HOW TO SHOP FOR FREE ONLINE – SECURITY ANALYSIS OF CASHIER-AS-A-SERVICE BASED WEB STORES

Rui Wang (Indiana Univ.)

Joint work with
Shuo Chen (MSR), XiaoFeng Wang (Indiana Univ.), Shaz Qadeer (MSR)

Free goodies

- Random items bought from web stores

Alcohol Tester

Power Strip

DVD

Agility Cream

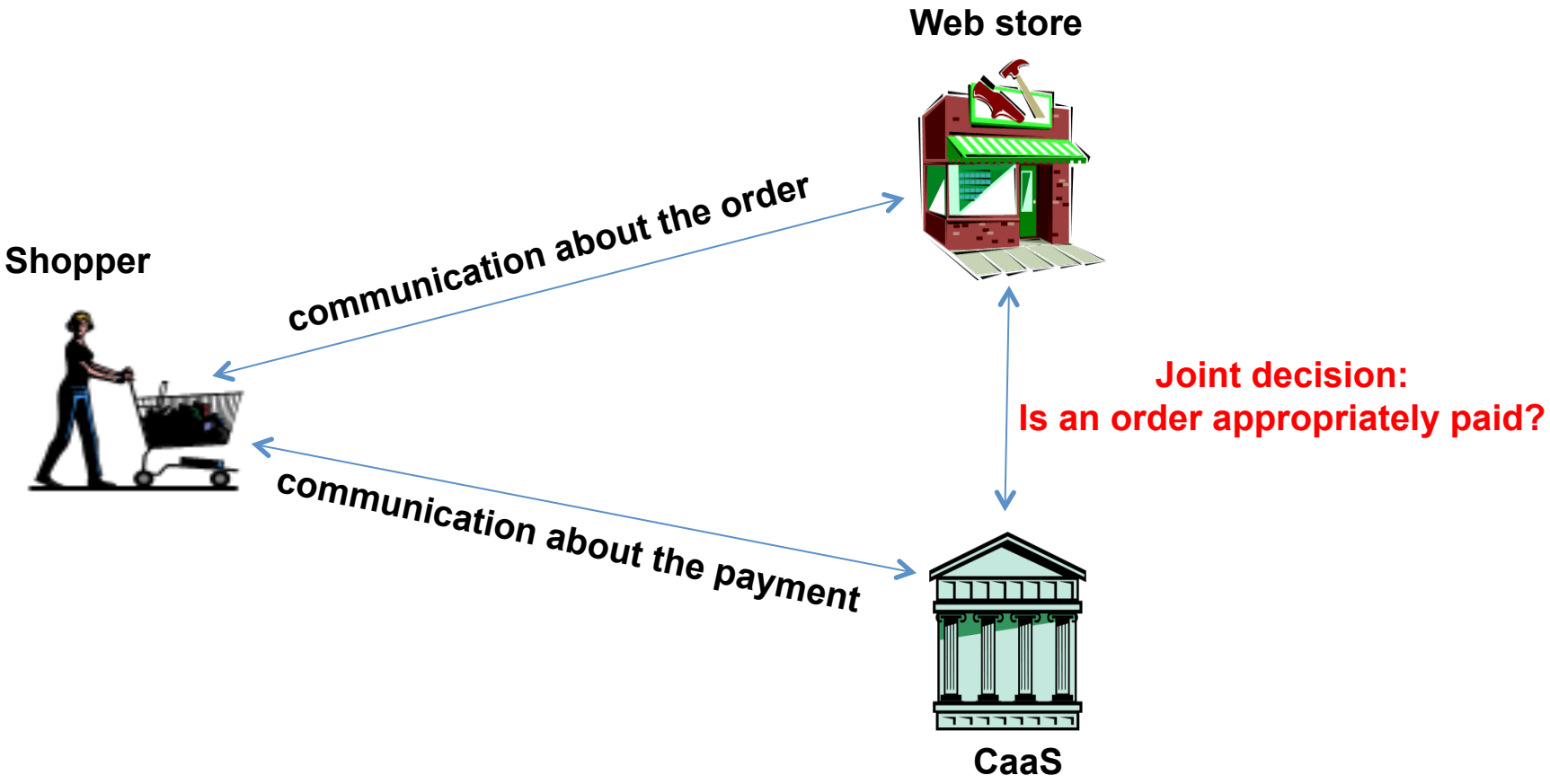
Digital Magazine

- Did not pay, or with an arbitrary price
- Due to logic bugs in checkout mechanisms

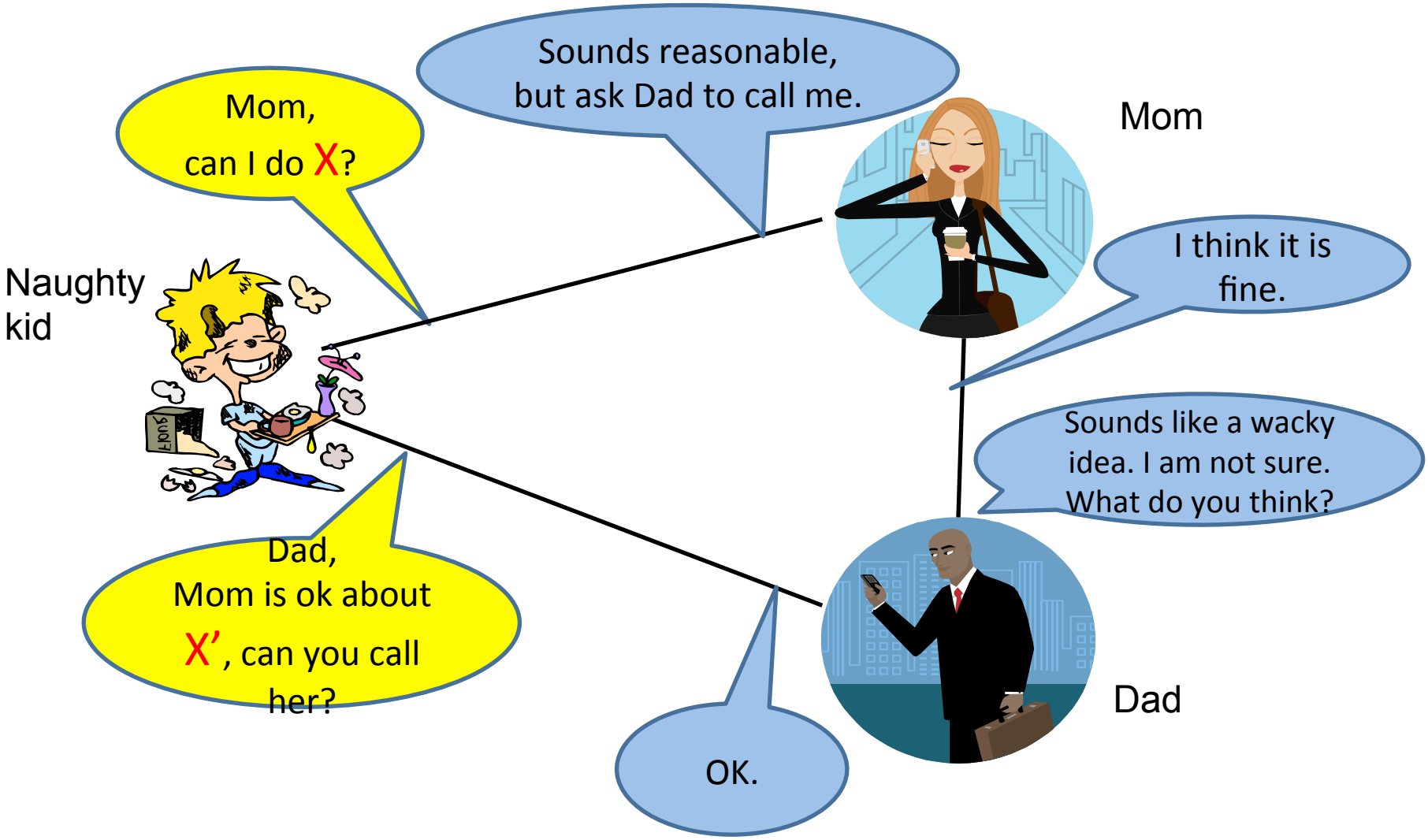
Web stores integrating 3rd party cashier services

- 3rd-party cashiers
  - e.g., PayPal, Amazon Payments, Google Checkout
  - We call them CaaS (Cashier-as-a-Service)
    - The CaaS exposes services through web APIs
    - Web stores call APIs to integrate services
  - A great number of stores use CaaS services.

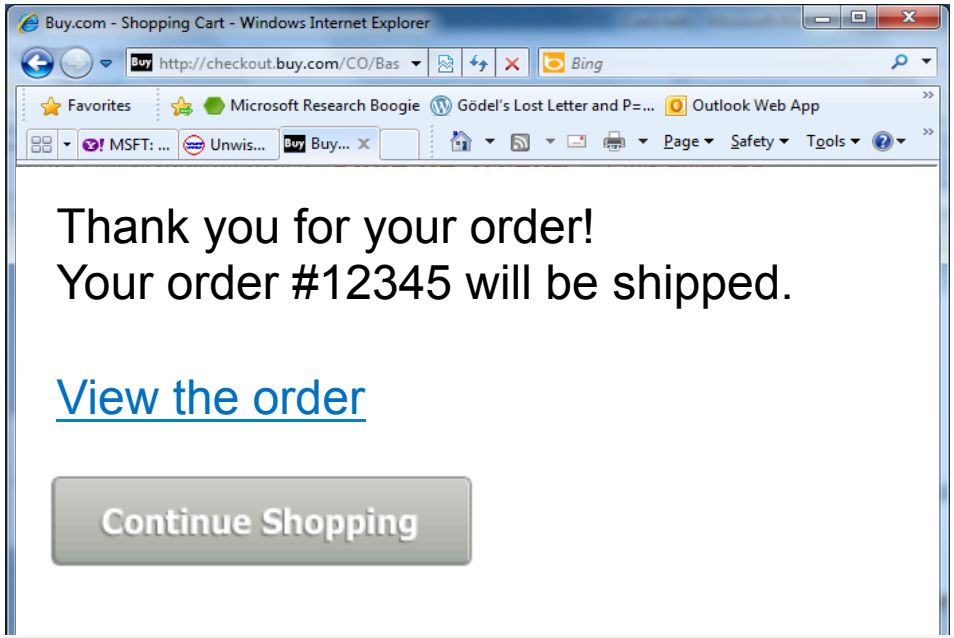
Need to make a joint decision



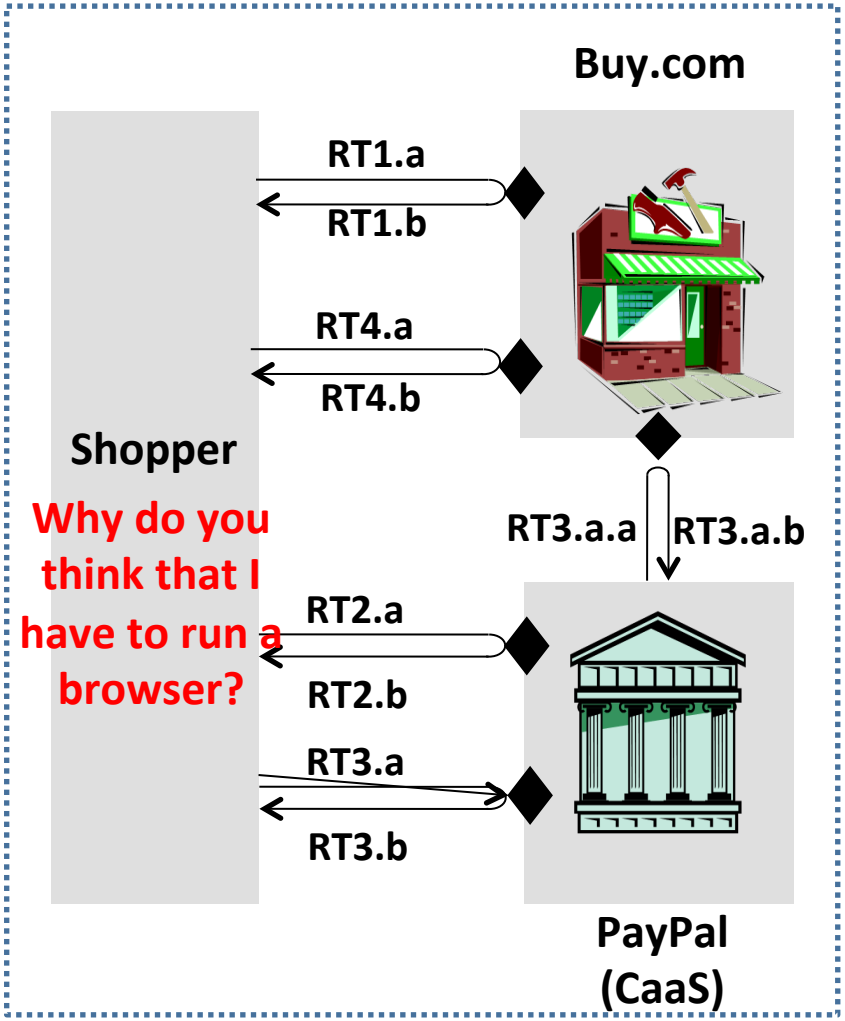
Why challenging, intuitively?



Example of a normal checkout workflow



- **There are many payment methods, such as PayPal Standard, Amazon Simple Pay, Google Checkout**
- **Even for one payment method, each store integrates it in a different way**



RT: HTTP round-trip ◆ **:** Web API

What we studied

- Merchant software – with source code
  - Used to build web stores
    - **NopCommerce** – popular open-source
    - **Interspire** – ranked #1 by Top10Reviews.com
    - **Amazon SDKs** – used by stores to integrate Amazon Payments
- High-profile web stores – no source code
  - **JR.com**
    - A store for consumer electronics since 1971
  - **Buy.com**
    - 12 million shoppers

- What do the seller and charger need to verify:
  - Seller owns the item
  - A payment will be transferred to seller from charger
  - The payment is for the right amount
  - The payment is for the right item

- Why is it so complicated
  - Whose responsibility to verify the information
    - This transaction number is correct, but is it for my store?
  - The attacker can pretend to be a buyer as well as a seller
  - Many parallel transactions
  - The APIs are public and the attackers can analyze them as long as they want

Results

- Logic flaws in 9 checkout scenarios

Explained in this talk

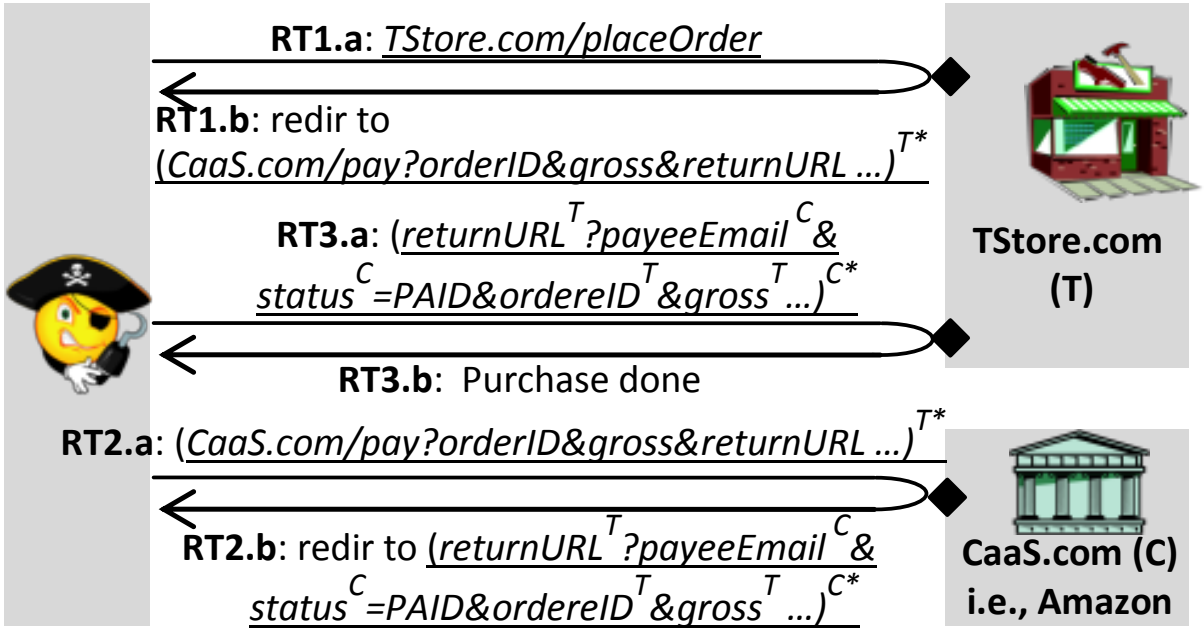
Merchant	CaaS	Flaw	Result
NopCommerce	PayPal Standard	Insufficient check of payment total	Pay arbitrary price
NopCommerce	Amazon Simple Pay	Insufficient protection against a shopper with a malicious merchant	Shop for free
Interspire	Amazon Simple Pay	Incorrect use of signature	Shop for free
Interspire	PayPal Express	Insufficient protection against a shopper with two shopping sessions	Pay arbitrary price
Interspire	PayPal Standard	Payment notification can be replayed under certain condition	Pay arbitrary price
Interspire	Google Checkout	Can add items to cart after payment total is fixed	Pay arbitrary price
JR.com	Checkout By Amazon	Insufficient protection against a shopper with a malicious merchant	Pay arbitrary price
Buy.com	PayPal Express	Paypal token allowed to be reused	Pay arbitrary price
Web stores using Amazon SDKs	Amazon Flexible Payments	Insufficient signature validation	Shop for free

Three Flaw Examples

Note:

1. Only high-level summaries, not full picture of the flaws
2. Details in the source code are critical, but skipped
3. Please read the paper for the whole stories

NopCommerce's integration of Amazon Simple Pay



TStore.com/placeOrder: orderID=InsertPendingOrder ()

TStore.com/finishOrder **(handler of RT3.a)**:

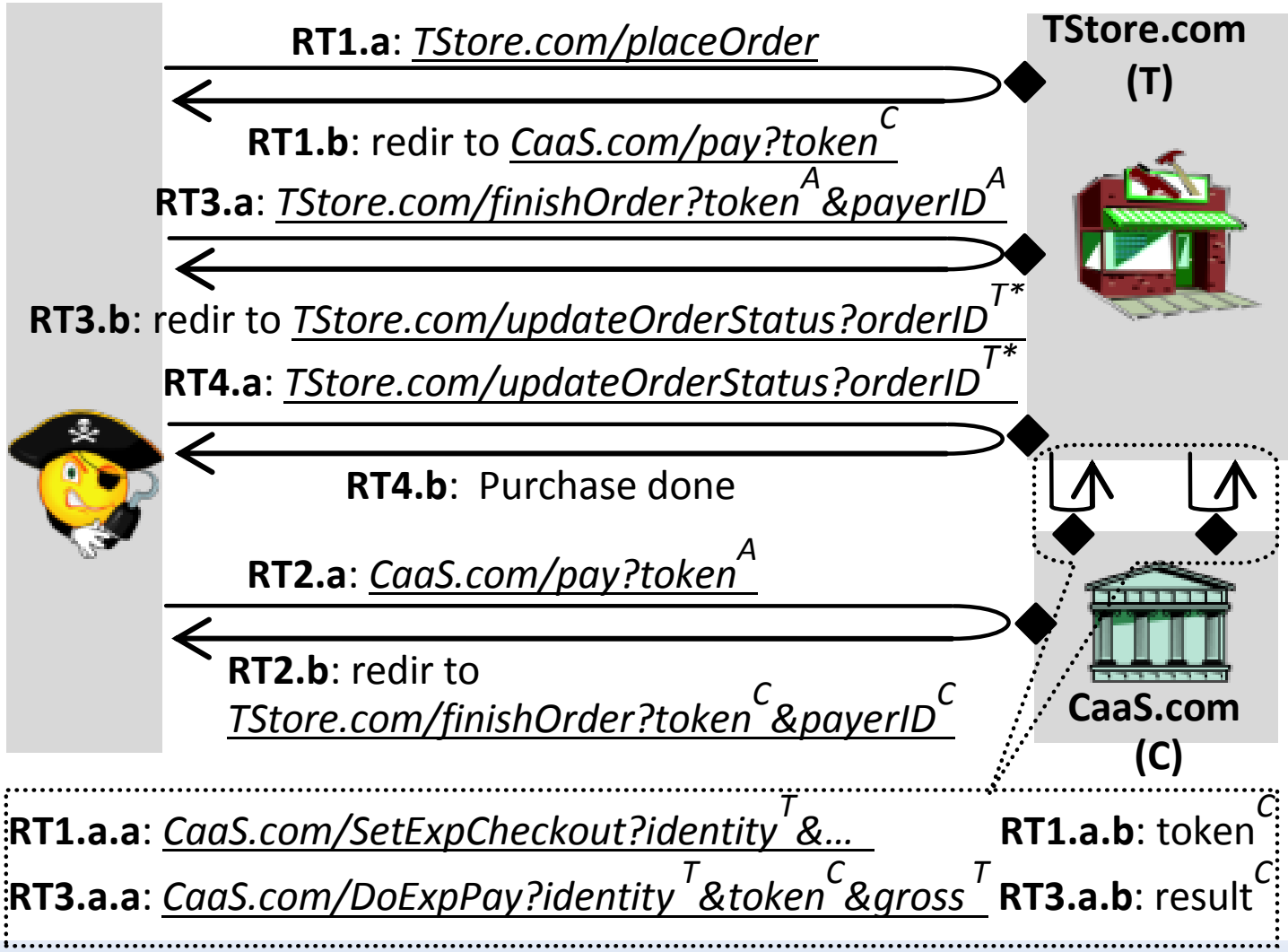
```
if (verifySignature(RT3.a) ≠ CaaS) exit;
if (GetMsgField("status") ≠ PAID) exit;   /*payment status*/
order= GetOrderByID(ordereID);
if (order==NULL or order.status ≠ PENDING) exit;
order.status=PAID;
```

Flaw & exploit

- Anyone can register an Amazon seller account, so can Chuck.
  - We purchased a $25 MasterCard gift card by cash
  - We registered it under the name "Mark Smith" with fake address/ phone number
  - Registered for seller accounts in PayPal, Amazon and Google using the card

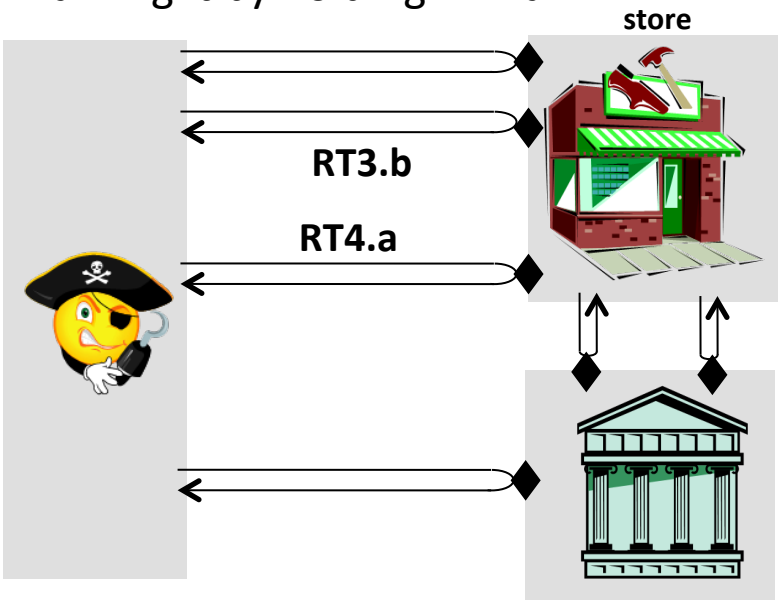
- Chuck's trick
  - Pay to Mark (i.e., Chuck himself), but check out from Jeff
  - Amazon is tricked to tell Jeff a payment between Chuck and Mark
  - Jeff is confused by Amazon

Interspire's integration of PayPal Express



Interspire's integration of PayPal Express (cont.)

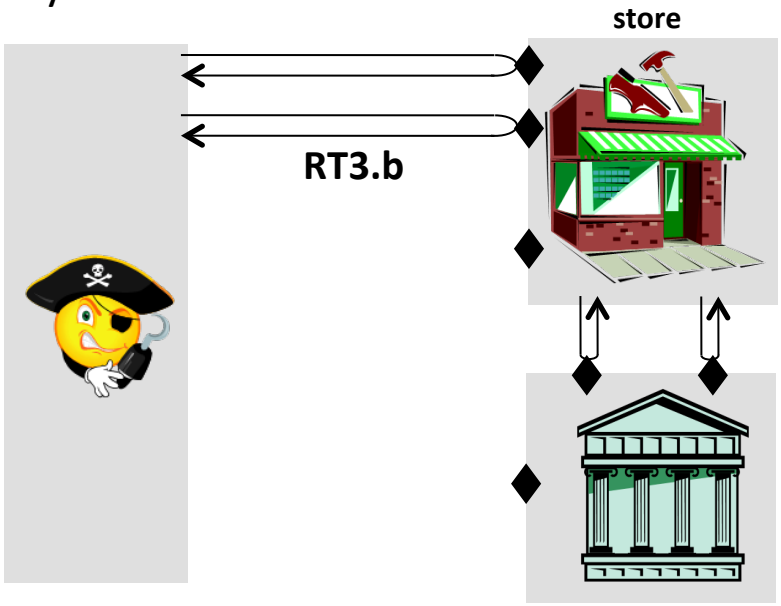
Session1: pay for a cheap order (orderID1) in PayPal, but avoid the merchant from finalizing it by holding RT4.a



(RT3.b) redir to store.com/finalizeOrder?$[orderID1]_{store}$

(RT4.a) call store.com/finalizeOrder?$[orderID1]_{store}$

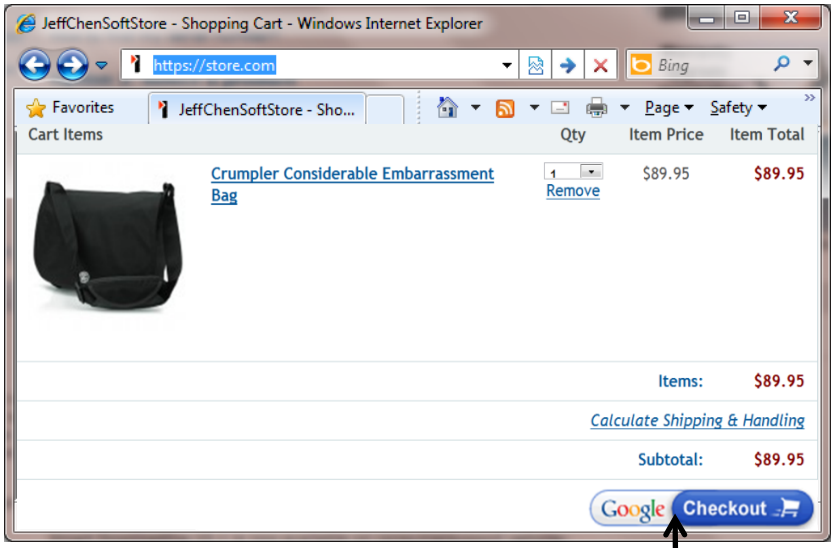
Session 2: place an expensive order (orderID2) , but skip the payment step in PayPal

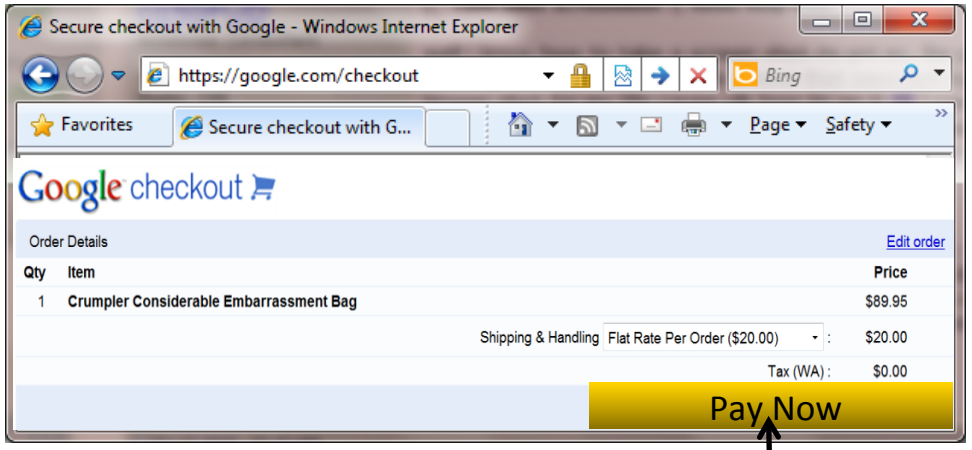


(RT3.b) redir to store.com/finalizeOrder?$[orderID2]_{store}$

- Expensive order is checked out but the cheap one is paid

Interspire's integration of Google Checkout





Payment total is calculated based on cart.

Order is calculated based on cart.

time

Oops! Cart is not locked.

Confirming the Presence of These Flaws in Real World

Our systematic validation

- Against stores on our own web server
- Against our store on Interspire's popular hosting service
  - BigCommerce
- Against real stores powered by NopCommerce and Interspire
  - GoodEmotionsDVD.com, PrideNutrition.com, LinuxJournalStore.com
- Similar attacks against stores running closed-source software, e.g., Buy.com and JR.com
  - Without source code access, some exploit ideas are still applicable

Responsible experiments

- Under close guidance of an Indiana University lawyer.
- Support from Dean of School of Informatics
- Principles
  - No intrusion
  - No monetary loss to the stores
  - Communicated full details to affected parties
- Pleasant outcome
  - No negative opinions on our tests, responsible efforts appreciated by most of them
  - News articles are all positive

How hard to detect the attack?

Dear buy.com customer service,

I am a Ph.D. student doing research on e-commerce security. I bumped into an unexpected technical issue in buy.com's mechanism for accepting the paypal payments. I appreciate if you can forward this email to your engineering team.

After our refund–eligible period, we mailed the products back by a certified mail. We disclosed technical details to them.

Re: Other

(KMM3545

Buy.Com Support <customerhelp@noreply.buy.com> Wed, Jun 16, 2010 at 6:25 PM

To: Test Wang <ruiwangworm@gmail.com>

Hello Test,

Thank you for contacting us at Buy.com.

Based on our records you were billed on 6/10/2010 for $5.99. To confirm your billing information please contact PayPal at https://www.paypal.com/helpcenter or at 1-402-935-2050.

Companies are very serious about these bugs

- They were very responsive
  - Most emails were replied
- All 9 bugs have been quickly fixed
  - Amazon SDK vulnerability
    - 15 days after our reporting, Amazon released a new set of SDKs for all supported languages and a security advisory, crediting Rui Wang
    - 40 days after the advisory, Amazon disabled the support of vulnerable SDKs, forcing all stores to upgrade to the new version

Also in the paper

- Complexity of CaaS-based checkout logic
- Attacker Anonymity
  - Attacks can happen without disclosing the attacker's identity

Conclusions

- Multi-party web apps fundamentally more complicated than traditional web apps
  - Confusion in coordination
  - Concurrency and atomicity
  - Weak bindings among data fields
  - Adversary playing multiple roles
- CaaS-based stores are under imminent threats
  - Shown by real purchases.
- The issue is not specific to cashier service integration
  - It has a broader domain: web service integration
    - Social Network, e.g., Facebook, LinkedIn
    - 3rd Authentication, e.g., Google, Yahoo, Twitter

- The real challenge that I see in system security in general

Some thoughts on solution

- Security-conscious programming guides
- Certified Integration
- Verification/Testing tools