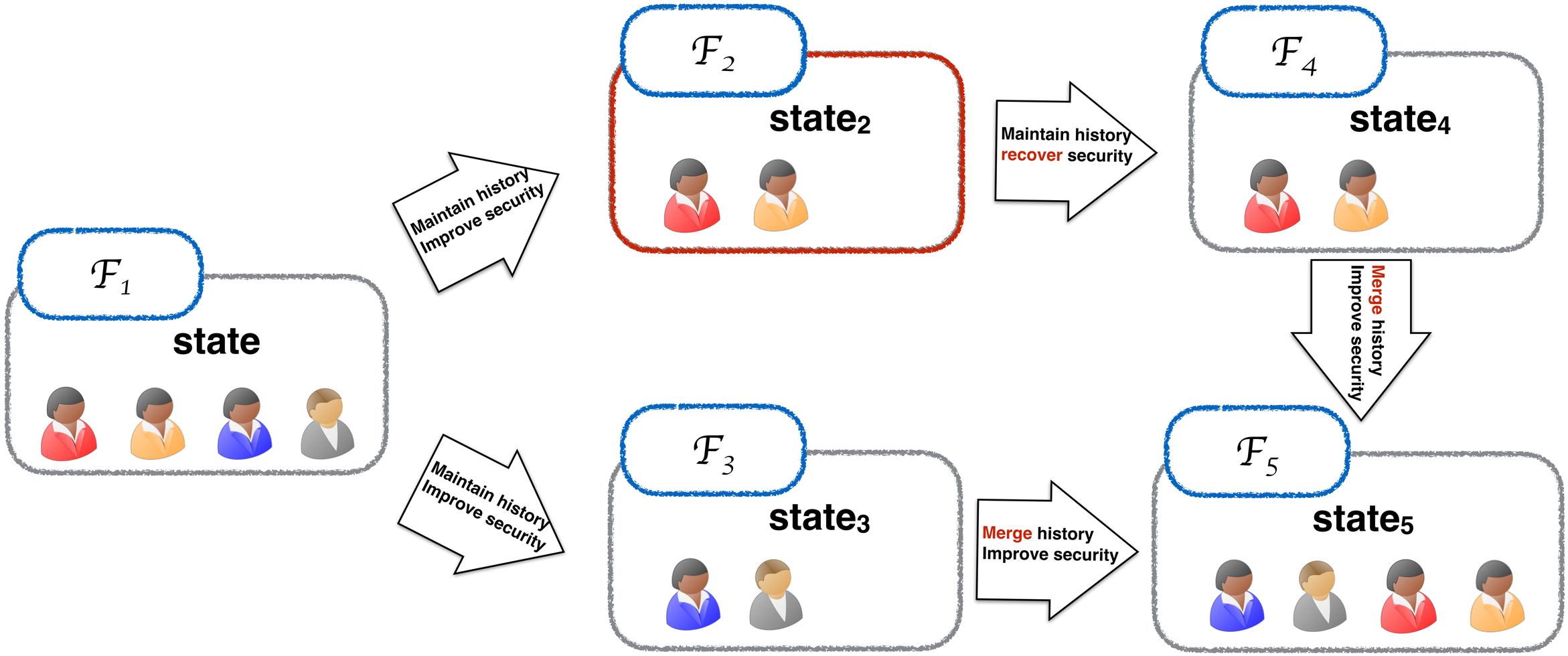
Cryptographic agility



Cryptographic agility



Cryptographic agility



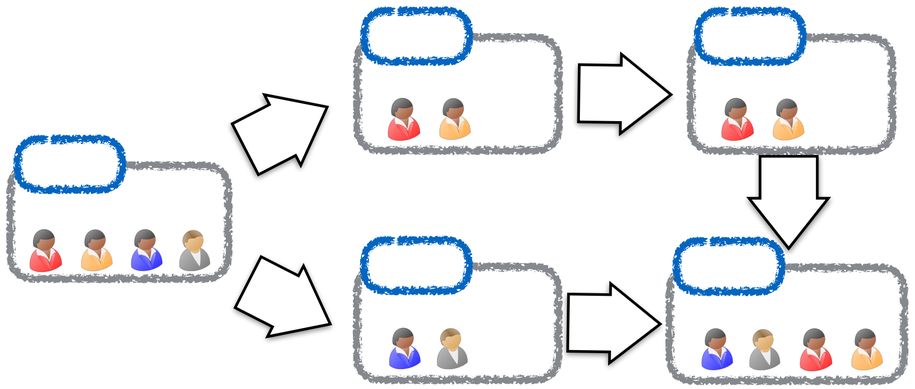
Cryptographic agility

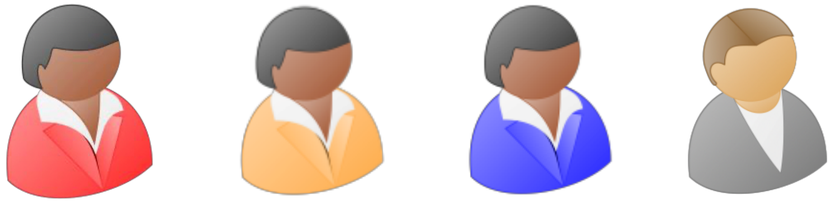
Second contribution

$\mathcal{F}_{update}$

Parameters

- Class of supported functionalities $C_{\mathcal{F}}$
- UpdatePredicate()
- StateUpdate()





Cryptographic agility

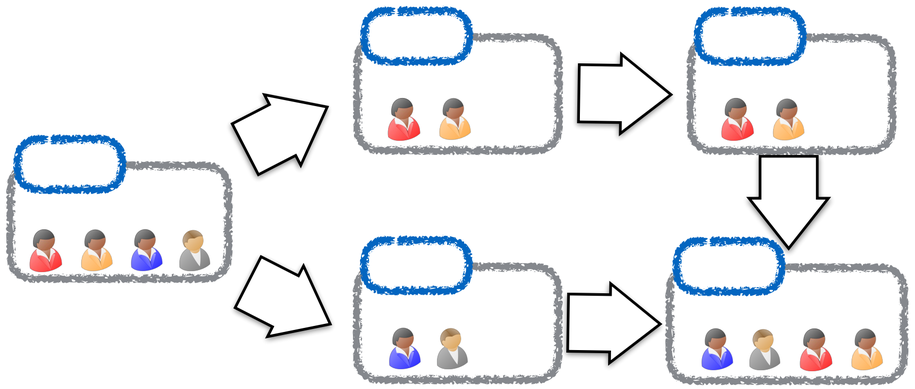
Second contribution

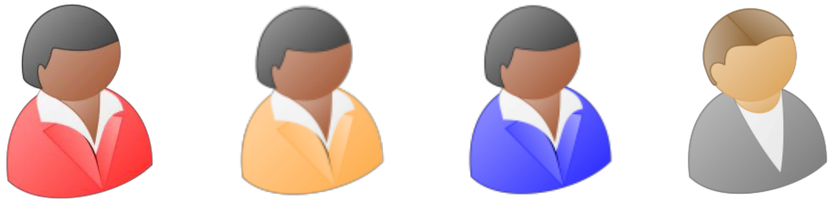
$\mathcal{F}_{update}$

Parameters

- Class of supported functionalities $C_{\mathcal{F}}$
- UpdatePredicate()
- StateUpdate()

e.g., functionalities for which such that **state** $\in \mathcal{L}$ or which have the same interface





Cryptographic agility

Second contribution

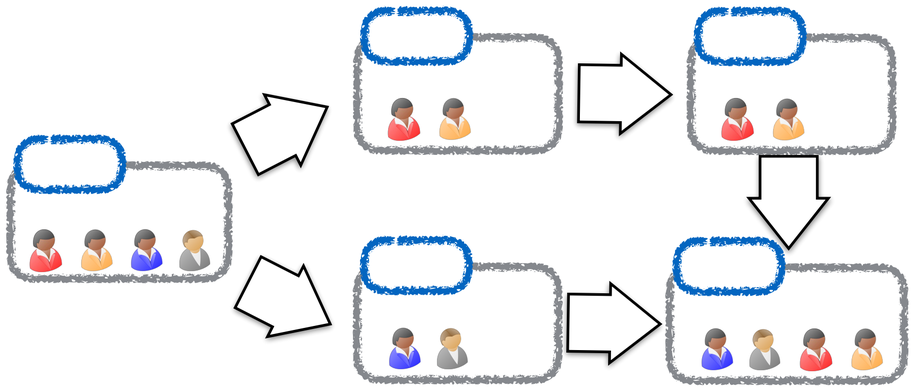
$\mathcal{F}_{update}$

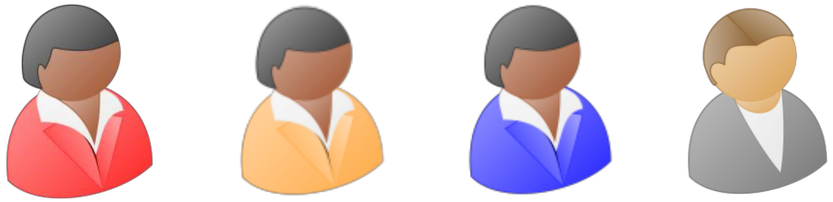
Parameters

- Class of supported functionalities $C_{\mathcal{F}}$
- UpdatePredicate()
- StateUpdate()

e.g., functionalities for which such that **state** $\in \mathcal{L}$ or which have the same interface

e.g., update if the majority of the parties want to update, or update if a leader-party wants to update





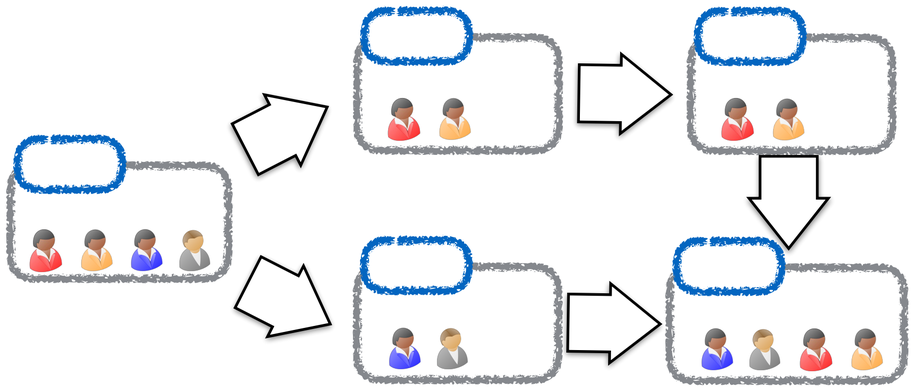
Cryptographic agility

Second contribution

$\mathcal{F}_{update}$

Parameters

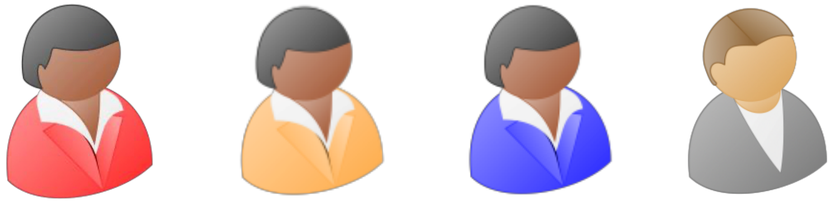
- Class of supported functionalities $C_{\mathcal{F}}$
- UpdatePredicate()
- StateUpdate()



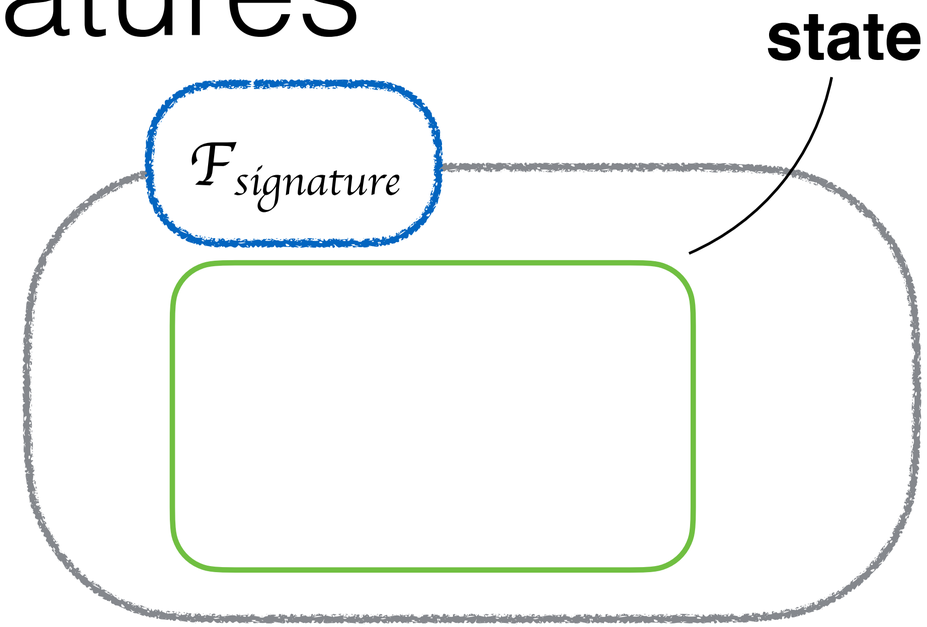
e.g., functionalities for which such that **state** $\in \mathcal{L}$ or which have the same interface

e.g., update if the majority of the parties want to update, or update if a leader-party wants to update

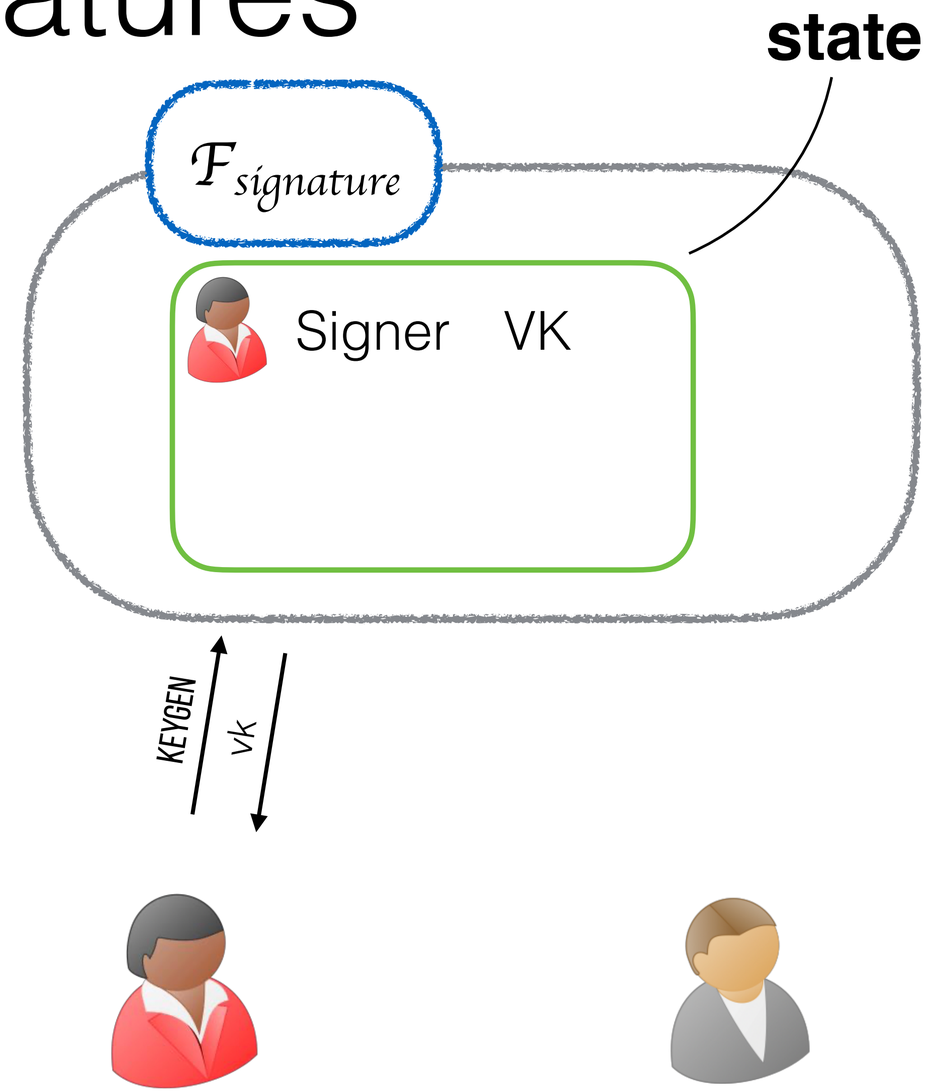
e.g., what part of the state should be preserved



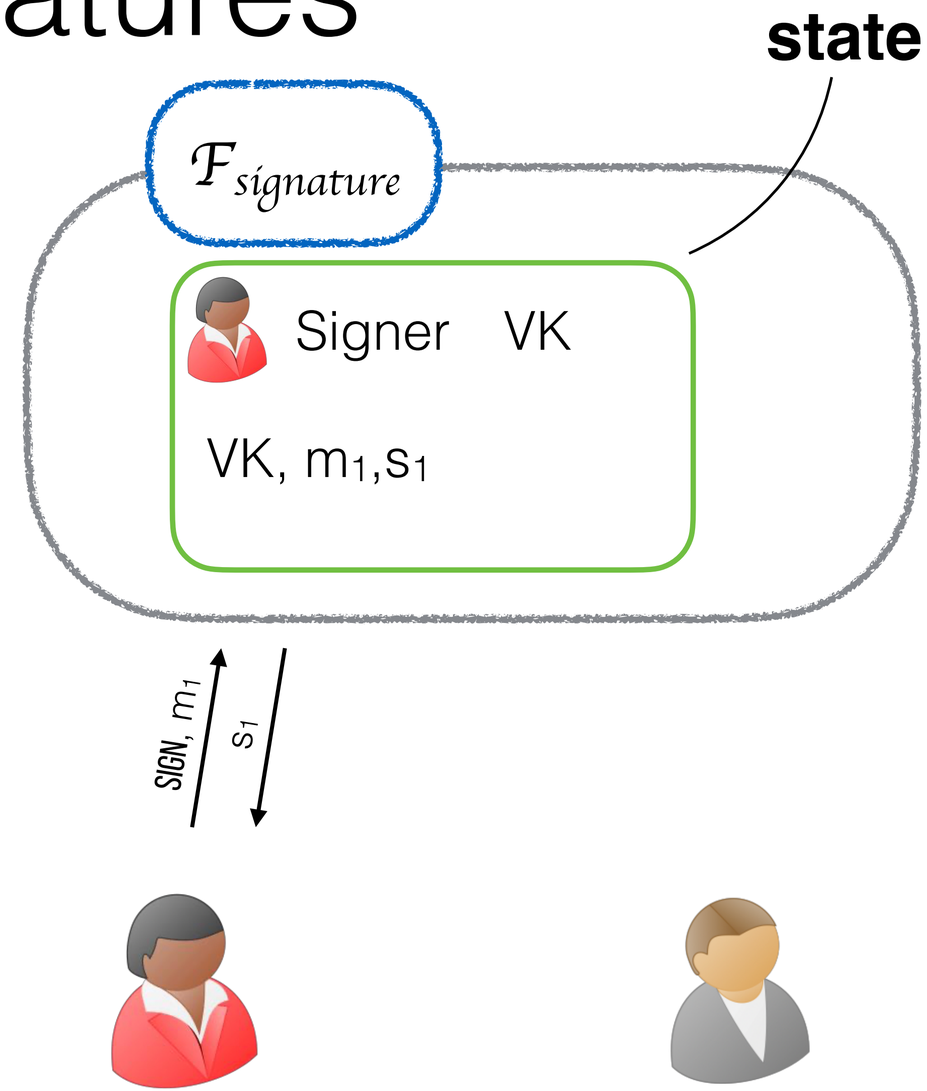
Updatable signatures



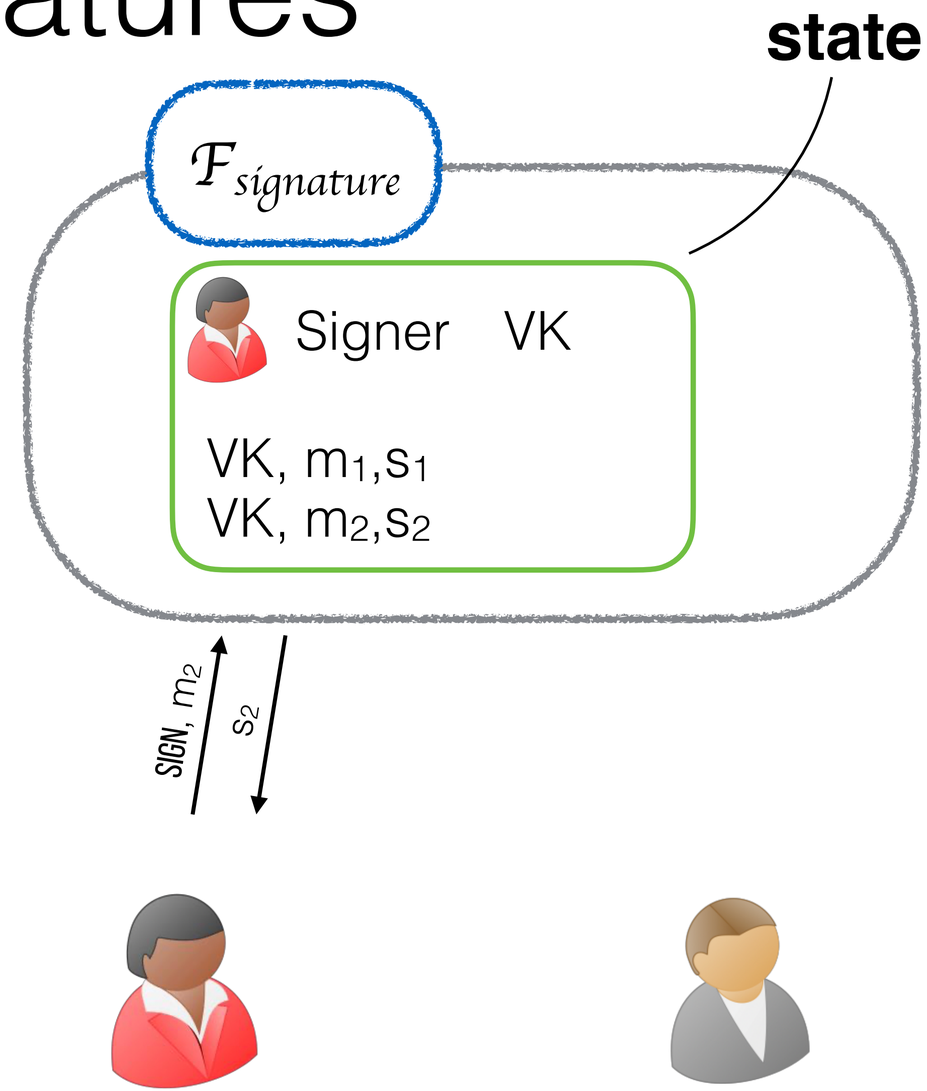
Updatable signatures



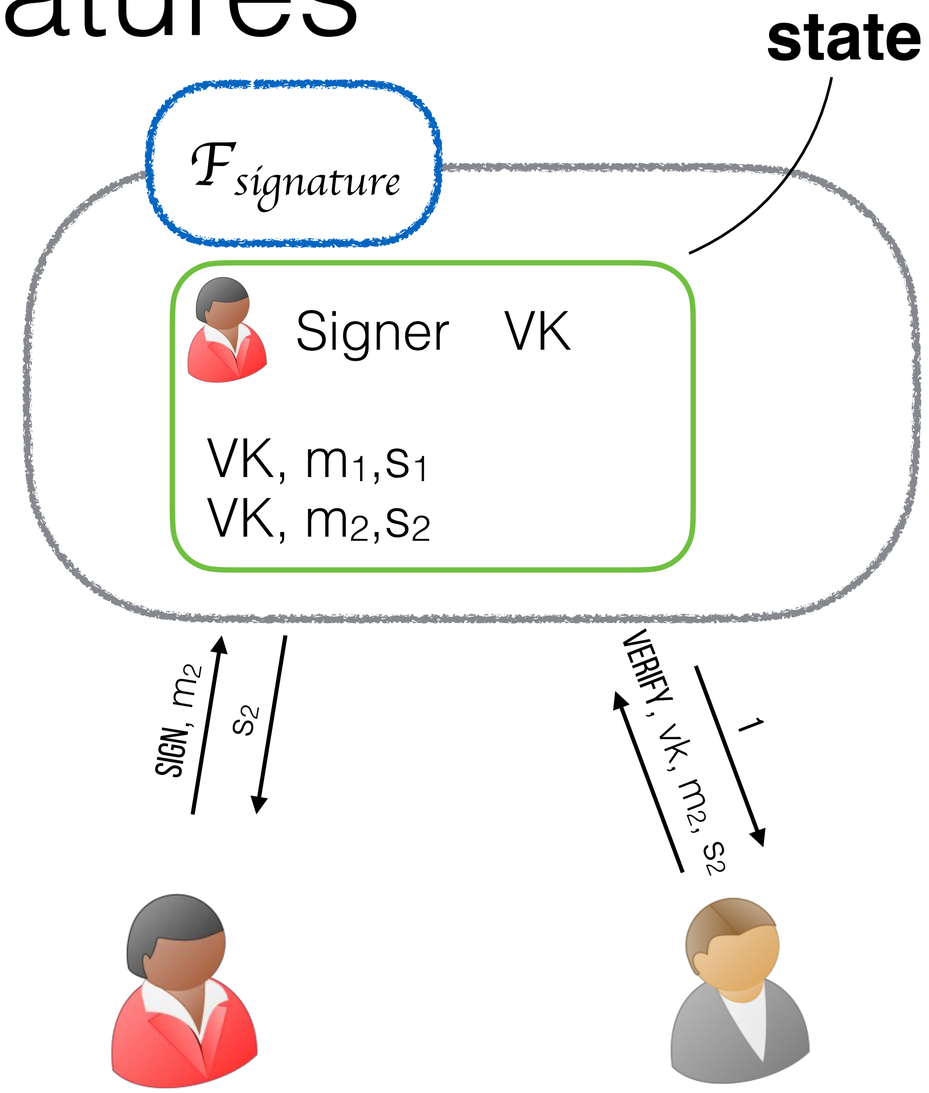
Updatable signatures



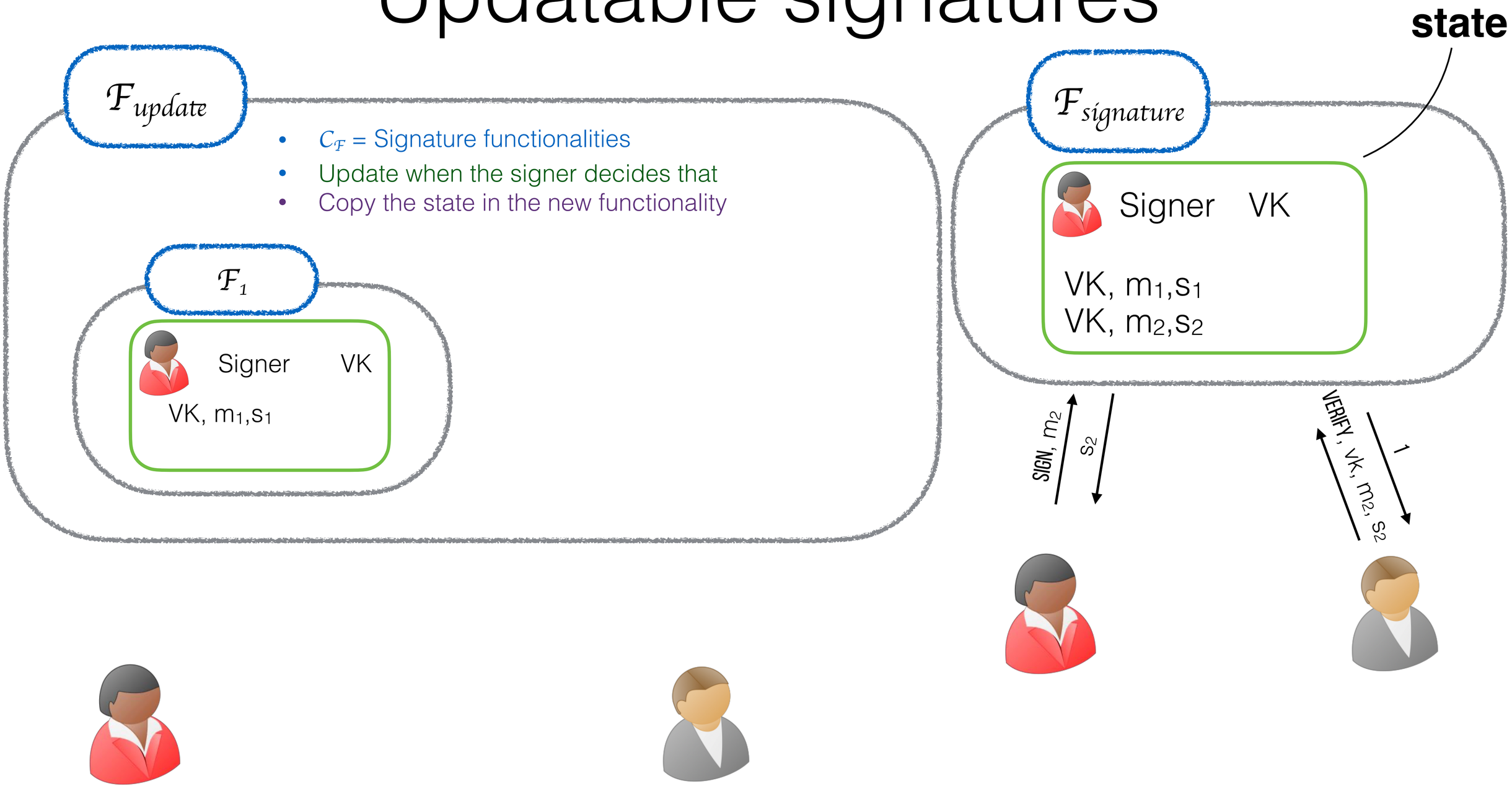
Updatable signatures



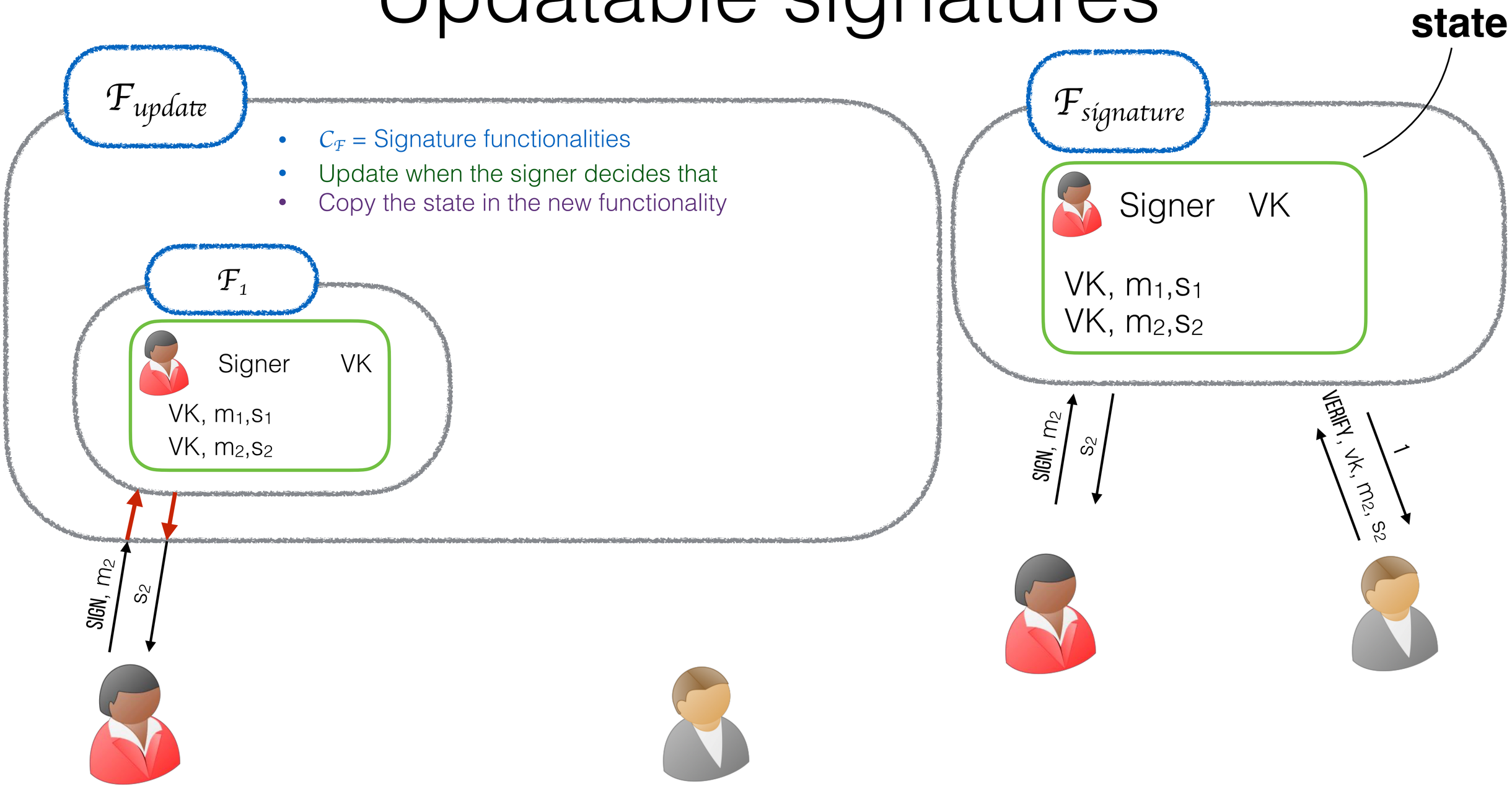
Updatable signatures



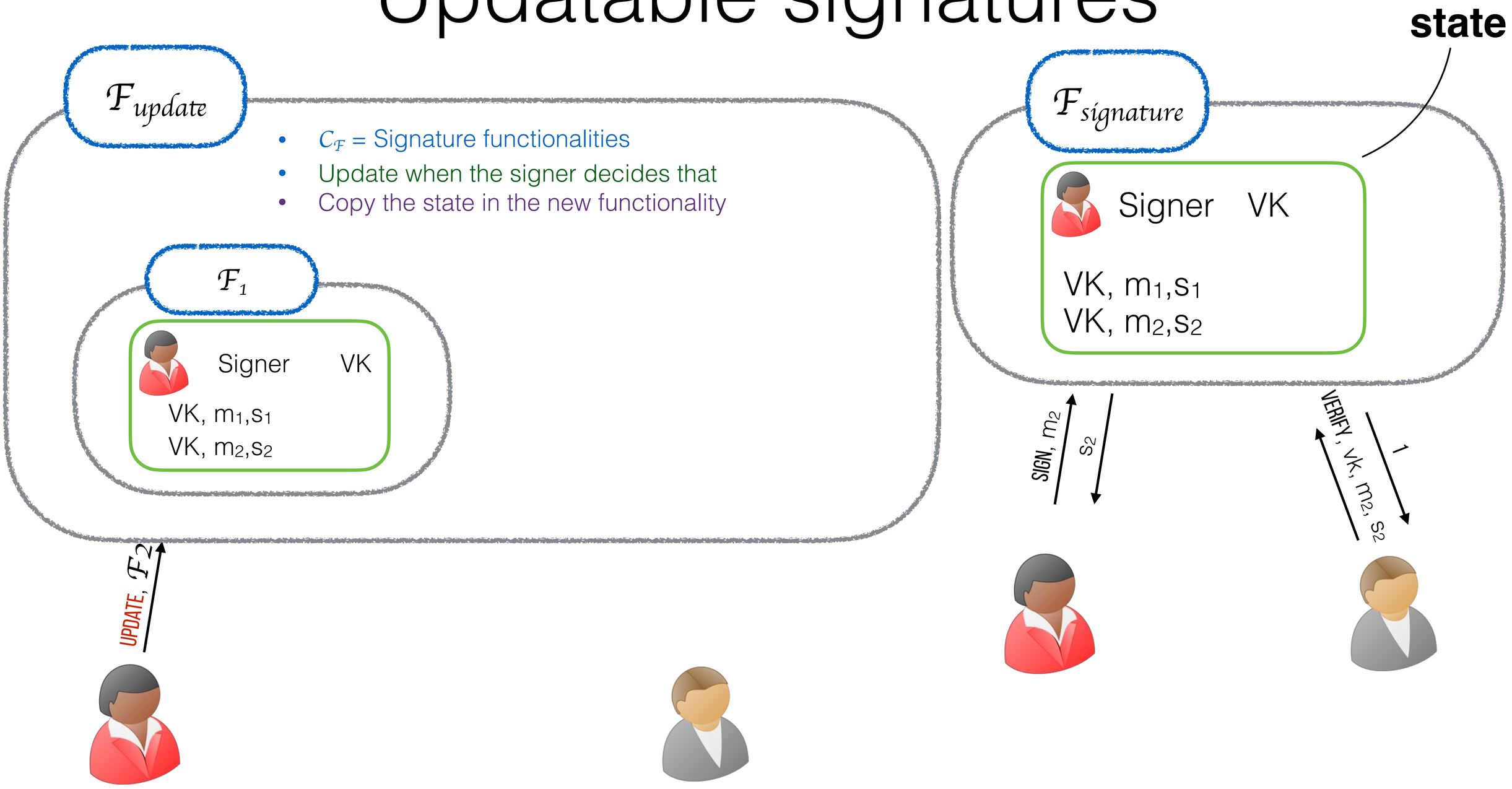
Updatable signatures



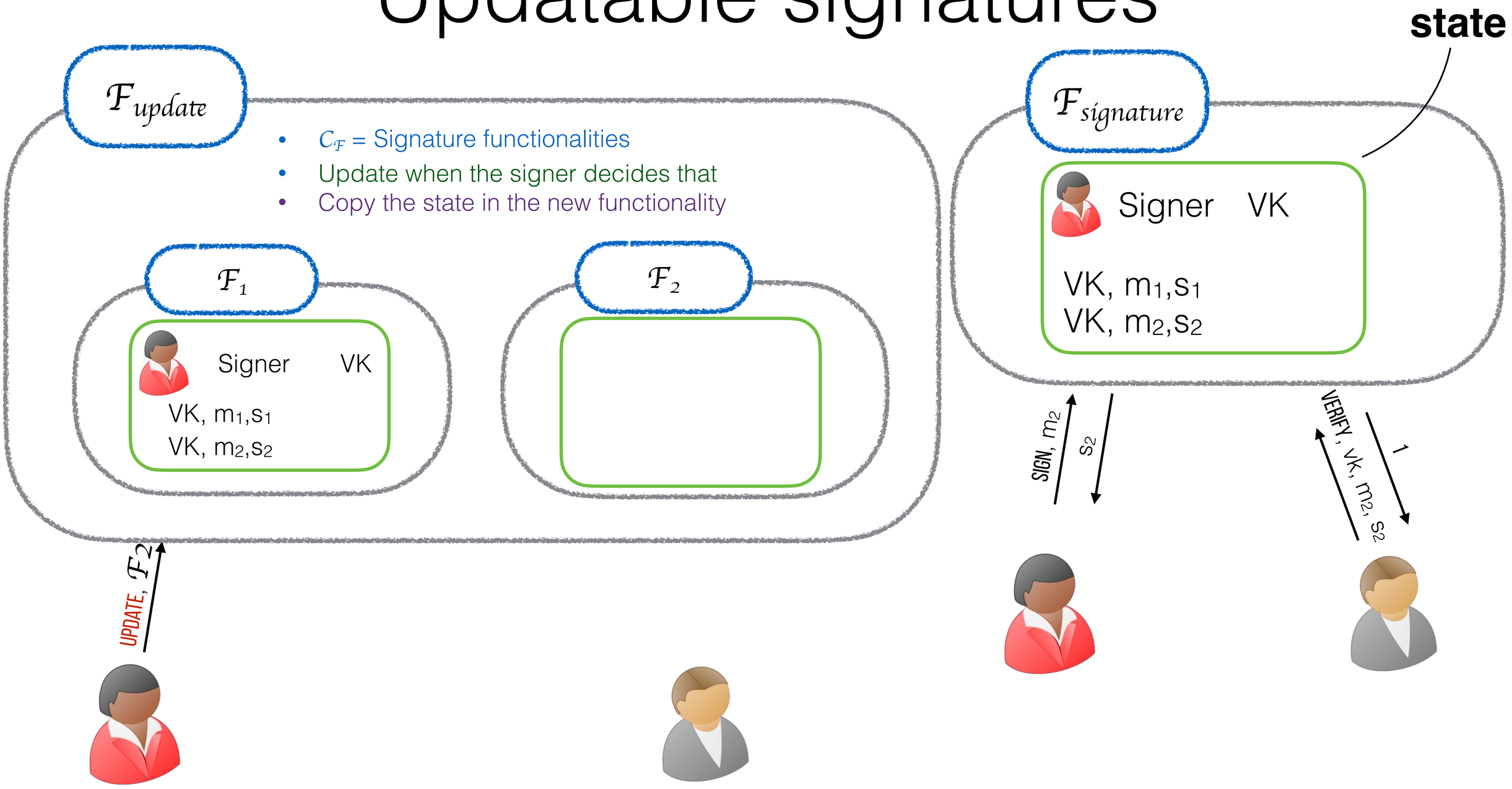
Updatable signatures



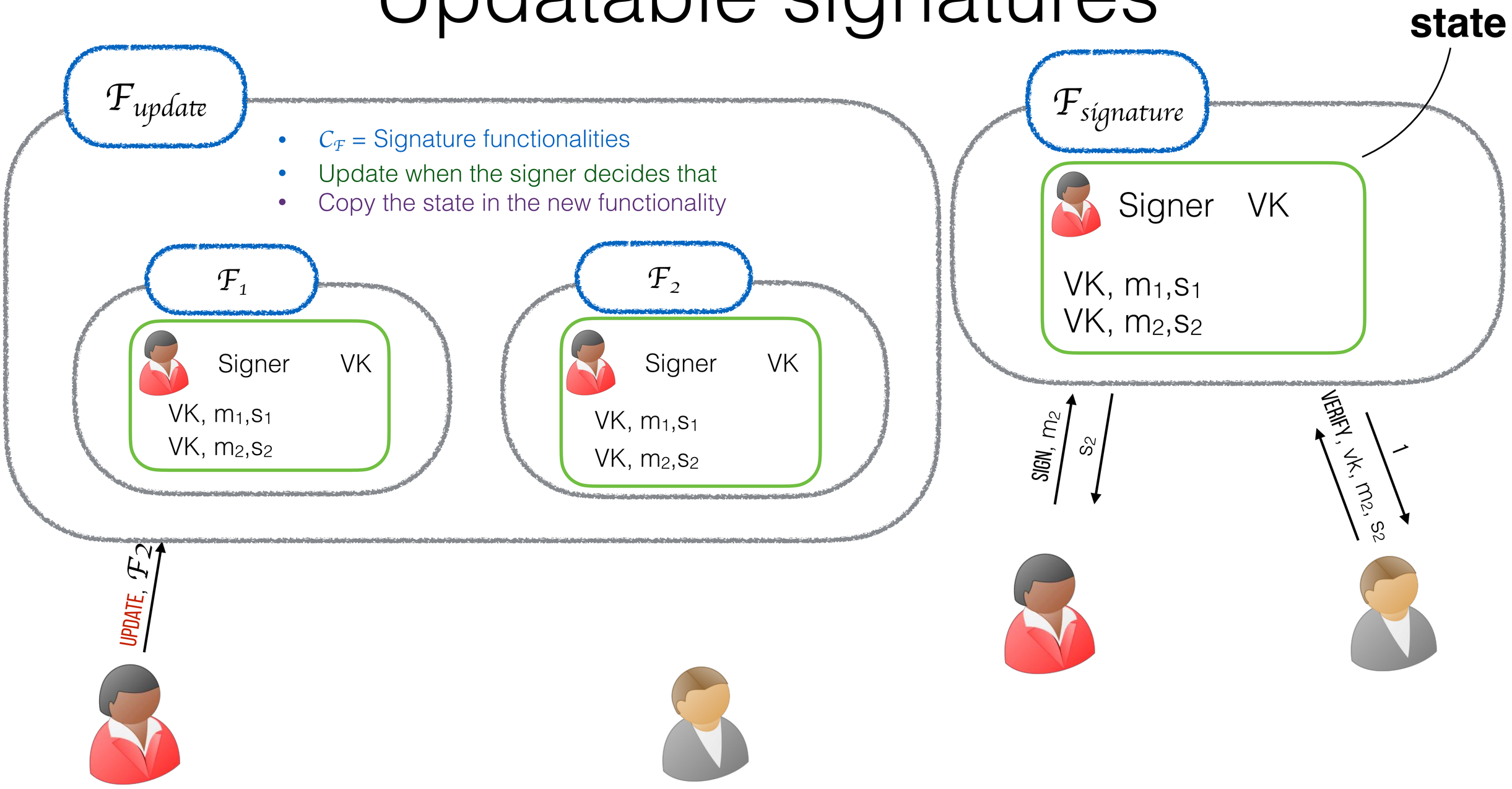
Updatable signatures



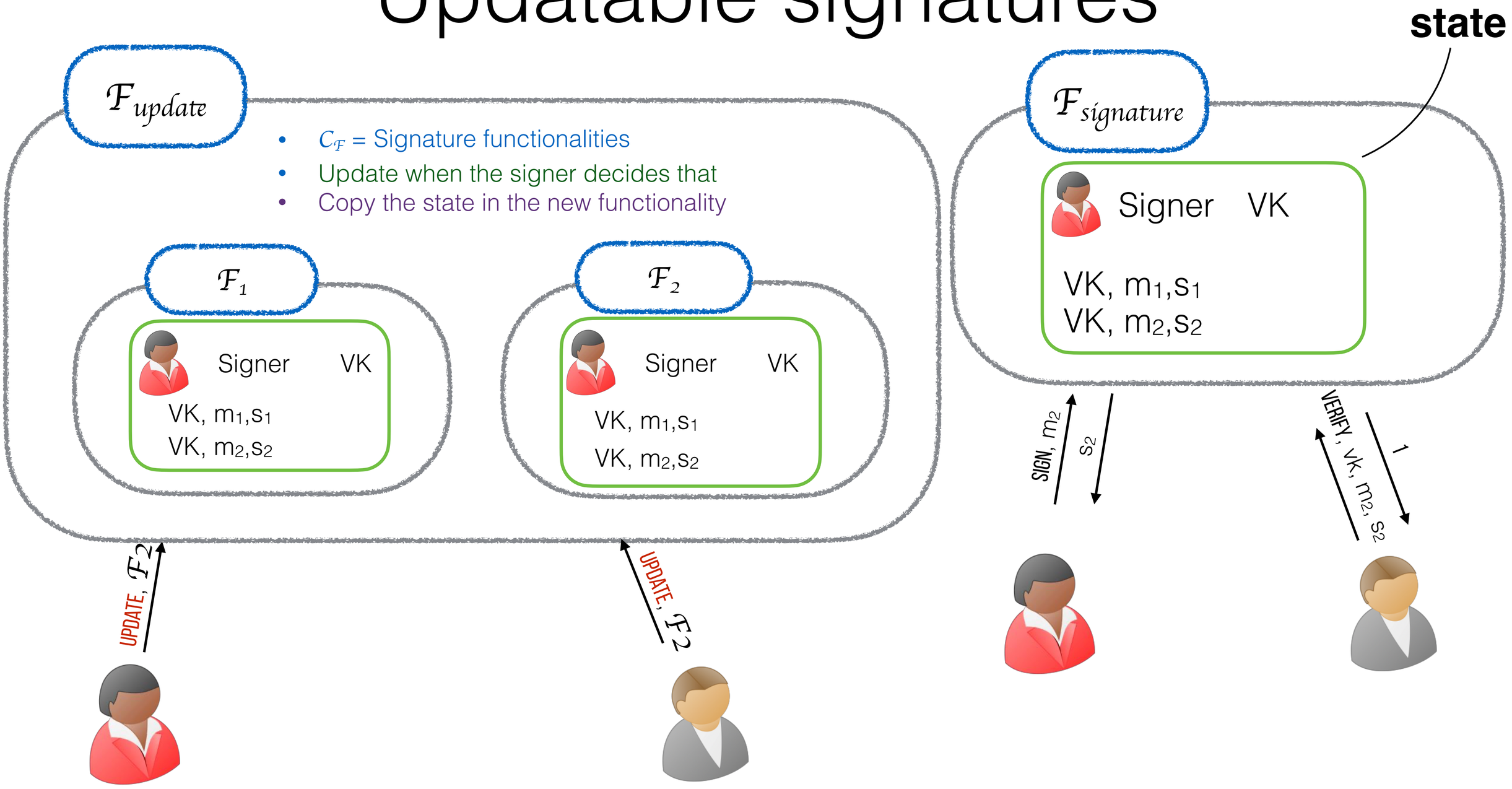
Updatable signatures



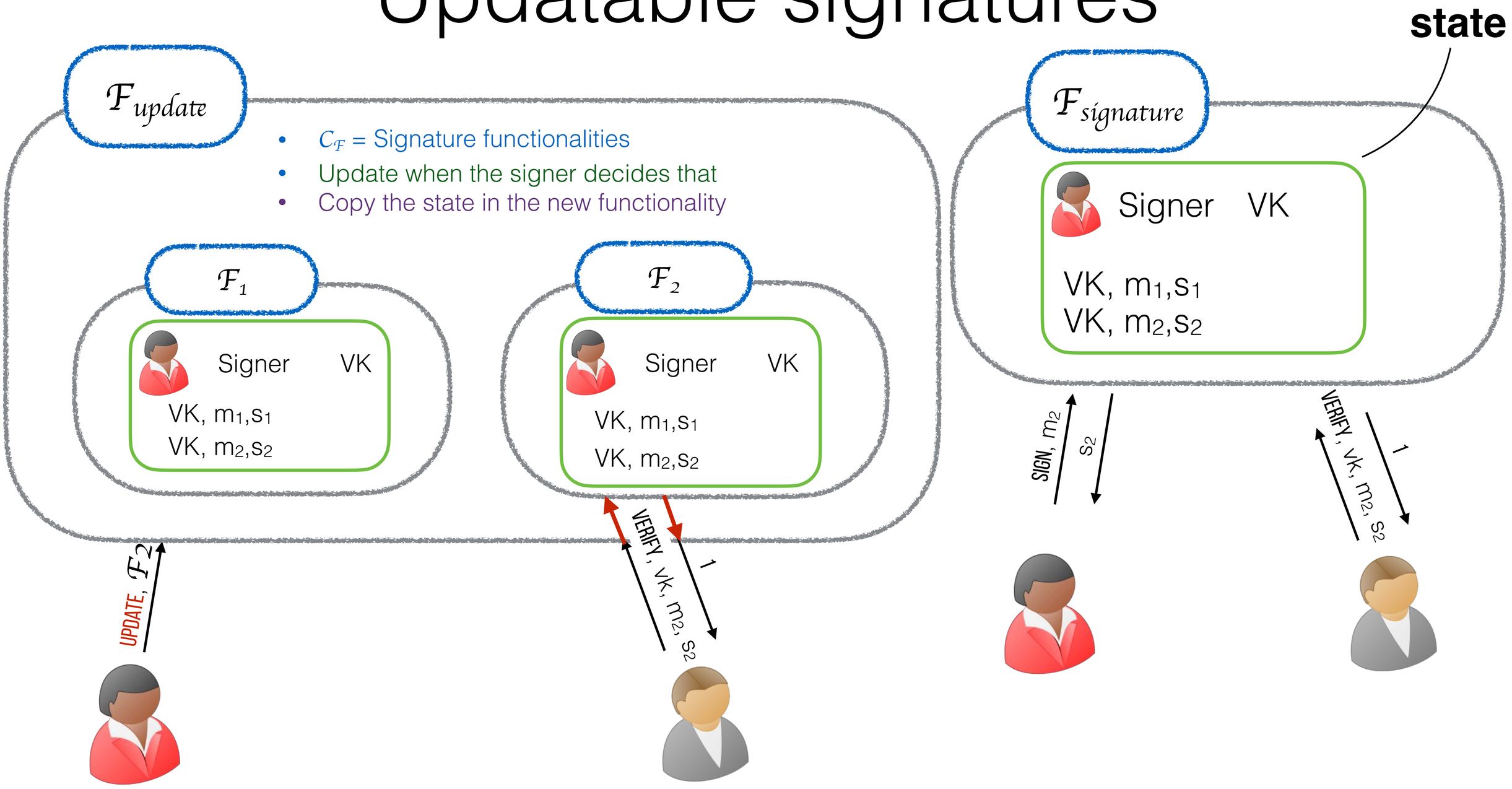
Updatable signatures



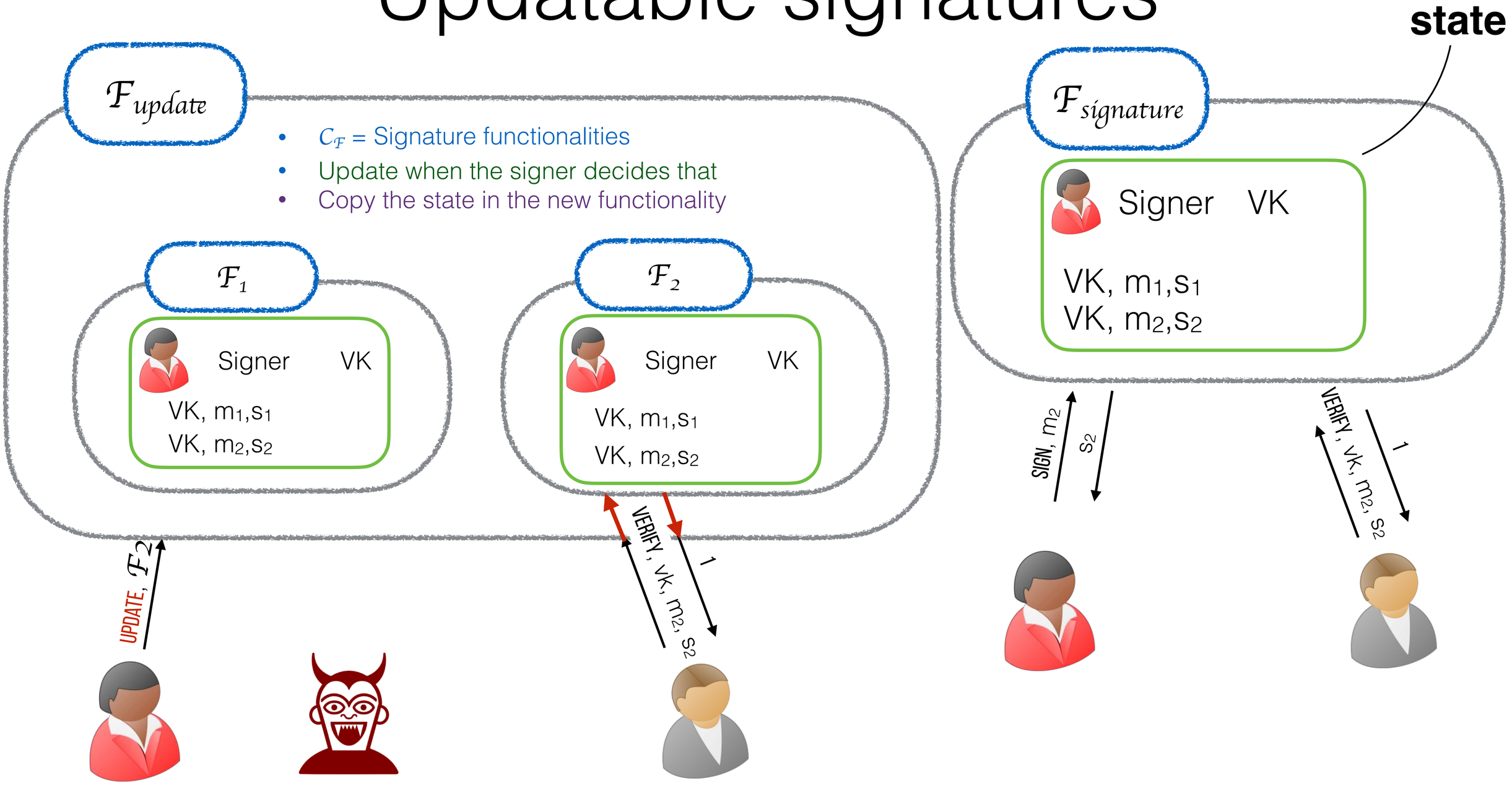
Updatable signatures



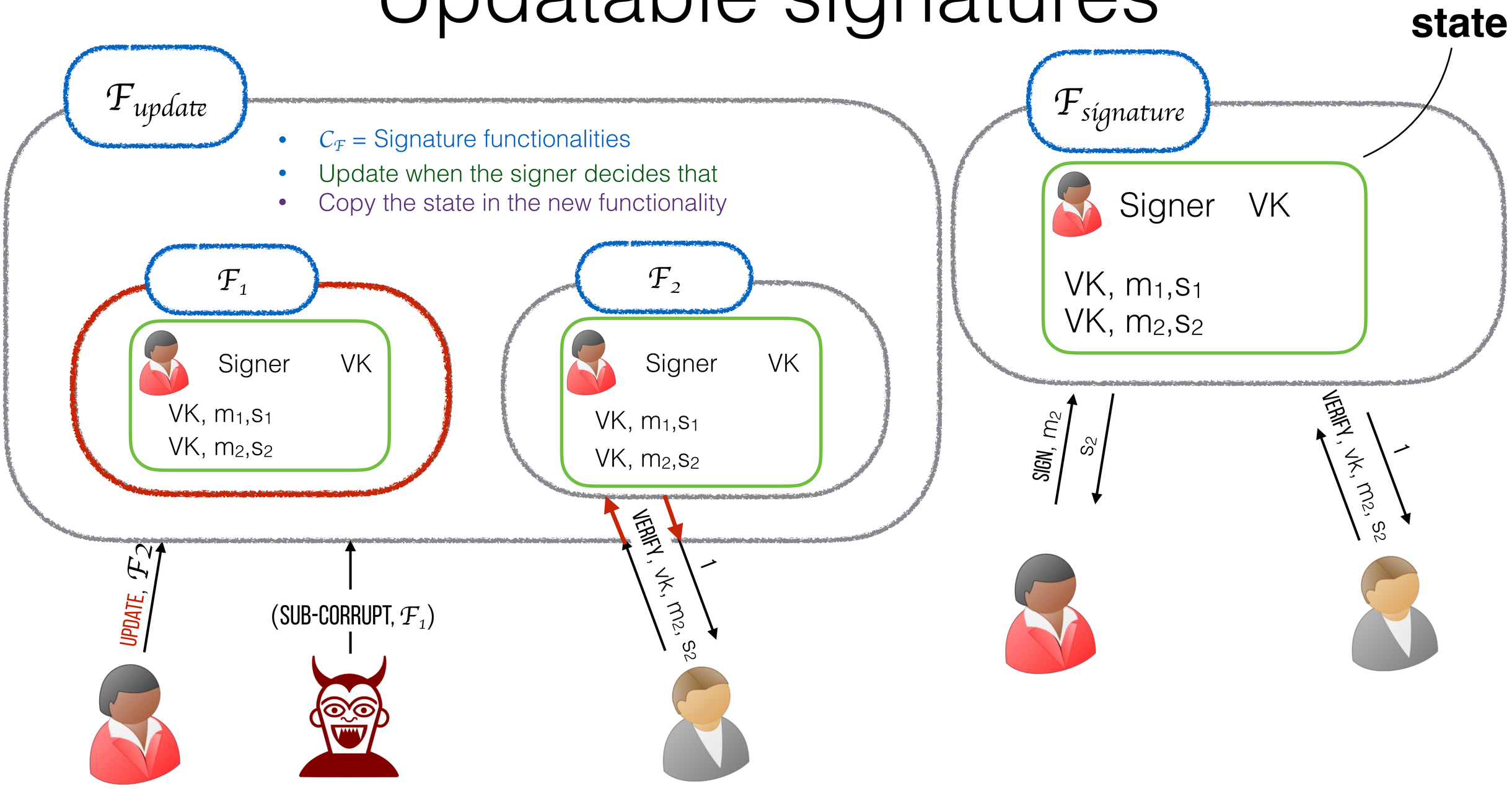
Updatable signatures



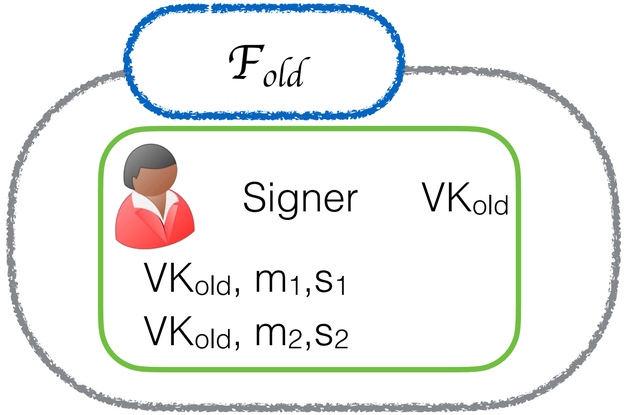
Updatable signatures



Updatable signatures



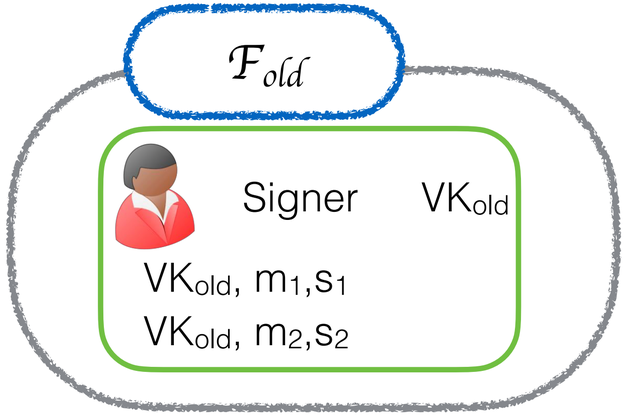
Updatable signatures



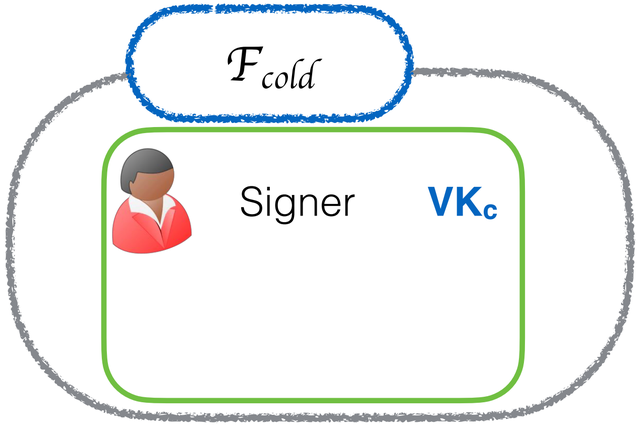
VK_C

VK_C

Updatable signatures

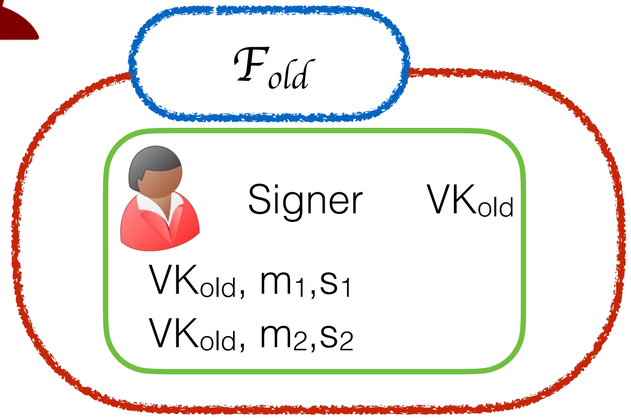


VK_C

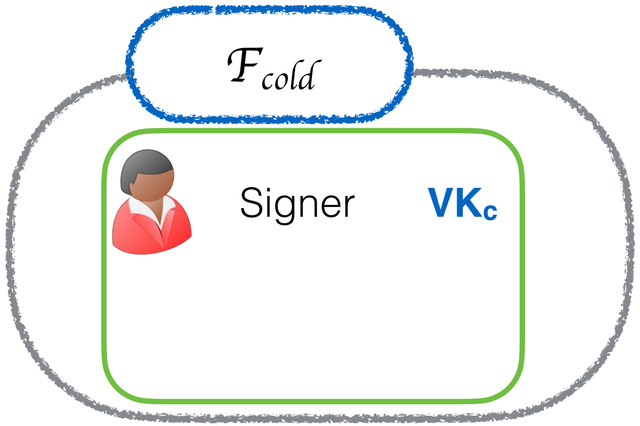


VK_C

Updatable signatures

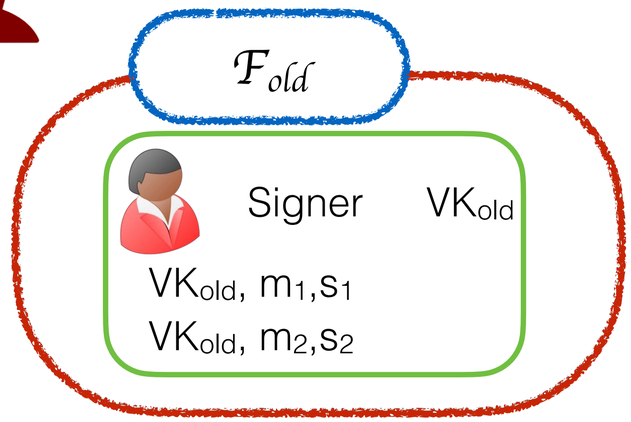


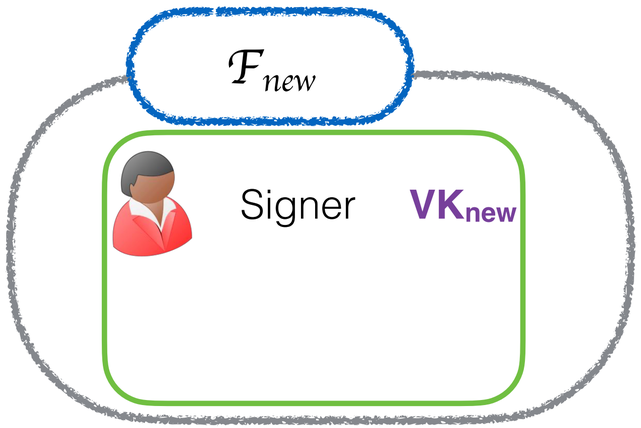
VK_C



VK_C

Updatable signatures

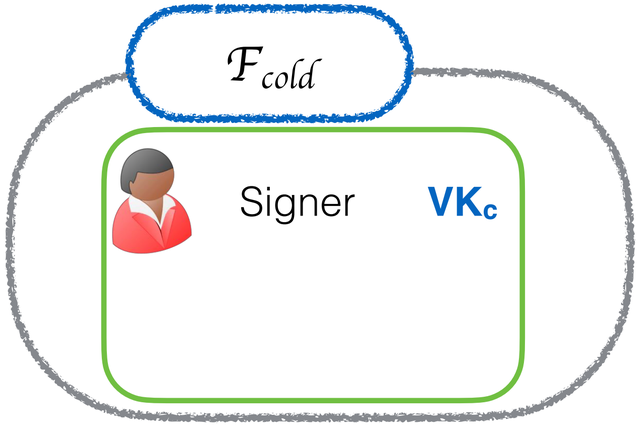




UPDATE, $\mathcal{F}_{new}$

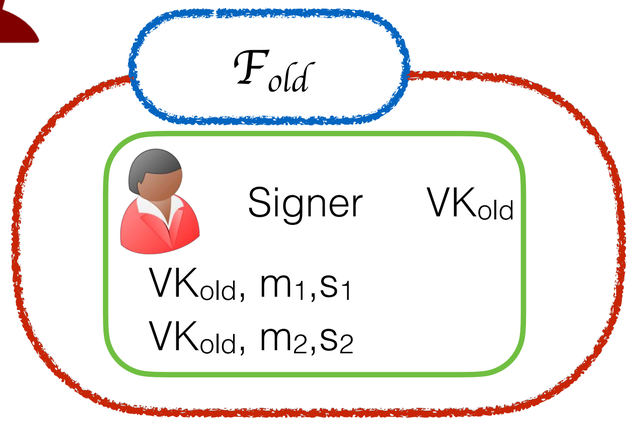
VK_c

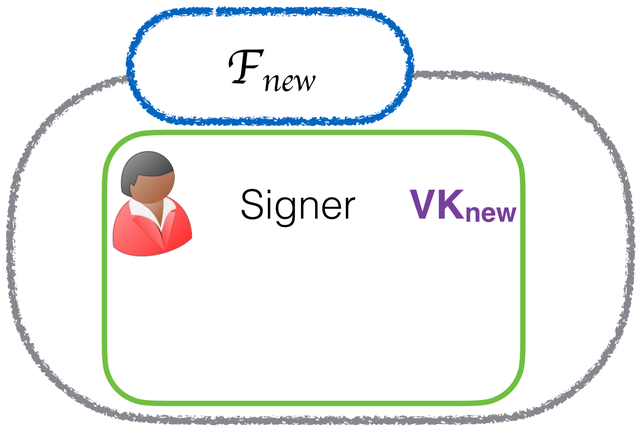
VK_{new}



VK_c

Updatable signatures



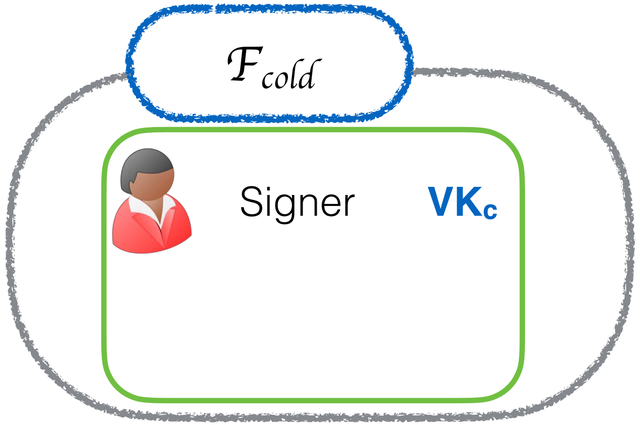


UPDATE, $\mathcal{F}_{new}$

VK_c

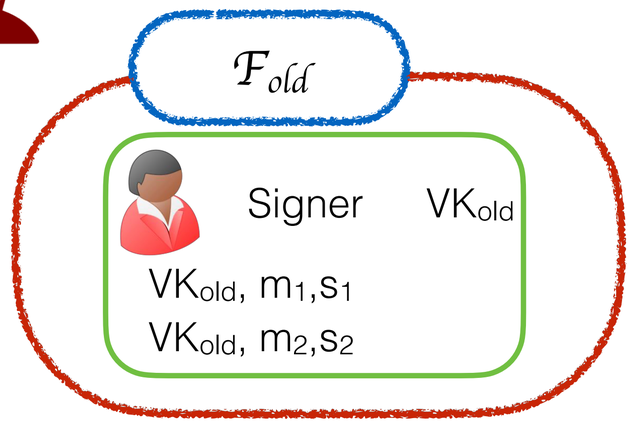
VK_{new}

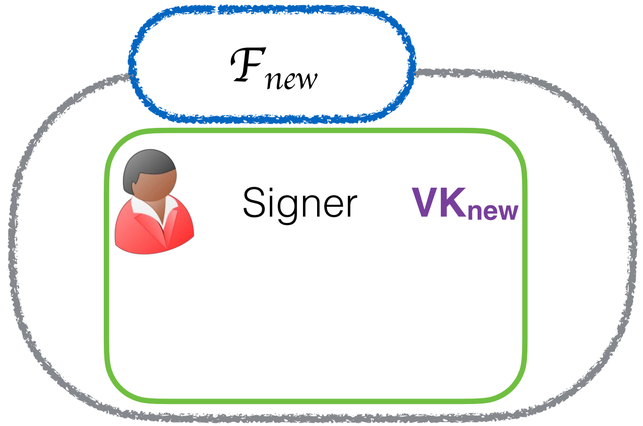
M<—{m_1,m_2}
h<—H(m_1||m_2)



VK_c

Updatable signatures



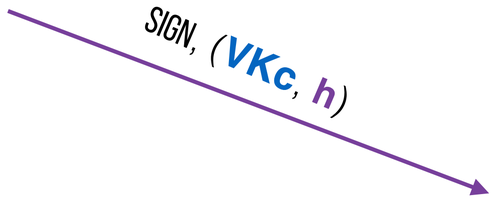


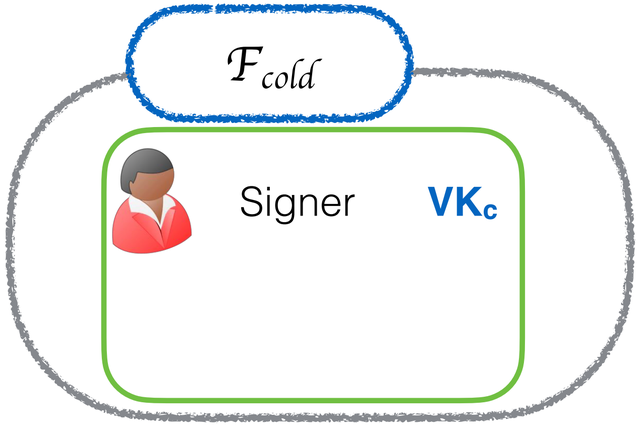
VK_c

VK_{new}

$M \leftarrow \{m_1, m_2\}$

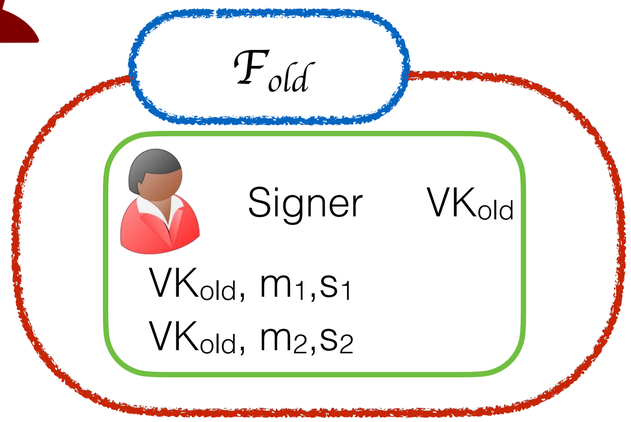
$h \leftarrow H(m_1 \| m_2)$

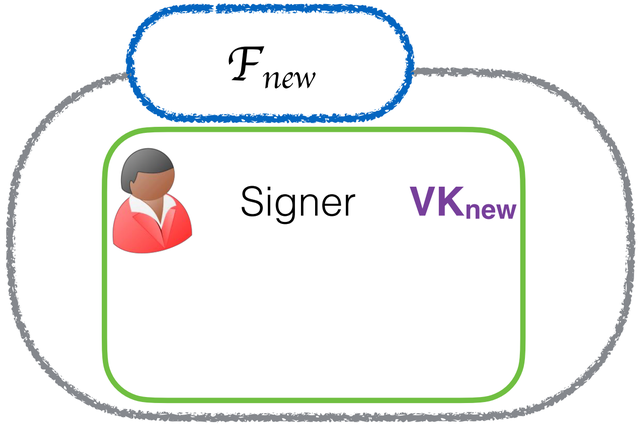




VK_c

Updatable signatures



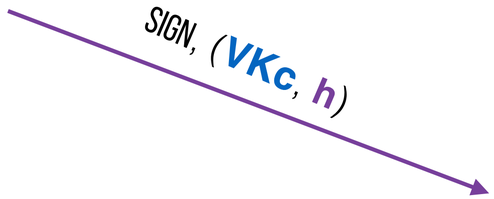


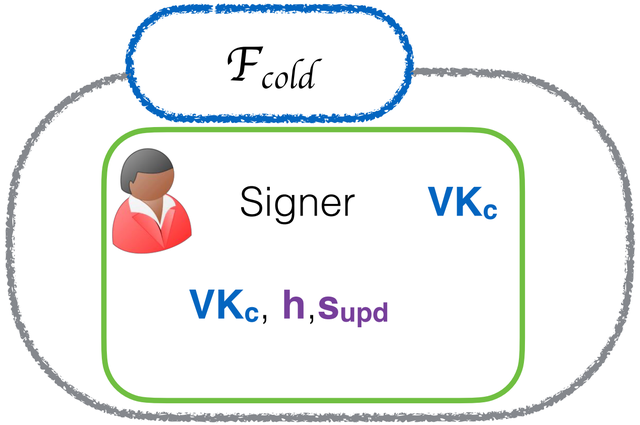
VK_c

VK_{new}

$M \leftarrow \{m_1, m_2\}$

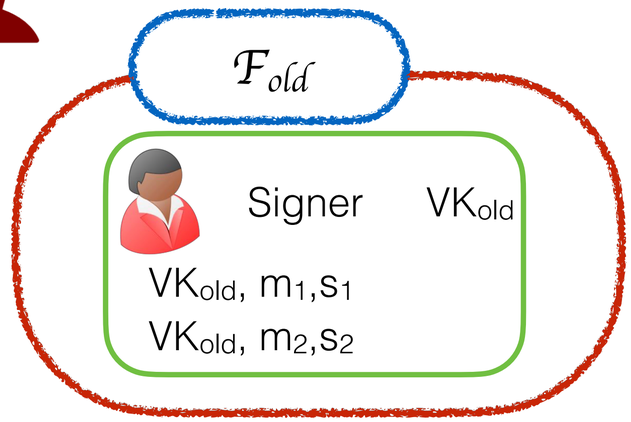
$h \leftarrow H(m_1 || m_2)$

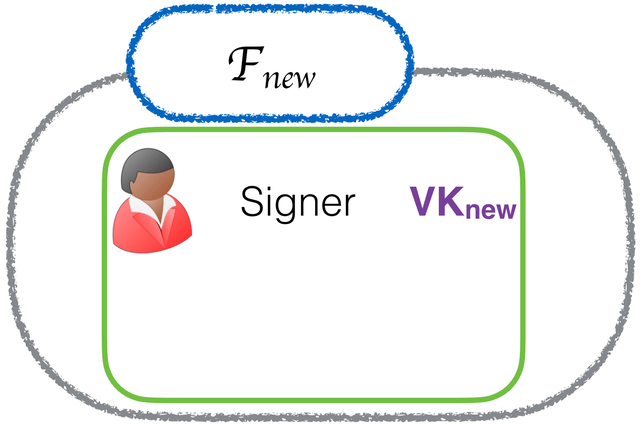




VK_c

Updatable signatures





VK_c

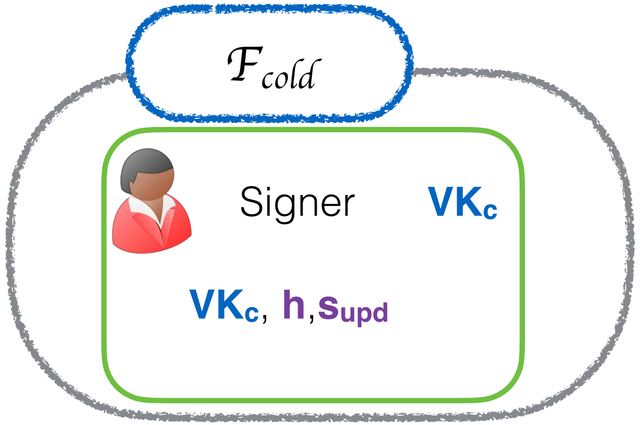
VK_{new}

$M \leftarrow \{m_1, m_2\}$

$h \leftarrow H(m_1 || m_2)$

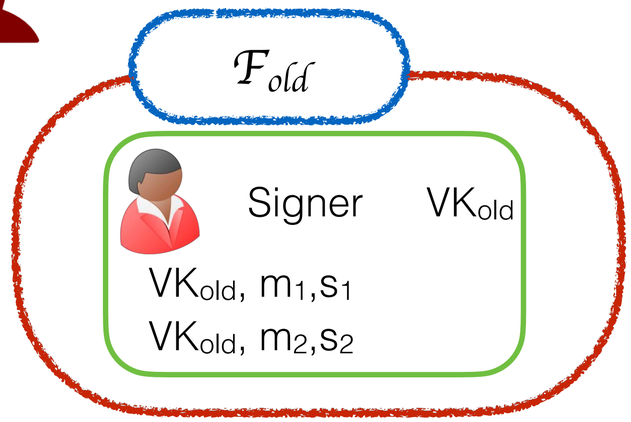
SIGN, (VK_c, h)

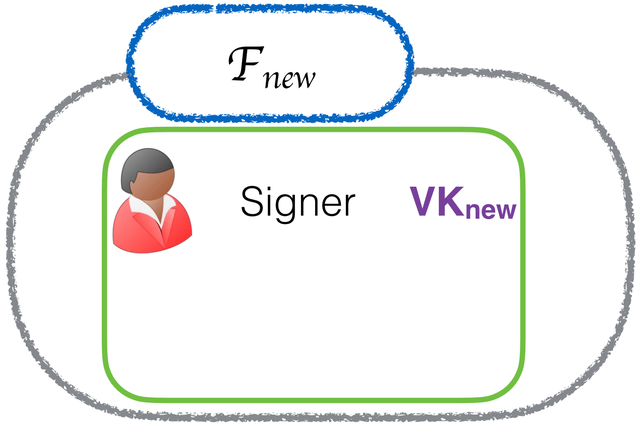
s_{upd}



VK_c

Updatable signatures





VK_c

VK_{new}

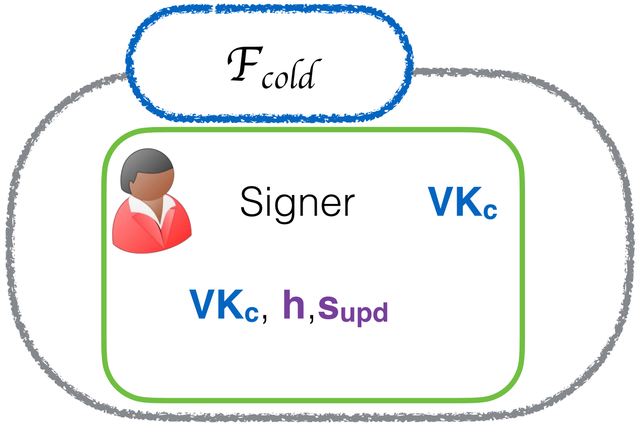
$M \leftarrow \{m_1, m_2\}$

$h \leftarrow H(m_1 \| m_2)$

VK_{new}, M, s_{upd}

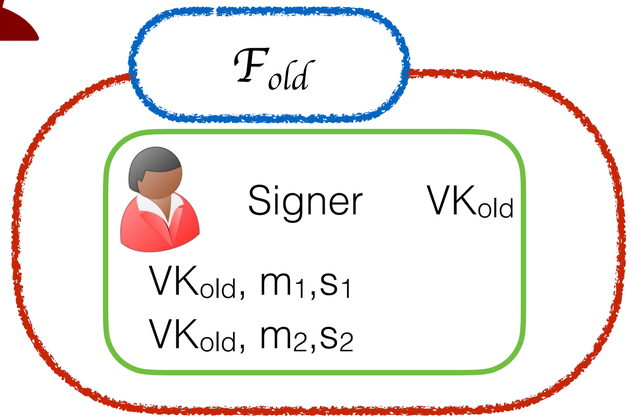
SIGN, (VK_c, h)

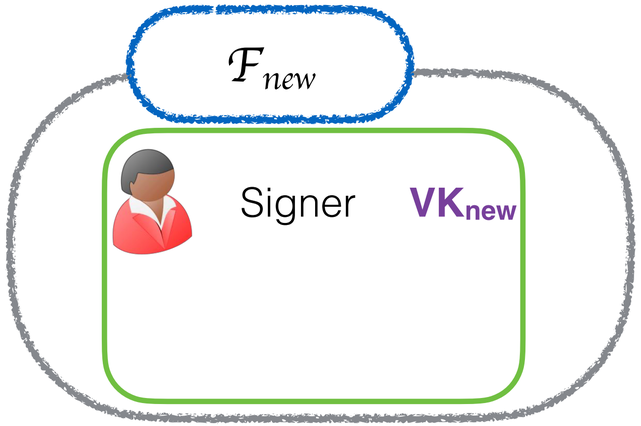
s_{upd}



VK_c

Updatable signatures





UPDATE, $\mathcal{F}_{new}$

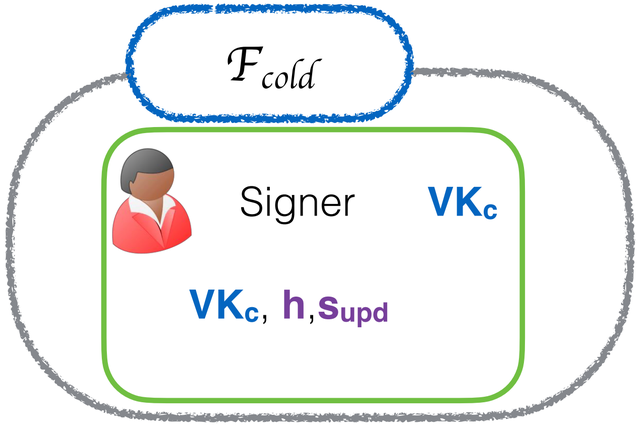
VK_c

VK_{new}

M<—{m_1,m_2}

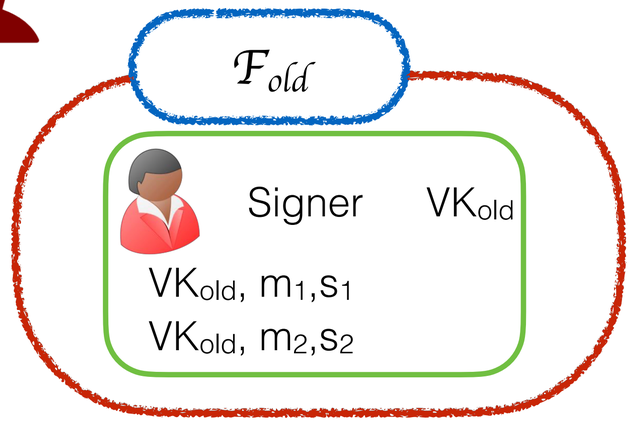
h<—H(m_1||m_2)

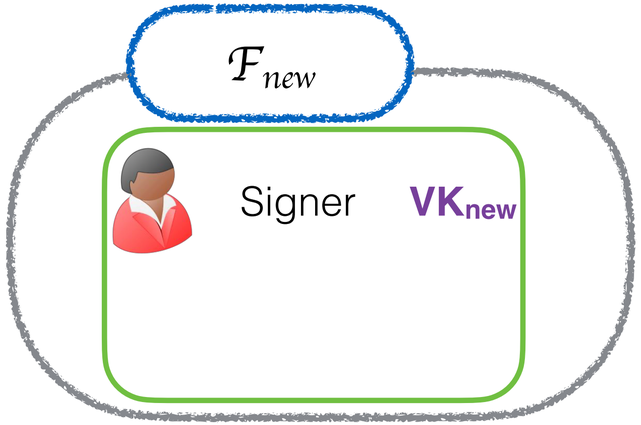
VK_{new}, M, s_{upd}



VK_c

Updatable signatures





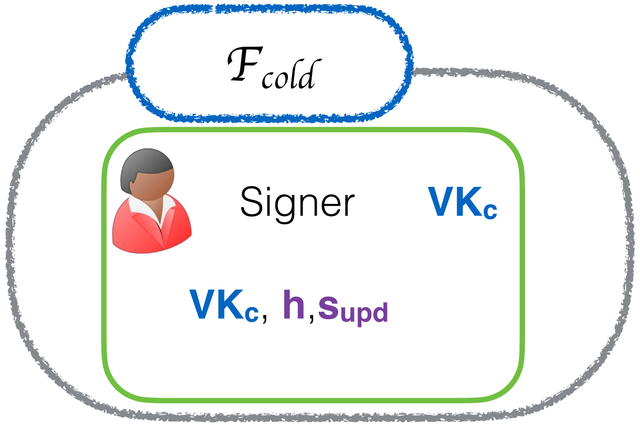
SIGN, m_3

VK_c

VK_{new}

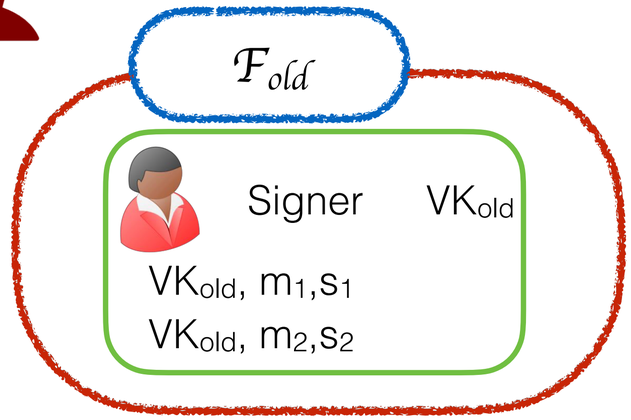
M<—{m_1,m_2}
h<—H(m_1||m_2)

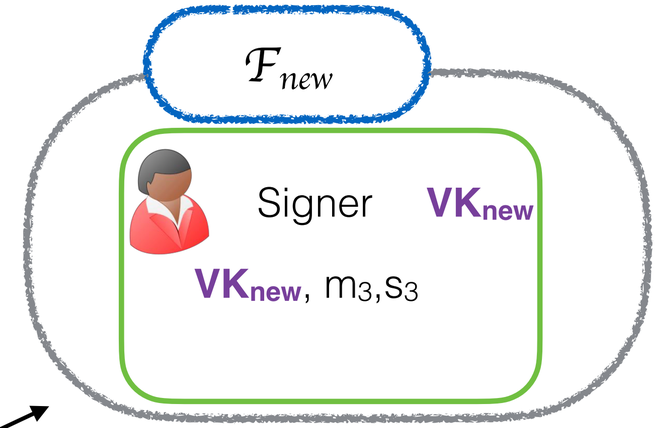
VK_{new}, M, s_{upd}



VK_c

Updatable signatures





SIGN, m_3

s_3

SIGN, m_3

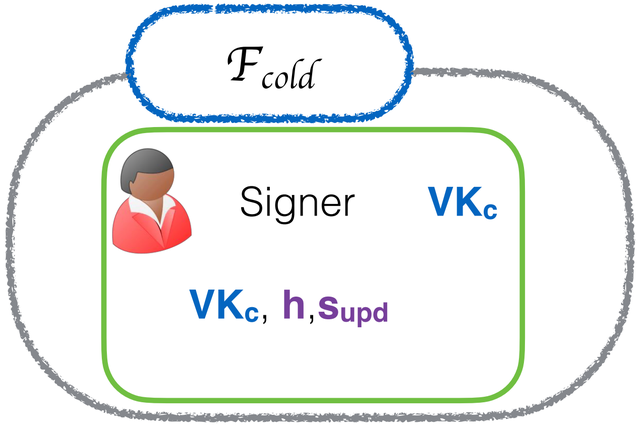
VK_c

VK_{new}

M<—{m_1,m_2}

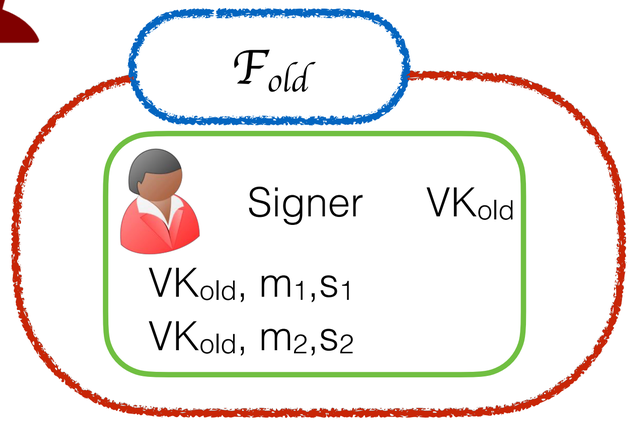
h<—H(m_1||m_2)

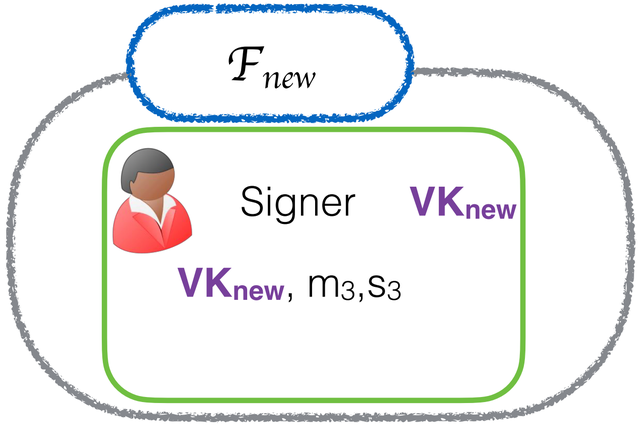
VK_{new}, M, s_{upd}



VK_c

Updatable signatures





SIGN, m_3

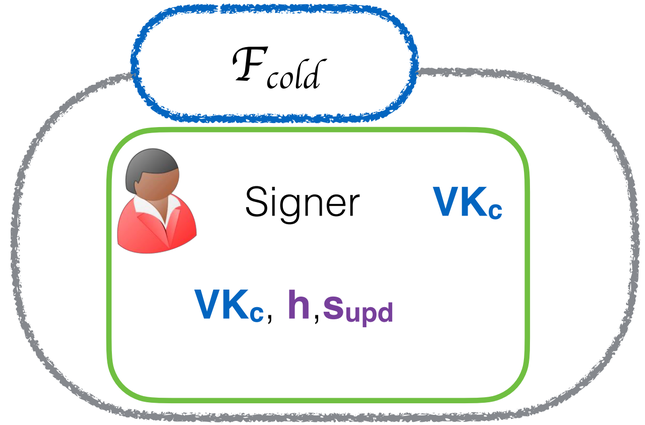
VK_c

VK_{new}

M<—{m_1,m_2}

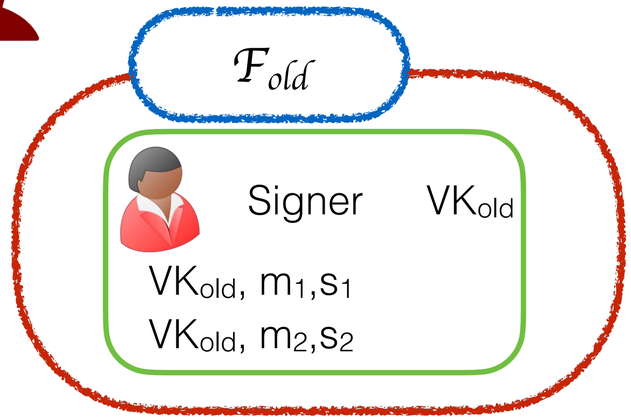
h<—$H(m_1 || m_2)$

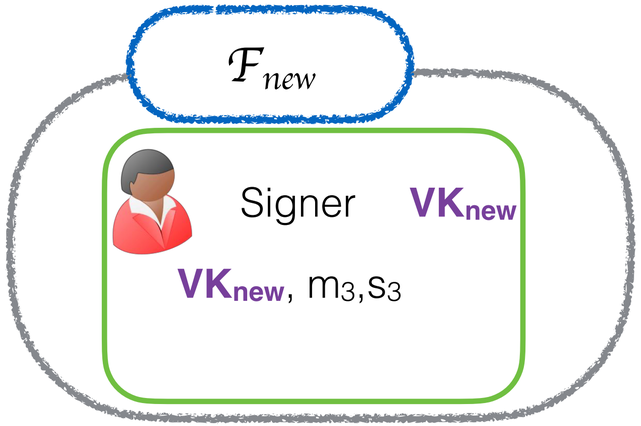
VK_{new}, M, s_{upd}



VK_c

Updatable signatures





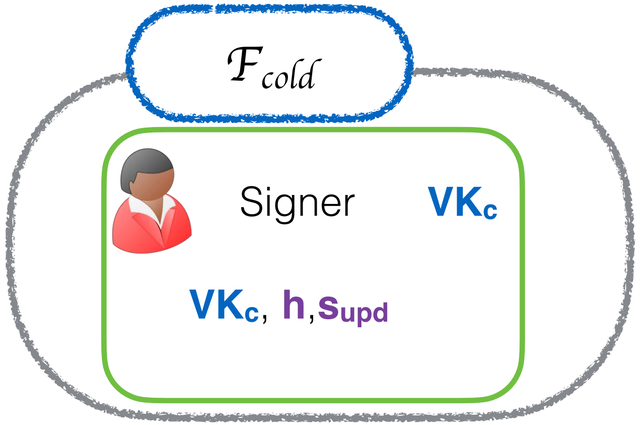
VK_c

VK_{new}

$M \leftarrow \{m_1, m_2\}$

$h \leftarrow H(m_1 || m_2)$

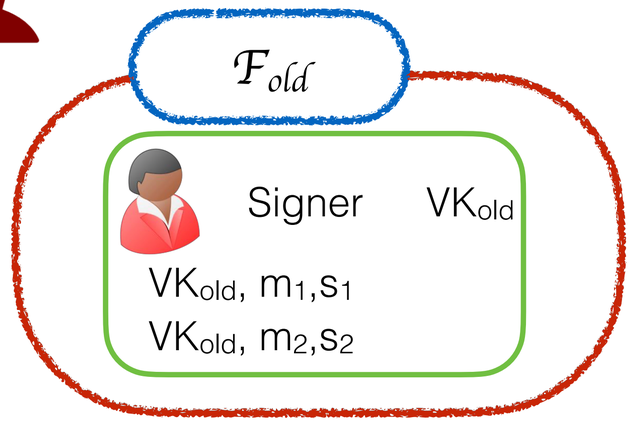
VK_{new}, M, s_{upd}

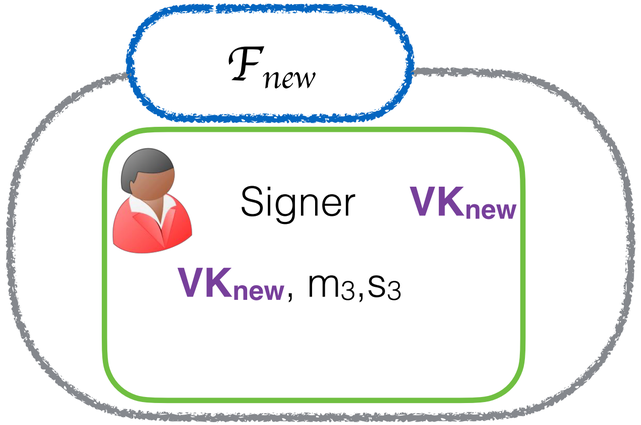


VK_c

VK_{new}, M, s_{upd}

Updatable signatures



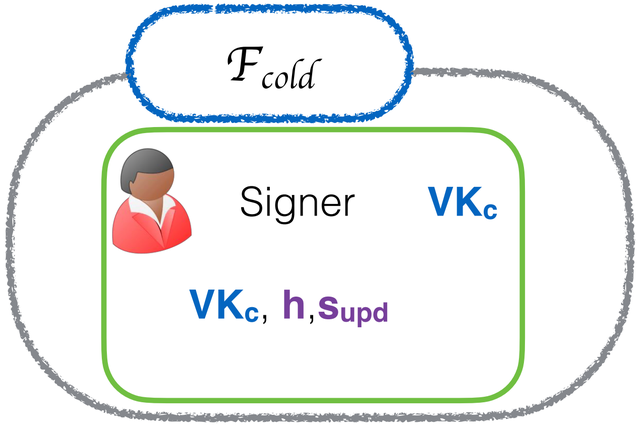


VK_c

VK_{new}

$M \leftarrow \{m_1,m_2\}$
$h \leftarrow H(m_1 \| m_2)$

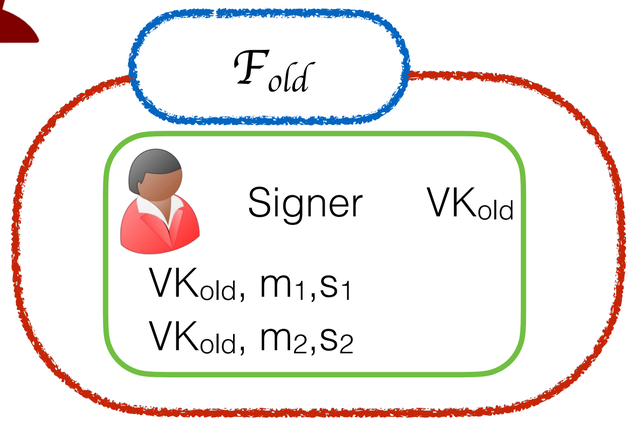
VK_{new}, M, s_{upd}

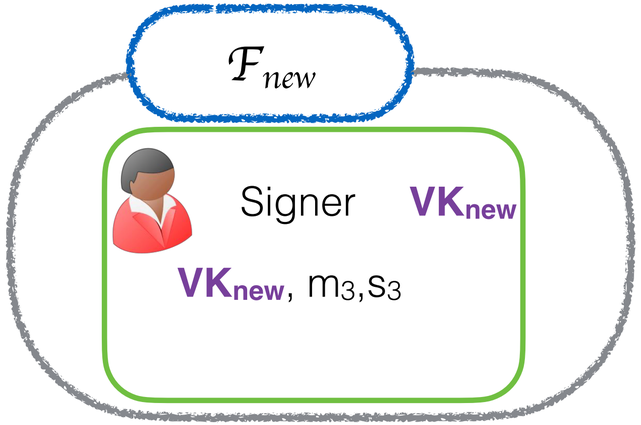


VERIFY, **VK_c**, H(**M**), **s_{upd}**

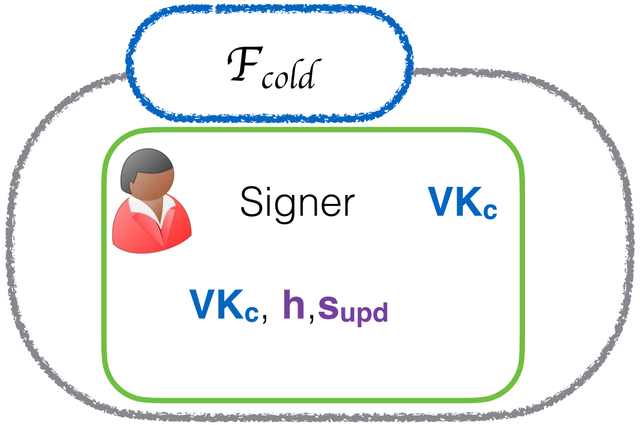
VK_c
VK_{new}, M, **s_{upd}**

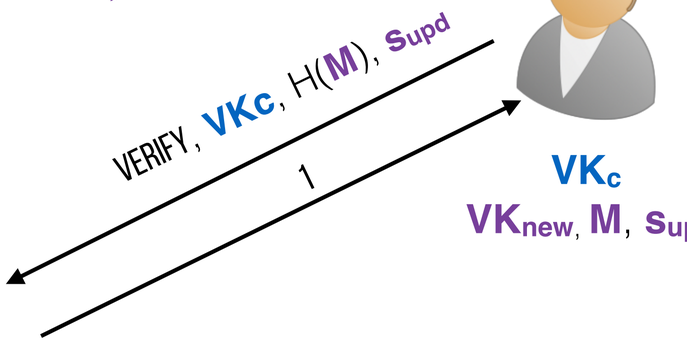
Updatable signatures



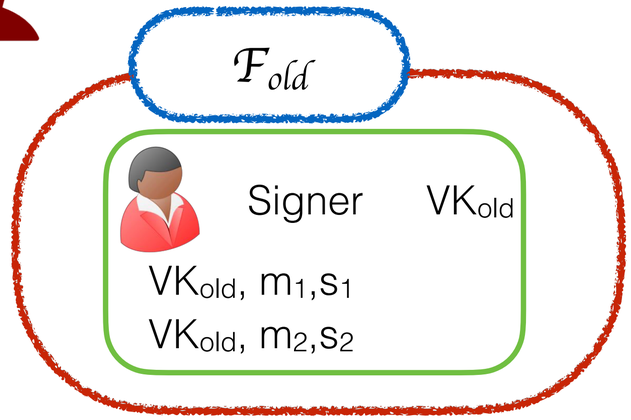


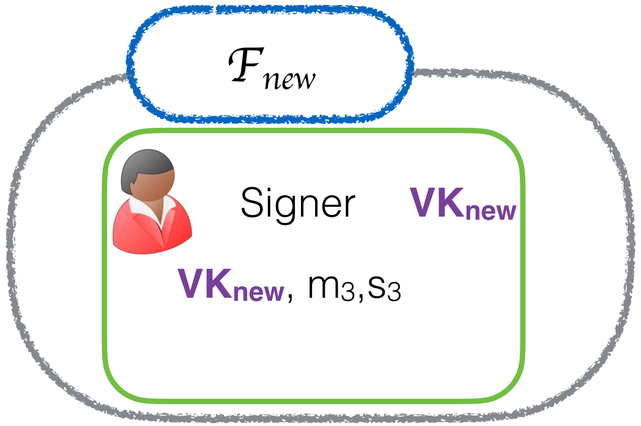
VK_{new}, M, s_{upd}





Updatable signatures





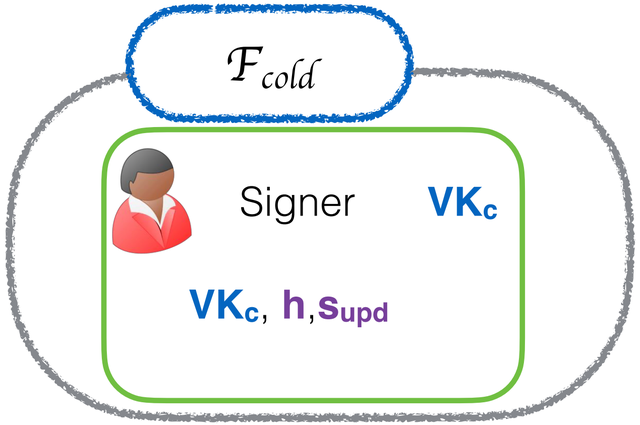
VK_c

VK_{new}

$M \leftarrow \{m_1, m_2\}$

$h \leftarrow H(m_1 || m_2)$

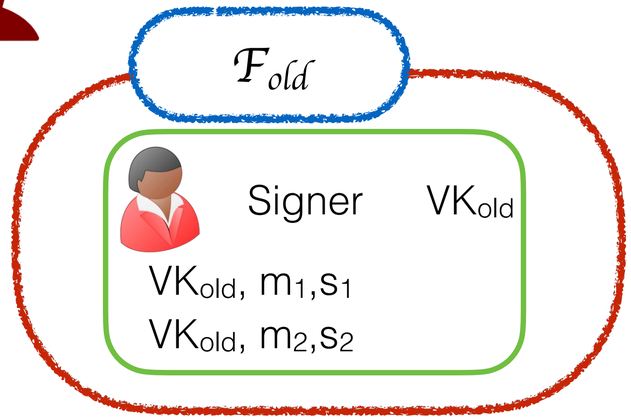
VK_{new}, M, s_{upd}

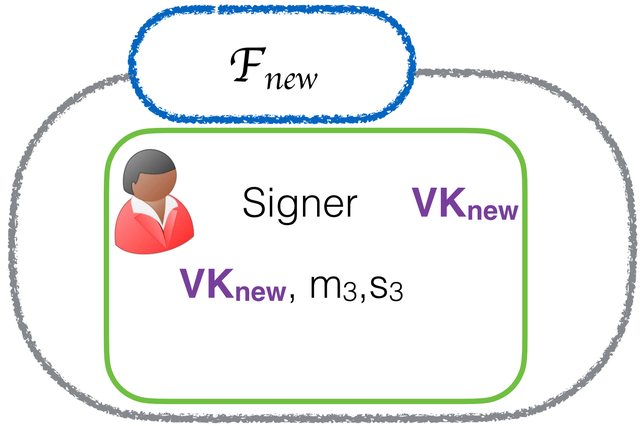


VK_c

VK_{new}, M, s_{upd}

Updatable signatures





SIGN, m_3

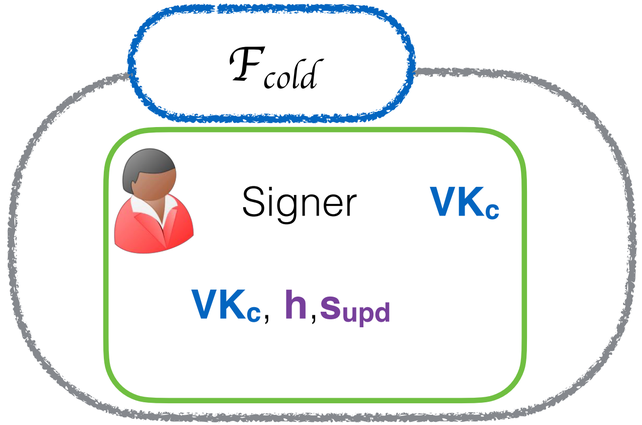
VK_c

VK_{new}

$M \leftarrow \{m_1,m_2\}$

$h \leftarrow H(m_1 \| m_2)$

VK_{new}, M, s_{upd}

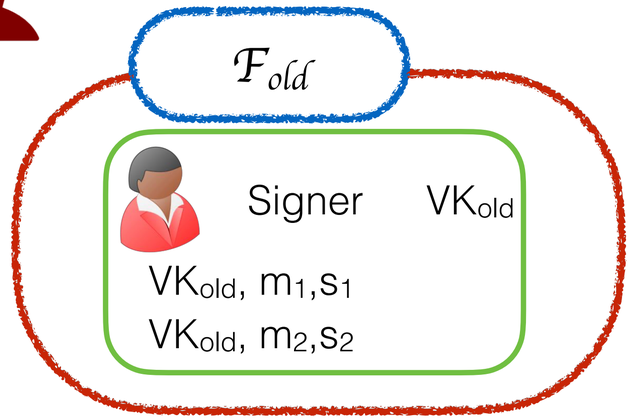


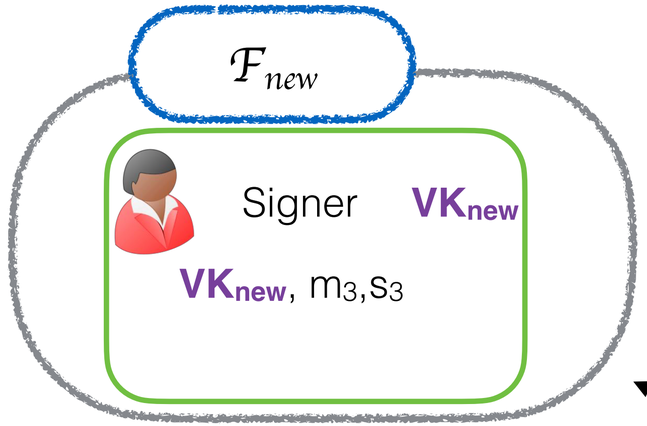
VERIFY, vk_{old}, m_2, s_2

VK_c

VK_{new}, M, s_{upd}

Updatable signatures





VERIFY, VK_{new}, m_2, s_2

SIGN, m_3

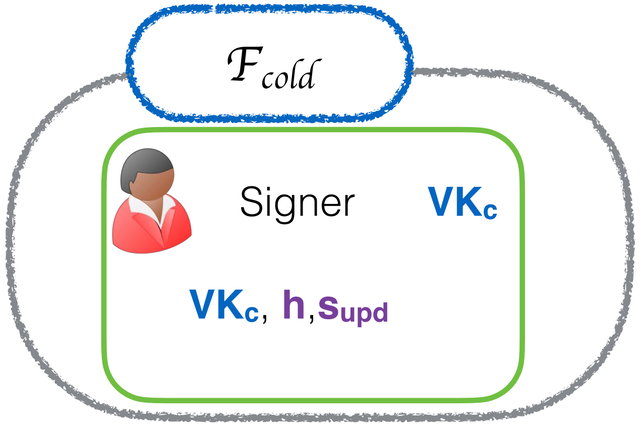
VK_c

VK_{new}

$M \leftarrow \{m_1,m_2\}$

$h \leftarrow H(m_1 \| m_2)$

VK_{new}, M, s_{upd}

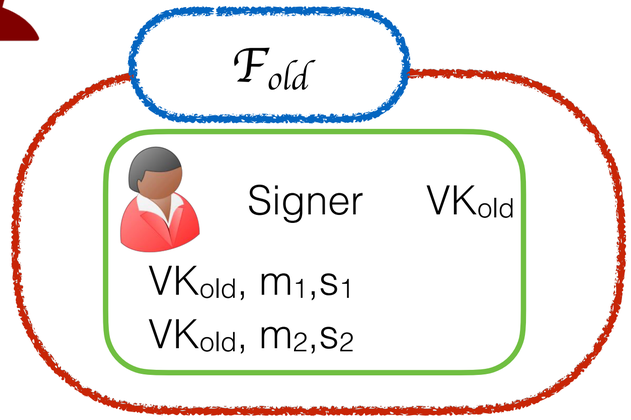


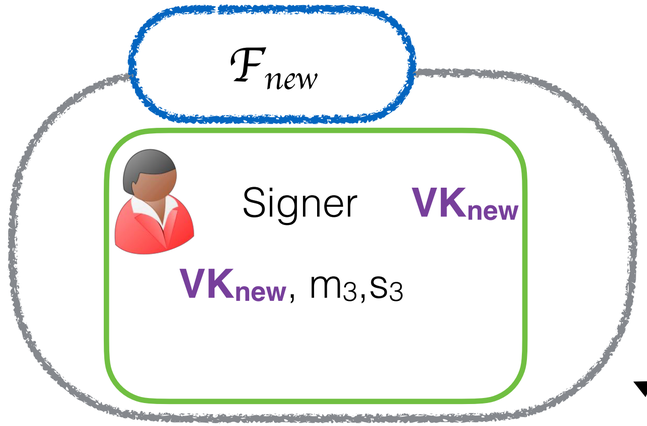
VERIFY, vk_{old}, m_2, s_2

VK_c

VK_{new}, M, s_{upd}

Updatable signatures





VERIFY, VK_{new}, m_2, s_2

b

SIGN, m_3

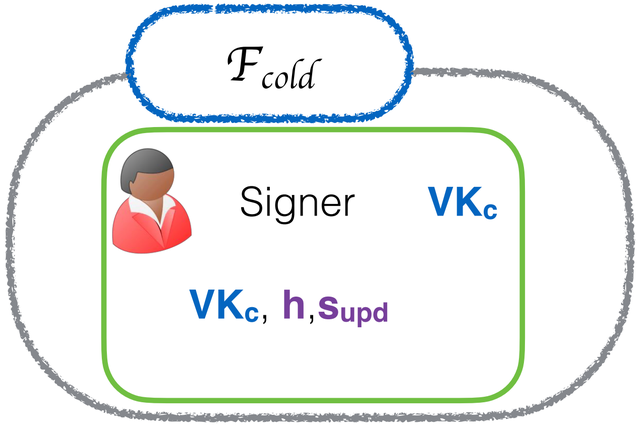
VK_c

VK_{new}

M<—{m_1,m_2}

h<—$H(m_1||m_2)$

VK_{new}, M, s_{upd}

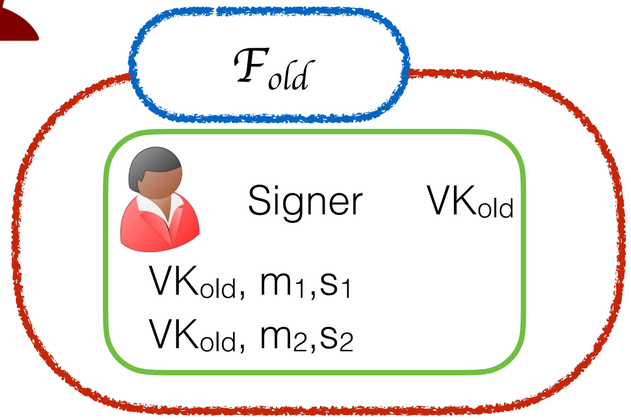


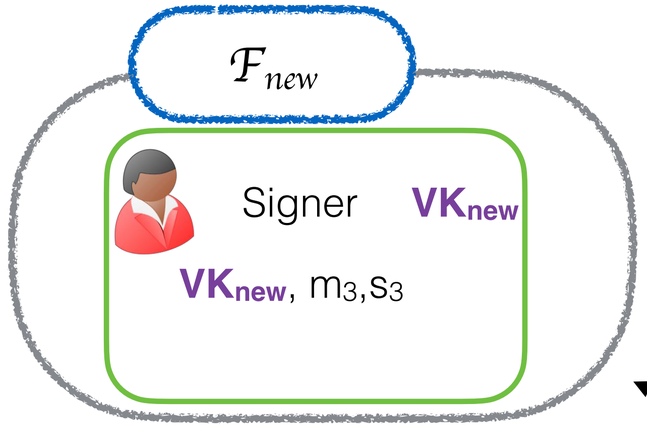
VERIFY, vk_{old}, m_2, s_2

VK_c

VK_{new}, M, s_{upd}

Updatable signatures





VERIFY, VK_{new}, m_2, s_2

b

SIGN, m_3

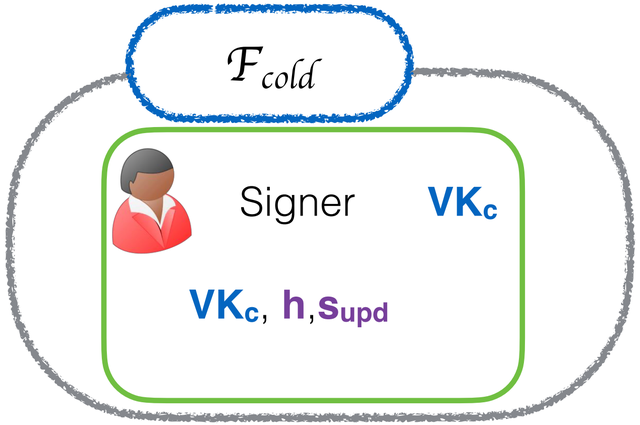
VK_c

VK_{new}

M<—{m_1,m_2}

h<—$H(m_1 || m_2)$

VK_{new}, M, s_{upd}



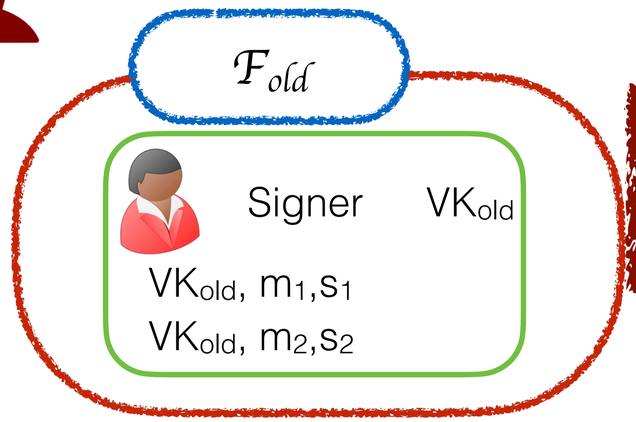
VERIFY, vk_{old}, m_2, s_2

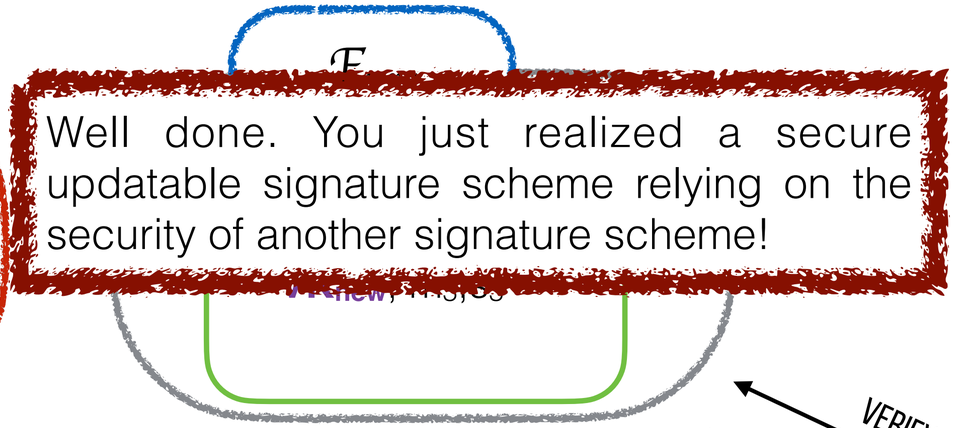
VK_c

VK_{new}, M, s_{upd}

if $m_2 \in$ **M** or b=1 then return 1

Updatable signatures





Well done. You just realized a secure updatable signature scheme relying on the security of another signature scheme!

VERIFY, **VK_{new}**, m_2, s_2

b

SIGN, m_3

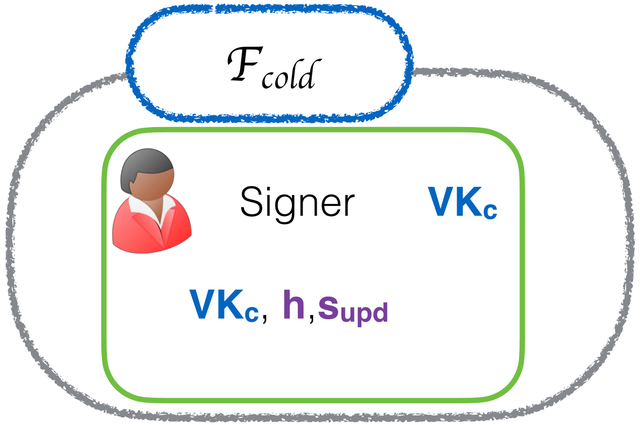
VK_c

VK_{new}

M<—{m_1,m_2}

h<—H(m_1||m_2)

VK_{new}, M, s_{upd}



VERIFY, vk_{old}, m_2, s_2

VK_c

VK_{new}, M, s_{upd}

if m_2 ∈ **M** or b=1 then return 1

Updatable signatures

$\mathcal{F}_{old}$

Signer VK_{old}

VK_{old}, m_1,s_1

VK_{old}, m_2,s_2

$\mathcal{F}$

Well done. You just realized a secure updatable signature scheme relying on the security of another signature scheme!

- $\mathcal{F}_{cold}$ could be a post-quantum secure, **inefficient** scheme used only for the updates
- The support of $\mathcal{F}_{cold}$ is needed only if $\mathcal{F}_{old}$ is believed to be compromised

SIGN, m_3

VK_c

VK_{new}

$M \leftarrow \{m_1, m_2\}$

$h \leftarrow H(m_1 \| m_2)$

$\mathcal{F}_{cold}$

Signer **VK_c**

VK_c, **h**,**s_{upd}**

VK_{new}, m_2, s_2

b

VERIFY, vk_{old}, m_2, s_2

VK_c

VK_{new}, **M**, **s_{upd}**

if $m_2 \in$ **M** or b=1 then return 1

Conclusions

- Cryptographic protocols need to be updated. This should be considered when designing the protocol
- Propose a framework to capture updates
- Model compromised protocols via the corruption of hybrid functionalities
- Extend our result to the case of non-interactive zero-knowledge and PRF

Willing to do a postdoc in multi-party computation at The University of Edinburgh? Reach me out

No hot summers!

Thank you