

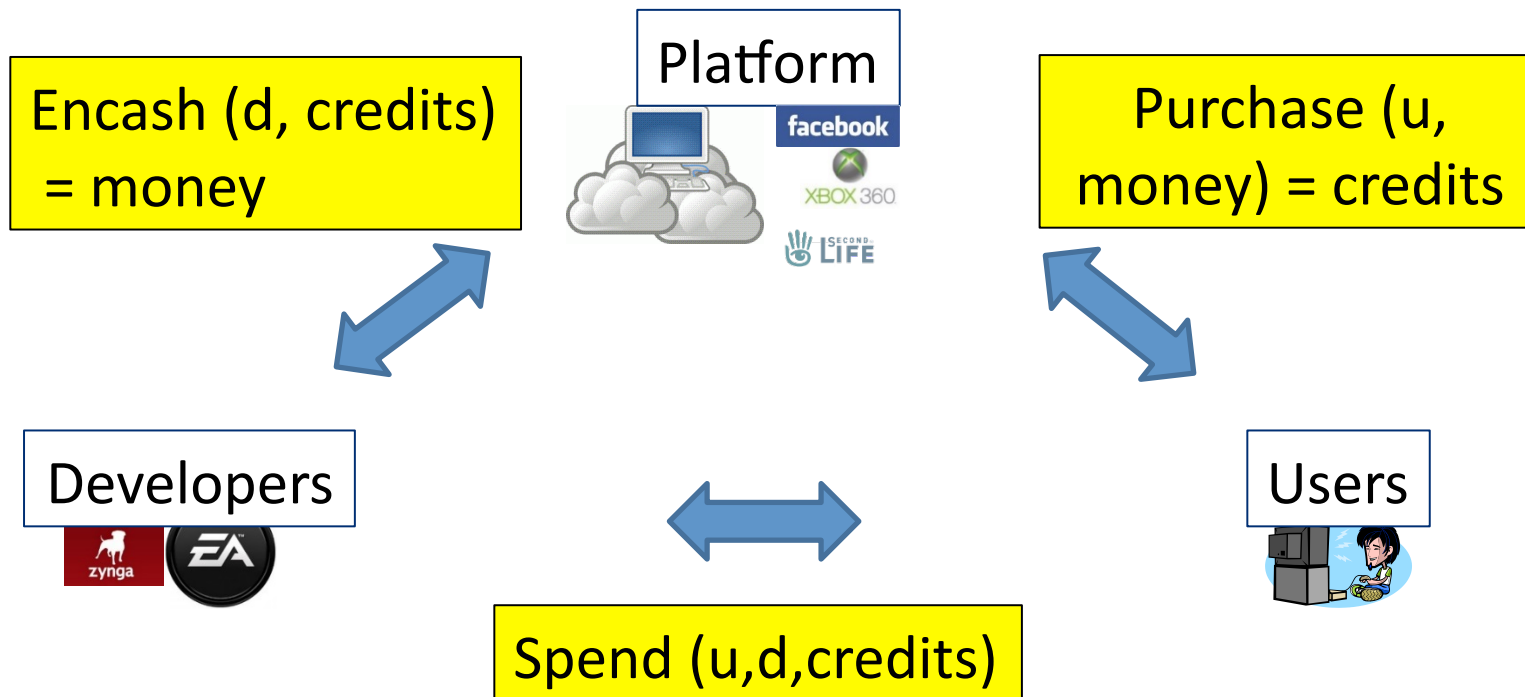
Verito: A Practical System for Transparency
and Accountability in Virtual Economies
(Don't count your credits before they are cashed)

Raghav Bhaskar, Saikat Guha,
Srivatsan Laxman, **Prasad Naldurg**

Microsoft Research India



Virtual economies



All credits are not worth the same

- Value of credits
 - “Facebook has in the past issued a small amount of Credits at no cost to certain users (for example, a new user of Credits, or someone whose usage has lapsed). **If you receive those Credits in transactions, we will not redeem them.**”
 - Promotions by Facebook and Advertisers (cost borne by FB and advertisers; see image)
- Credits in different currencies
 - Arbitrage

The screenshot shows the 'Earn City Cash' interface with the following offers:

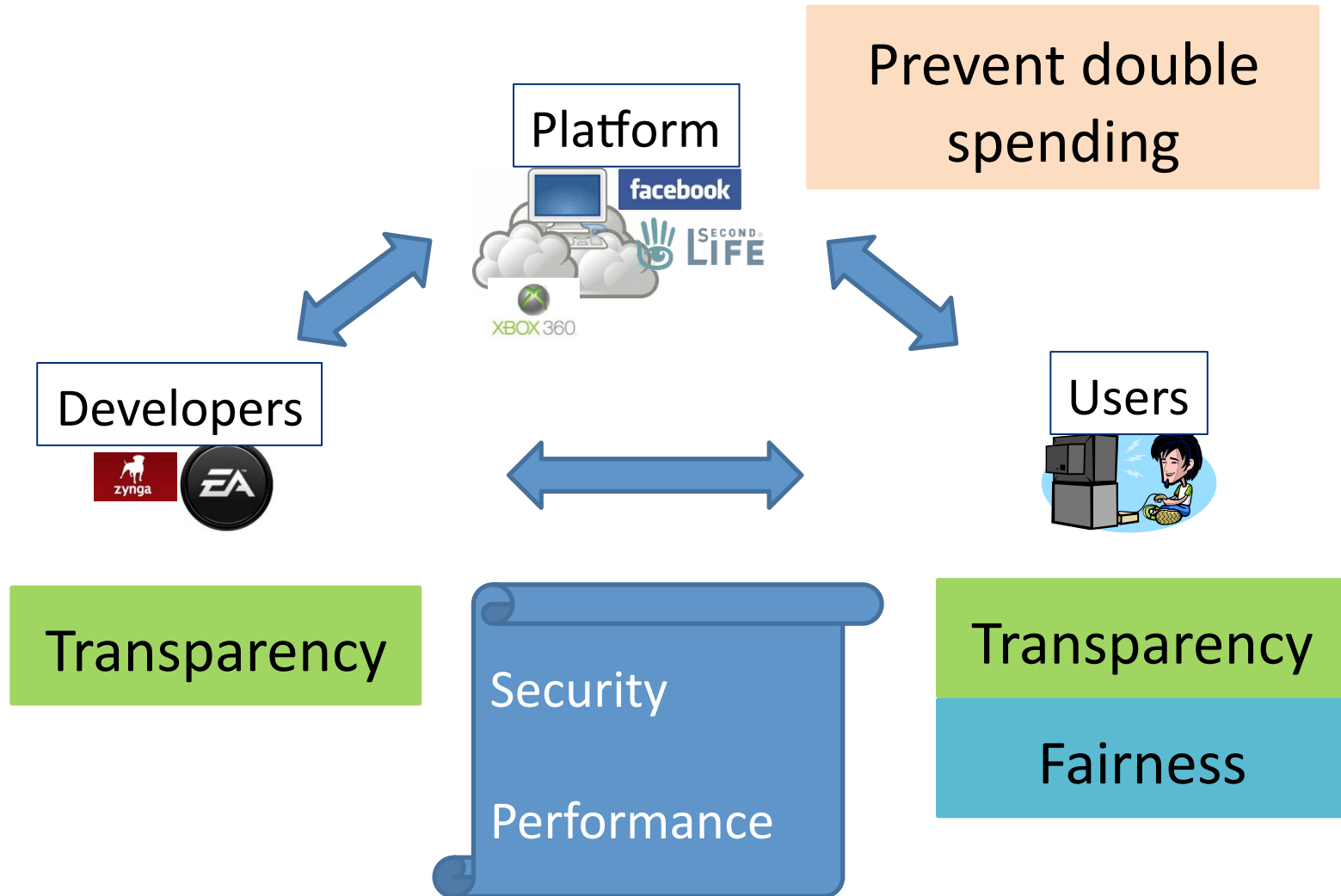
Offer	City Cash Value
FTD Flowers & Gifts - \$19.99 (Up to 50% off Valentine's Day gifts)	152 City Cash
ProFlowers & Gifts - \$19.99 (Only 5 days until Valentine's Day!)	144 City Cash
Special Facebook Credits Offer (Spend \$1 to get \$4 Free)	32 City Cash
Discover More Card (Get approved for a credit card)	760 City Cash
Netflix (Instantly watch movies from Netflix)	176 City Cash
GameFly (Sign up for video game rentals)	200 City Cash

The 'Special Facebook Credits Offer' is highlighted with a red border. A 'Done' button is visible at the bottom right of the interface.

Lack of Transparency

- Platform pays out different amounts of cash for same # of credits
 - Developers cannot do fine-grained accounting
 - Rely on trust, regulation
- Not very popular
 - FB does not do free credits any more
 - Cost of differentiation passed on to developers
 - Linden \$ and Bitcoin have private exchanges
 - You get what somebody else thinks you are worth
 - MS points were withdrawn as they were priced differently in different currencies
 - Arbitrage

Requirements wishlist



Solution: first cut

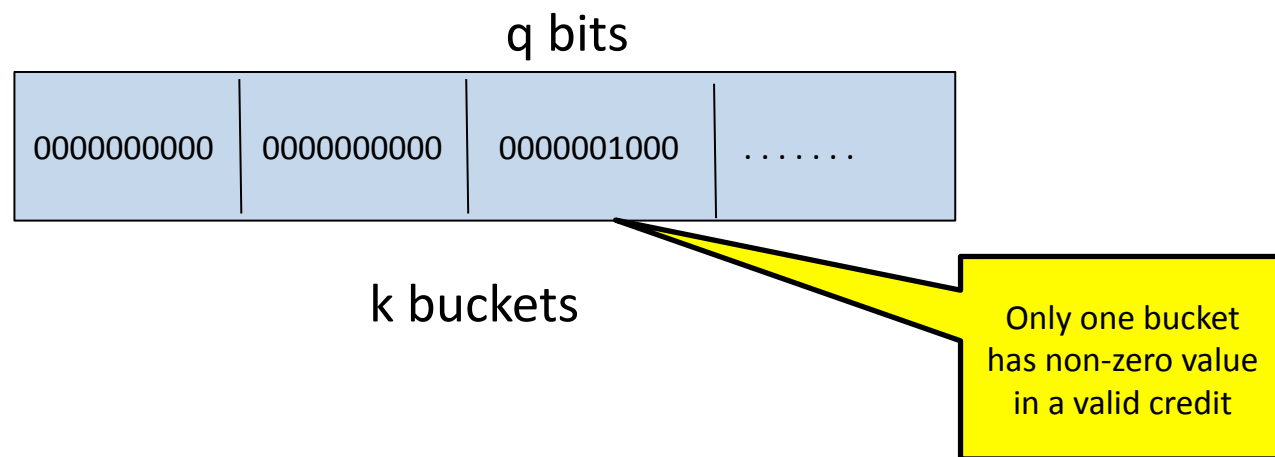
- Credits are just value-signature pairs
 - Transparency
 - Security
 - Fairness
 - Performance
 - Double spending





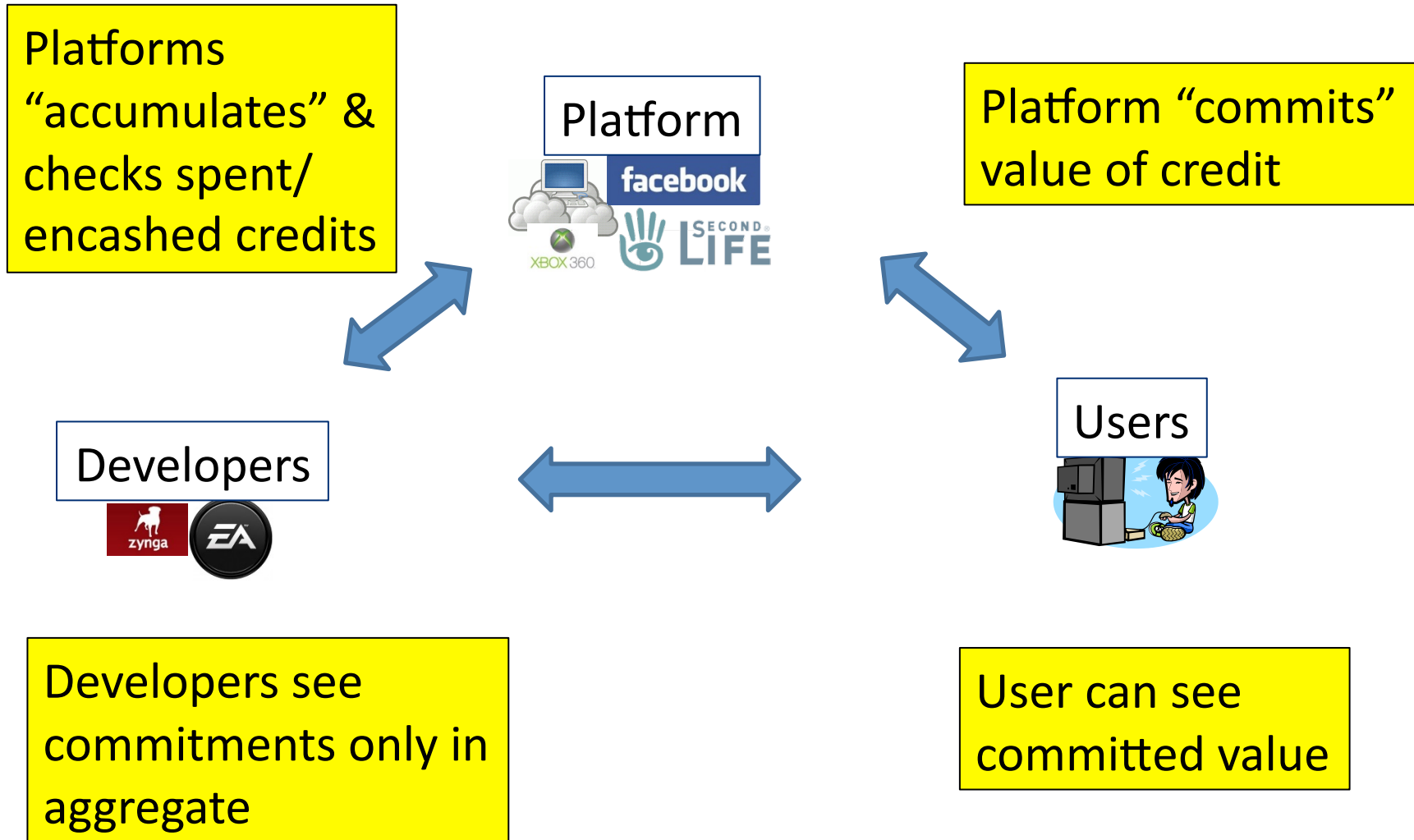
Verito construction

- A practical, efficient, secure credits system for transparency and fairness in virtual economies
- Design: Credit(q, k)
 - Credit is a commitment on a nominal value
 - q : security parameter, k : buckets (aggregation level)





Solution Overview



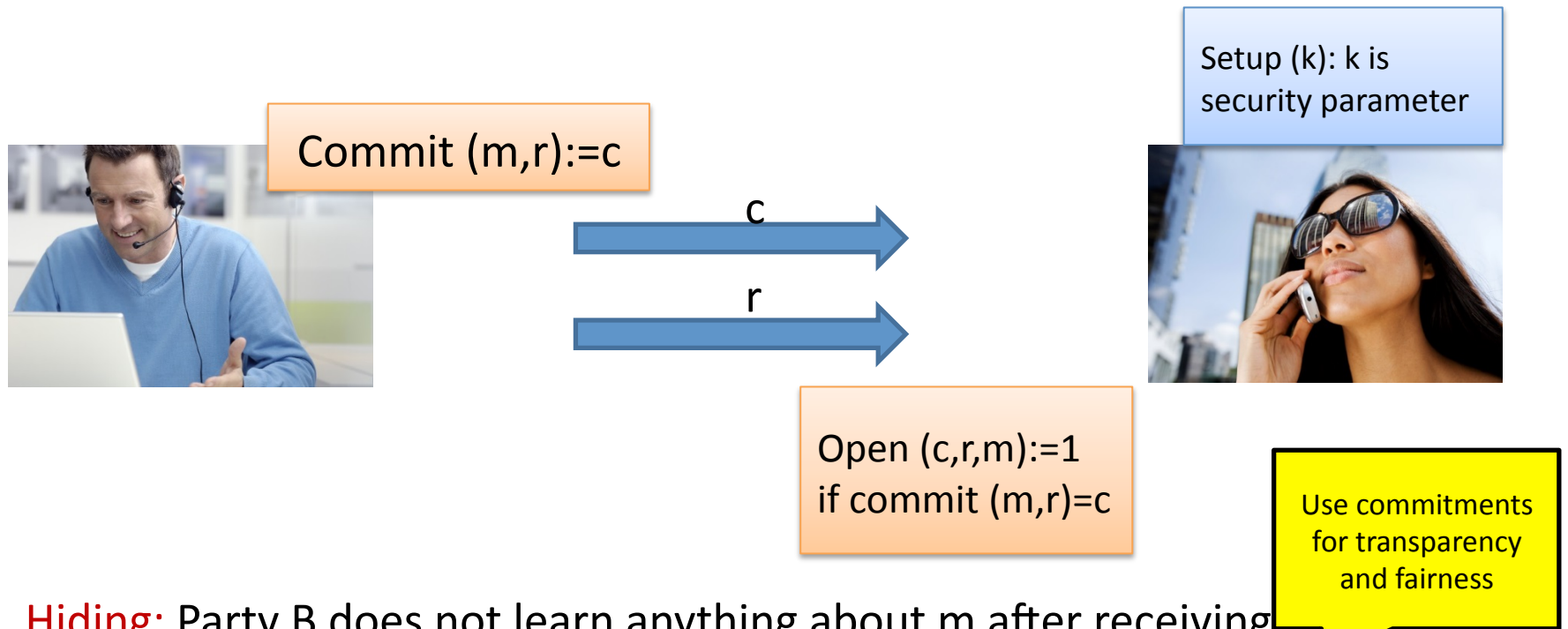


Key ingredients

- Homomorphic commitments
- Dynamic Accumulators

- Compare/contrast with ecash
 - Anonymity, unlinkability
- Anonymous credentials

Commitment schemes



Hiding: Party B does not learn anything about m after receiving c

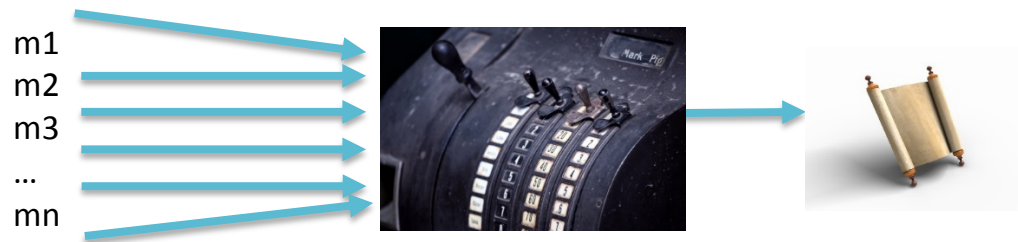
Binding: Party A cannot claim different m for committed c

Homomorphic: $\text{Commit}(m_1) * \text{Commit}(m_2) = \text{Commit}(m_1 + m_2)$

Homomorphic Commitment Scheme [Pedersen]

- Commitment on m is $c = g^r h^m$; Note $c_1 * c_2 = g^{r_1} h^{m_1} * g^{r_2} h^{m_2} = g^{r_1+r_2} h^{m_1+m_2}$
- User gets to see c , m and r and thus can check correctness of all its credits
- Developer gets only c and cannot infer anything about m
- During Encash, Platform reveals $\sum r_i$ and $\sum m_i$ to the Developer who can check it against the product of credits ($\prod c_i$) that it submitted

Dynamic Accumulators



- "Hash" a large set of input values to a single short value
- Check for value is (not) in accumulator using a "witness"
- Infeasible to find witness for a value not in accumulator
 - Typically used for efficient (credential) revocation
 - WE DON'T USE ZERO KNOWLEDGE
- Dynamic Accumulators
 - Addition, deletion efficient, independent of accumulator values
 - Used to check double spending, double encashment

$\text{AccGen}(1^k)$: Generate accumulator

Inputs: c_1, c_2, c_3, \dots

Outputs: α, w_1, w_2, w_3

$\text{AccAdd}(c_k, \alpha)$

Output: α', w_{k+1}

$\text{AccWitUpdate}(w_{k+1})$

No secrets needed

$\text{AccVerify}(\text{value}, \text{witness})$

= 1 if value in α , using witness

= 0 otherwise

Dynamic Accumulators

[ATSM09, ...]

- Note δ is secret per accumulator, known only to the Platform
- For a set $\{c_1, \dots, c_m\}$, the accumulating value $V = P_1 \prod_{i=1}^m (\delta + c_i)$
- For a credit c in the set, $(W = P_1 \prod_{i=1}^m (\delta + c_i) / (\delta + c), Q = VW^{-c})$ is the witness that c is accumulated in V
- When a new c' is accumulated, the new witness of c can be computed as $(W' = VW^{(c'-c)}, Q' = V'W'^{-c})$, where V' is the new accumulating value

Putting it all together



$\Sigma m, \Sigma r$



$w_u, w_d, auth$



$\{g^r h^m, E_K(r, m)\}, r, m, (w, q)\}$

$$acc_u = 1 \quad acc_d = \frac{1}{P} \frac{P^Y}{1} g^r h^m$$

$$acc_u = P^{(\delta + g^r h^m)}$$

Encash



Purchase



$w_d, auth$

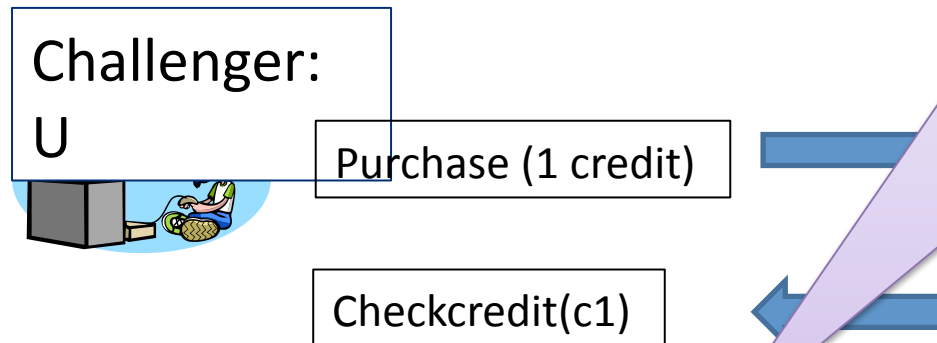


$\{g^r h^m, E_K(r, m)\}$

Spend

Properties: Transparency

- User Transparency
 - Modeled as a security game b/w



Proof: Prob of Adversary winning is non-negligible in security parameter

DL Assumption: Given h in G compute r st $h = g^r$

$$\Pr[\text{Adv wins}] = \Pr[\text{CheckCredit}(c1, m1) = 1 \text{ and } \text{CheckCredit}(c1, m1') = 1 \text{ and } m1 \neq m1']$$

$$= \Pr[c1 = g^{r1} h^{m1} \text{ and } c1 = g^{r1'} h^{m1'} \text{ and } m1 \neq m1']$$

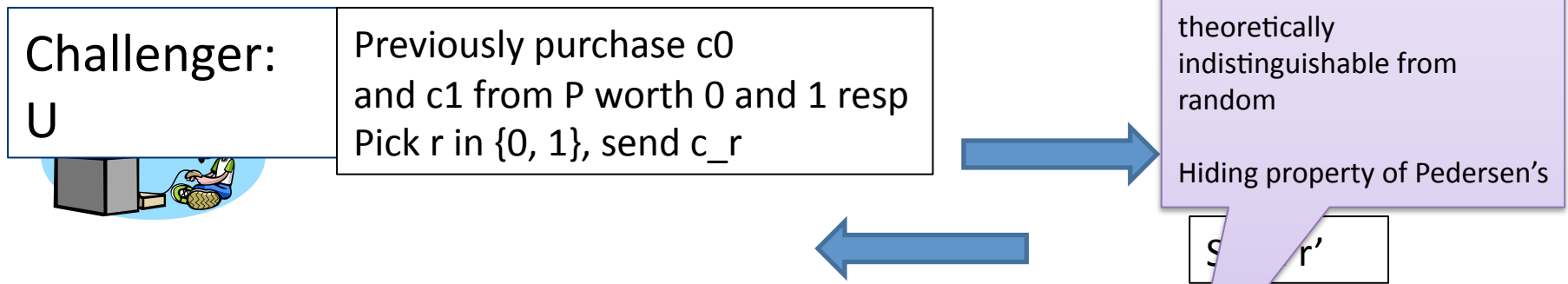
$$= \Pr[g^{r1} h^{m1} = g^{r1'} h^{m1'} \text{ and } m1 \neq m1']$$

Coming up with an $r1'$ that satisfies this is equivalent to finding the DLOG of $g^{r1} h^{m1-m1'}$, which is non-negligible

- Adversary wins attack game if Platform can produce $m1' \neq m1$ and $\text{CheckCredit}(c1, m1')$ is true.
- Similarly merchant transparency

Properties: User privacy

- Credits are opaque, P has no motivation to



- Adversary wins if D can correctly return $r' = r$ with prob $> 1/2$
- Other properties: double spending and encashment follow similarly

Fairness: Issues

- Merchant can attribute credit characteristics to users after encash
 - Policy: Enforce minimum number of tokens for each encash
 - Cannot resend credits across multiple encash transactions, as they are “used” up

Properties: Summary

Accountability: User transparency	Binding property of Pedersen's commitments	DLOG assumption
Accountability: Merchant transparency	Binding + homomorphic property of Pedersen's commitments	DLOG assumption
Security: Double spending	Dynamic accumulator scheme (both spending and encashment)	DDH and q-strong DH assumption
Security: Non repudiation	Signatures	Signature assumptions
Fairness	Hiding property of Pedersen's commitments	Information-theoretic

Flexibility (multiple currencies) and arbitrage prevention by design

Implementation and Deployment

- Performance
 - **Efficient**
 - Purchase (around 830 credits/sec, crypto dominates)
 - Spend /Encash (2X faster than purchase) for batches of 100
 - Verify most expensive
 - Witness updates can be batched
 - **Feasible**
 - Can generate around 71 million credits a day (26 B a year, around 100 B required)
- Can be incrementally deployed on FB

Admin Dashboard

Home

Credits

Users

Developers

More

Success! 100 Credits generated !

+ Generate Credits

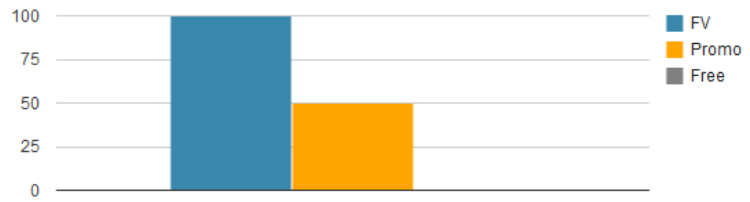
Full Value 0

Promo 0

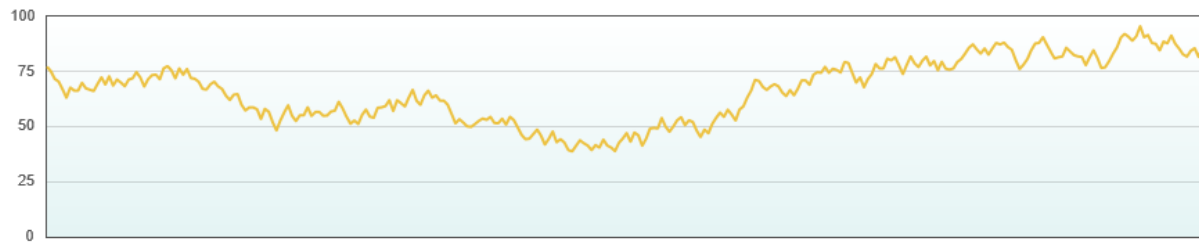
Free 0

Generate \$

Credit Stats



User Traffic



Unique Users per Second

Geographic distribution



Hi, Chetan

Home

Messages

Friends

Groups

More

+ Buy Credits

Full Value Promo Free

100 Buy \$

Credit History

100 Available












0 Spent

Success! Game purchased !

Games



Conclusions

	MS Points (W)	FB Credits (C)	Linden \$\$	Bitcoin	Verito
Security					
Transparency	Fixed		Fluctuates	Fluctuates	
Flexibility	Tied to USD	Tied to USD	Tied to exchange	Tied to exchange	
Fairness					
Performance				Expensive	

- Transparency and Fairness are vital requirements in a virtual economy
 - Collusions?
 - Simpler? More efficient?