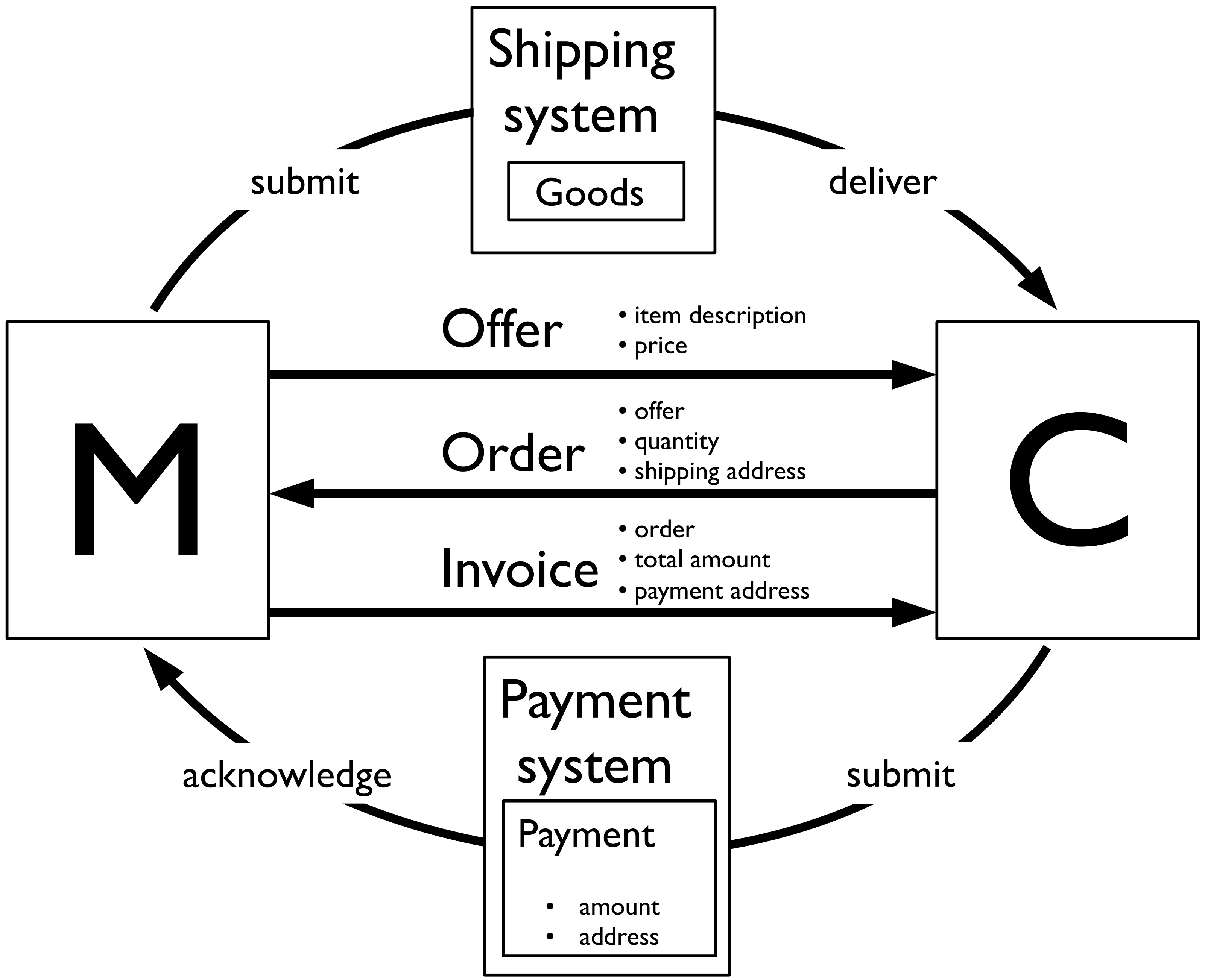
Secure eCommerce with Bitcoin

eCommerce with an irreversible payment system

Timo Hanke, RWTH Aachen
Ilja Gerhardt, MPI Stuttgart

PGP:
Timo Hanke 1EFF 69BC 6FB7 8744 14DB 631D 1BB5 D6E3 AB96 7DA8
Ilja Gerhardt 1986 0949 8102 817D C3E3 B5AA 3CF6 E44C 7EF3 637B

The Merchant-Customer relation



Assumptions

General scheme

- cash in advance
- no escrow
- irreversible payments and shipments

Asymmetry

Identities

- merchant has public identity (PKI)
- customer has no identity

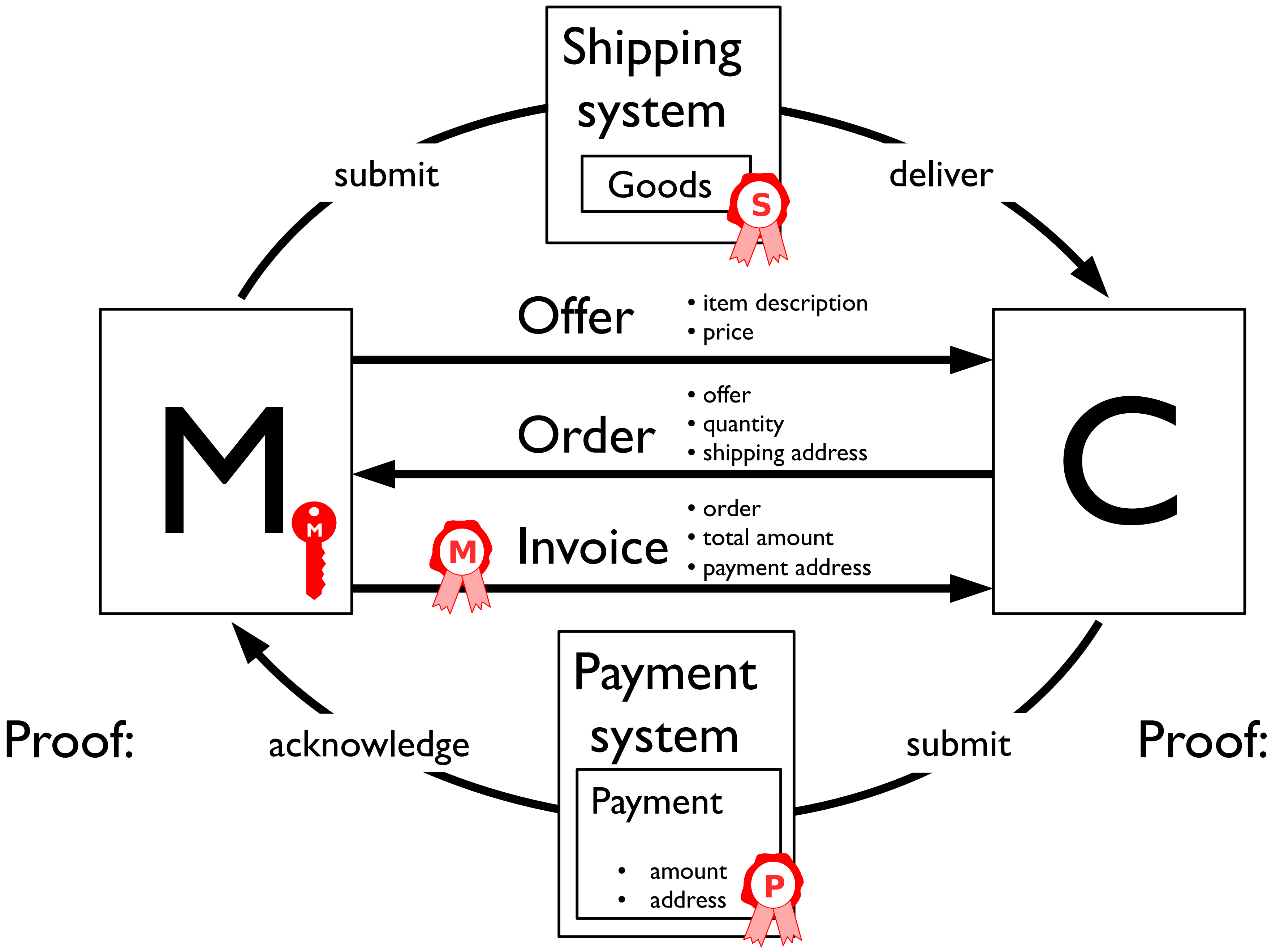
Trust

- both parties do not fully trust each other
- merchant cares about reputation

Consequence

- merchant can sign
- customer cannot sign anything

Signatures


Shipping system
Goods
S
submit
deliver
M
Offer
• item description
• price
Order
• offer
• quantity
• shipping address
M
Invoice
• order
• total amount
• payment address
C
Proof:
acknowledge
Payment system
Payment
• amount
• address
P
submit
Proof:

Attack Types

Attack on	Counter-measure	
Merchant bitcoin funds	Pay directly to cold storage	✓
Customer bitcoin funds	• Cold storage trusted device to sign transaction • Multisignature storage	✓
Payment protocol: payment and shipping address	?	

Cold Storage:

₿ **Secret key** offline	₿ **Public address** online

Presence of attackers

Assumption (new)

- all online infrastructure is compromised
- the communication channel is compromised

offline

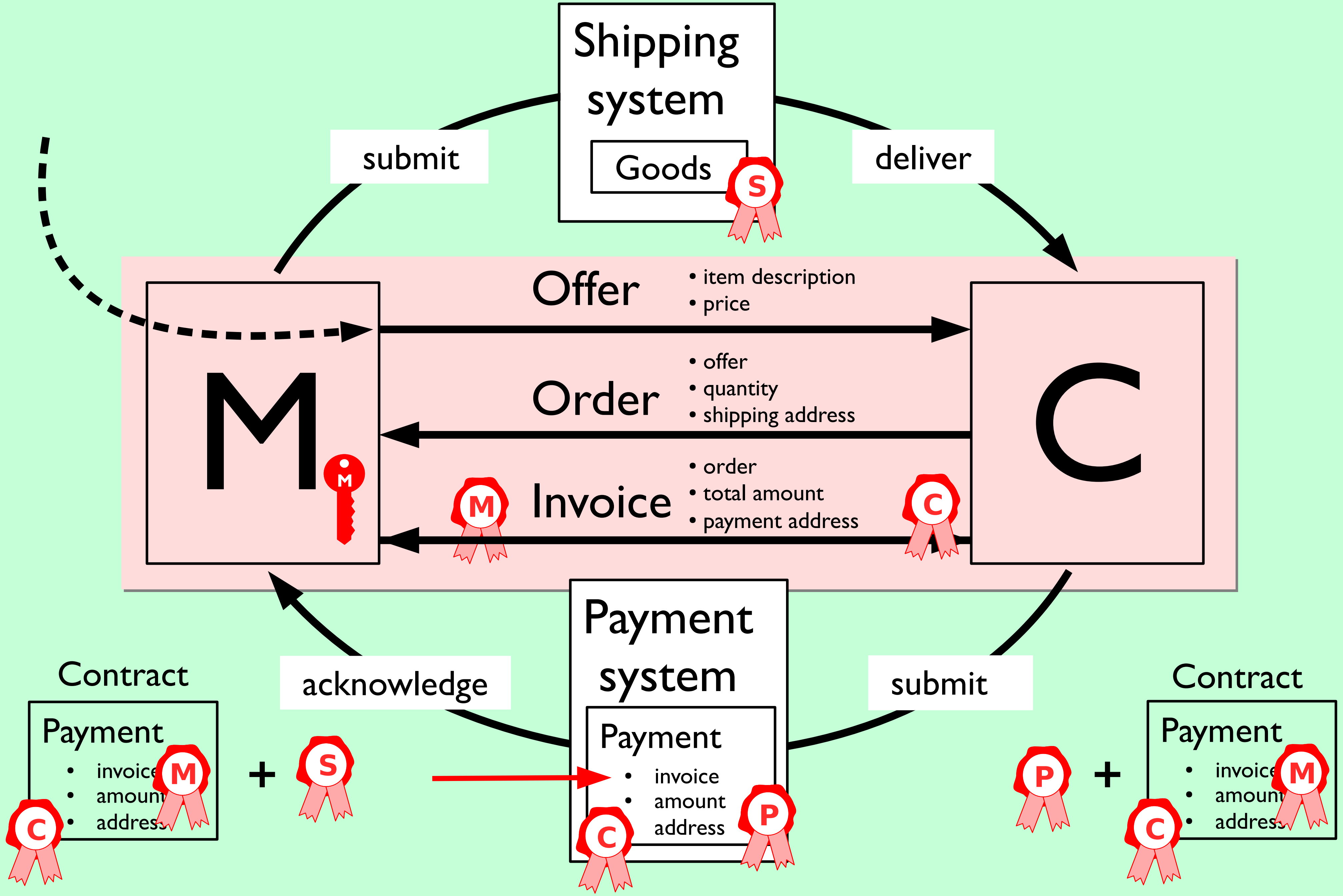
offline

Merchant conclusion

- don't sign at order time!
- can't sign invoice

Problem: need customer's signature

Payment as Contract



Link metadata to ₿ address

Task: generate a ₿ address such that
a unique given hash is verifiably linked to it
(not necessarily visible in clear text)

Owner of a bitcoin address

person who knows the corresponding private key

address owned, signed by merchant

payment address

Task: given *P*, derive a pubkey *P[m]* with the same owner such that the unique given *m* is verifiably linked to *P[m]*

hash of invoice

ECDSA keypair homomorphism

Keypair (s,P)

- fixed large prime N
- private key s is integer in range 0,...,N-1
- public key $P = P(s)$ is function of s (bitoin address)

Homomorphic property

Not possible with RSA!

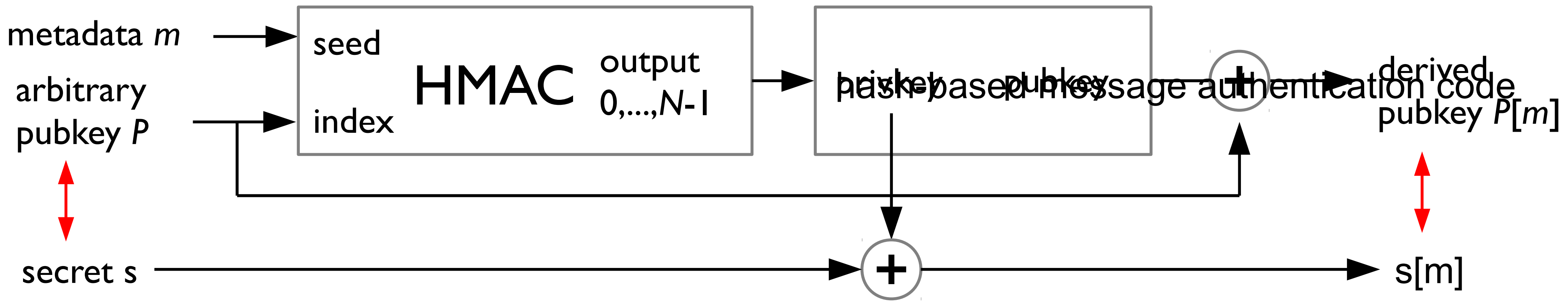
(s,P), (t,Q) keypairs => ($s+t$ mod N, $P+Q$) keypair

Owners

If t is publicly known then P and $P+Q$ have the same owner.

ECDSA linking

Task: given *P*, derive a pubkey *P[m]* with the same owner such that the unique given *m* is verifiably linked to *P[m]* ✓



Properties

- given (*P*,m), it is impractical to find (*P*',*m*') != (*P*,*m*) with *P*'[*m*'] = *P*[*m*], hence *m* the unique metadata linked to *P*[*m*] collision resistance

- P and P[m] have the same owner (*m* known) keypair homomorphism

Example

-----BEGIN PGP SIGNED MESSAGE-----
Hash: SHA1

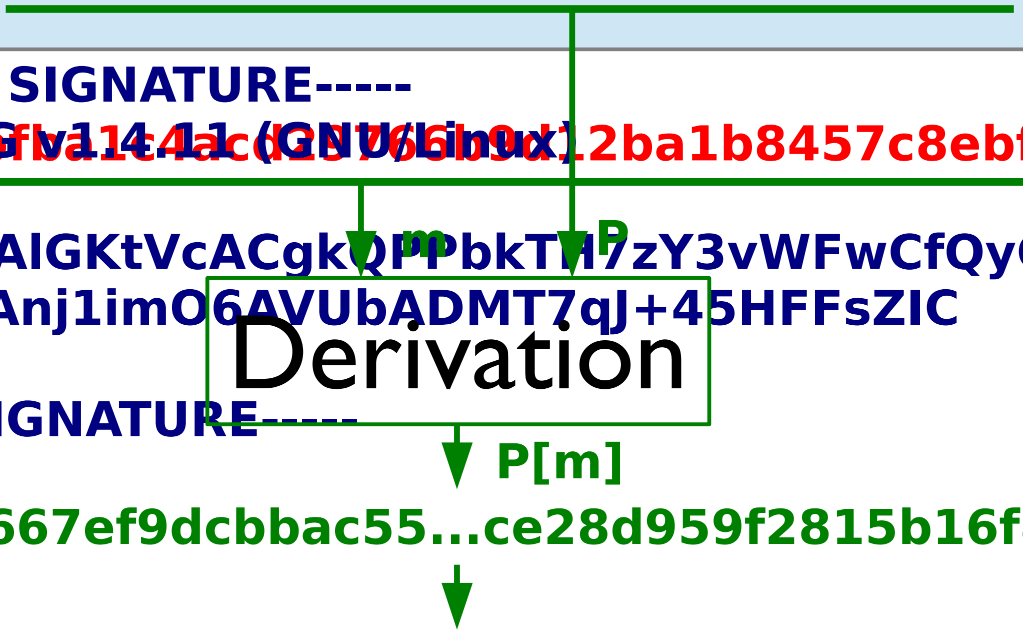
Tea set
Teapot classic (BTC 3.90): 1 pcs
Mug (BTC 0.89): 4 pcs
Teaspoon (BTC 0.49): 2 pcs
Delivery address: John Doe, 150 W San Carlos St, San Jose, CA 95113
PaymentBase: 043f30a7e...1bb6300bfc23aa7e0f03cd

M

SHA256SUM:
02a364f5231bfba1c4acd2370b1b9d12ba1b8457c8ebf46b87cd637...

-----BEGIN PGP SIGNATURE-----
Version: GnuPG v1.4.11 (GNU/Linux)

IEYEARECAAYFAlGKtVcACgkQPPbkTH7zY3vWFwCfQyCHg1zmYGZiftjL
C15m0hKBYsYAnj1imO6AVUbADMT7qJ+45HFFsZIC
=37MT
-----END PGP SIGNATURE-----



0279be667ef9dcbbac55...ce28d959f2815b16f81798

Transfer Ƀ 8.44 to 1BgGZ9tcN4rm9KBzDn7KprQz87SZ26SAMH C

Summary

Protocol

- bitcoin funds in cold storage
- merchant only pre-signs, not at order time
- customer generates invoice and payment address
- customer only signs payment (identity = payer)

Lack of mutual trust

- payment + invoice = contract

Third party attackers

- attacker cannot steal funds, nor redirect goods
- no SSL-communication required

Anonymity

Additional features

- anonymity well-protected (with randomized invoice)
- compatible with multi-transaction payments

Thank you!

Homomorphic Payment Addresses & the Pay-to-Contract Protocol:
http://arxiv.org/abs/1212.3257