CS 3525: Advanced Topics in Security and Privacy

Paper Discussing

Presenter: Huy Viet Nguyen

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

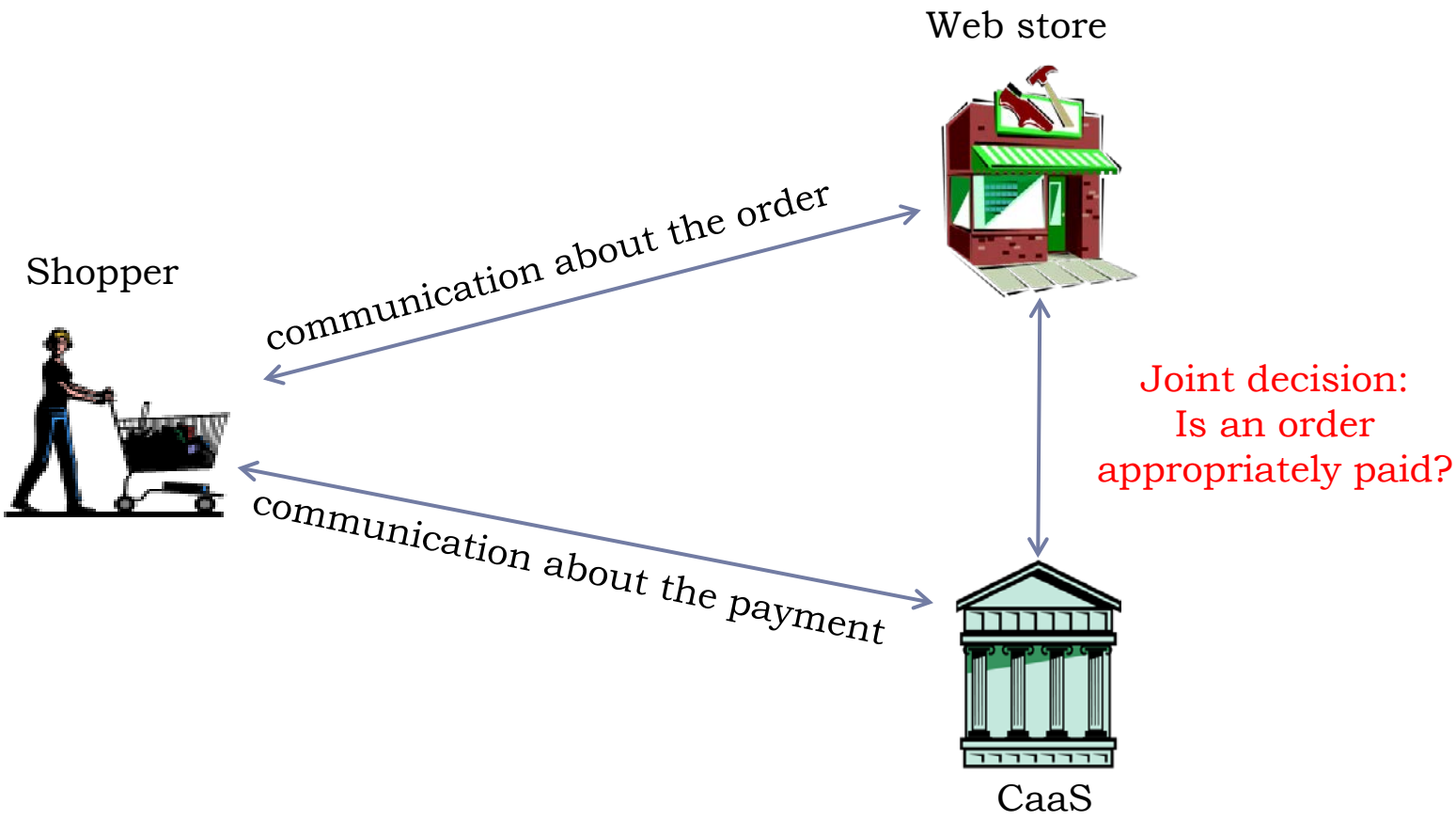
Rui Wang[1], Shuo Chen[2], XiaoFeng Wang[1], Shaz Qadeer[2]

[1] Indiana University, [2] Microsoft Research

CaaS-based web-store

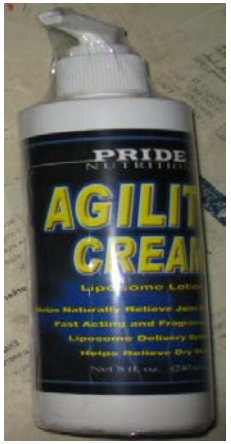
- 3rd party cashiers
- Cashier-as-a-Service (CaaS)
  - The CaaS exposes service through Web APIs
  - Web-store call APIs to integrate services
- Studied CaaS and web-store

Trilateral interaction



Some items checked out through flaws

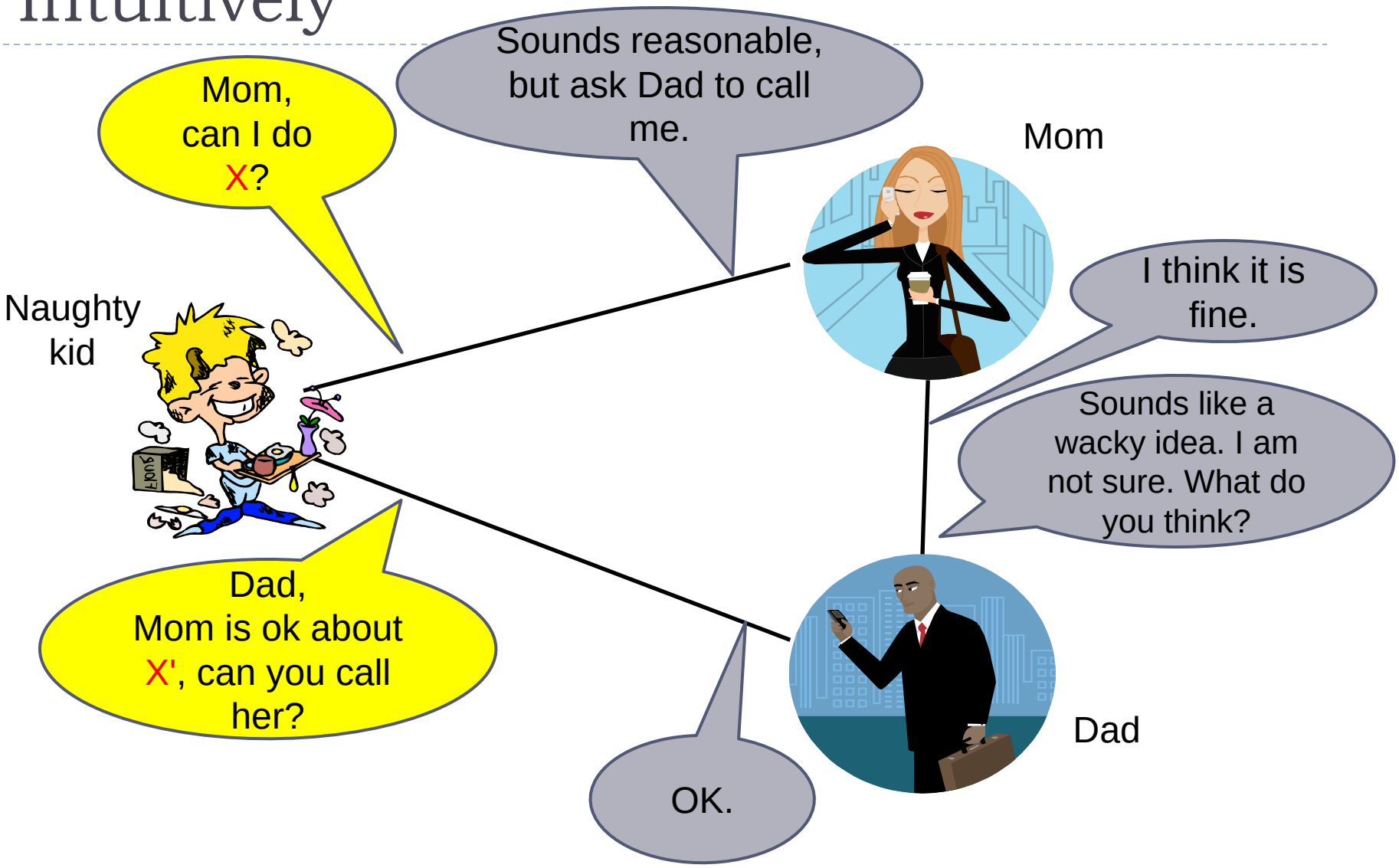
- Reveal security-related flaws in merchant systems that use cashier services
- Attack model is fairly simple



Challenges of checkout process

- Confusion in coordination
  - Incomplete views of transaction of merchant and CaaS
- Diversity in the adversary's roles
  - APIs are exposed public so adversary can gain deeper involvement in checkout process
- Parallel and concurrent services
  - Many customer, multiple purchase transaction
- Authentication and data integrity
  - Binding of fields in different messages

Intuitively

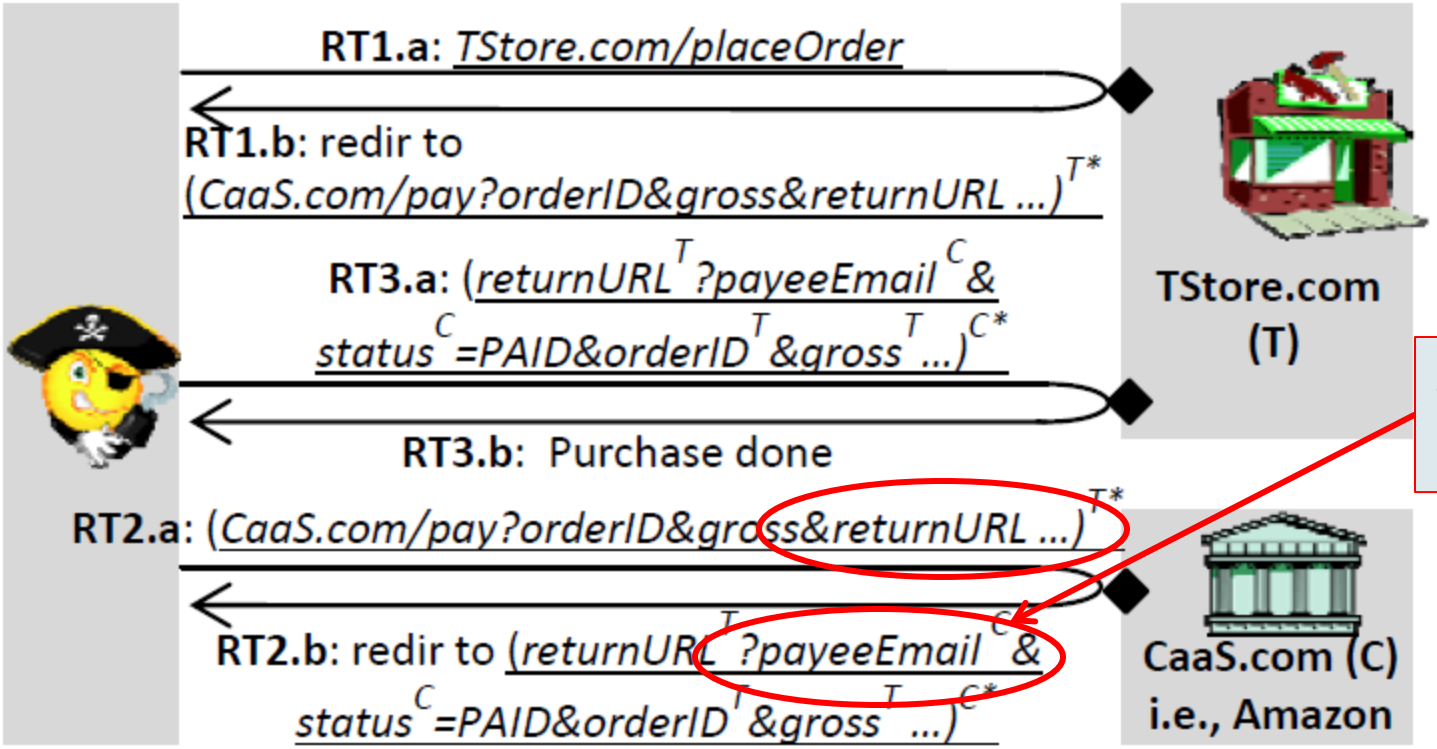


NopCommerce with PayPal Standard

- API *finishOrder* to finalize the invoice, check only the *orderID*
- Freely modify the gross when "talk" with CaaS

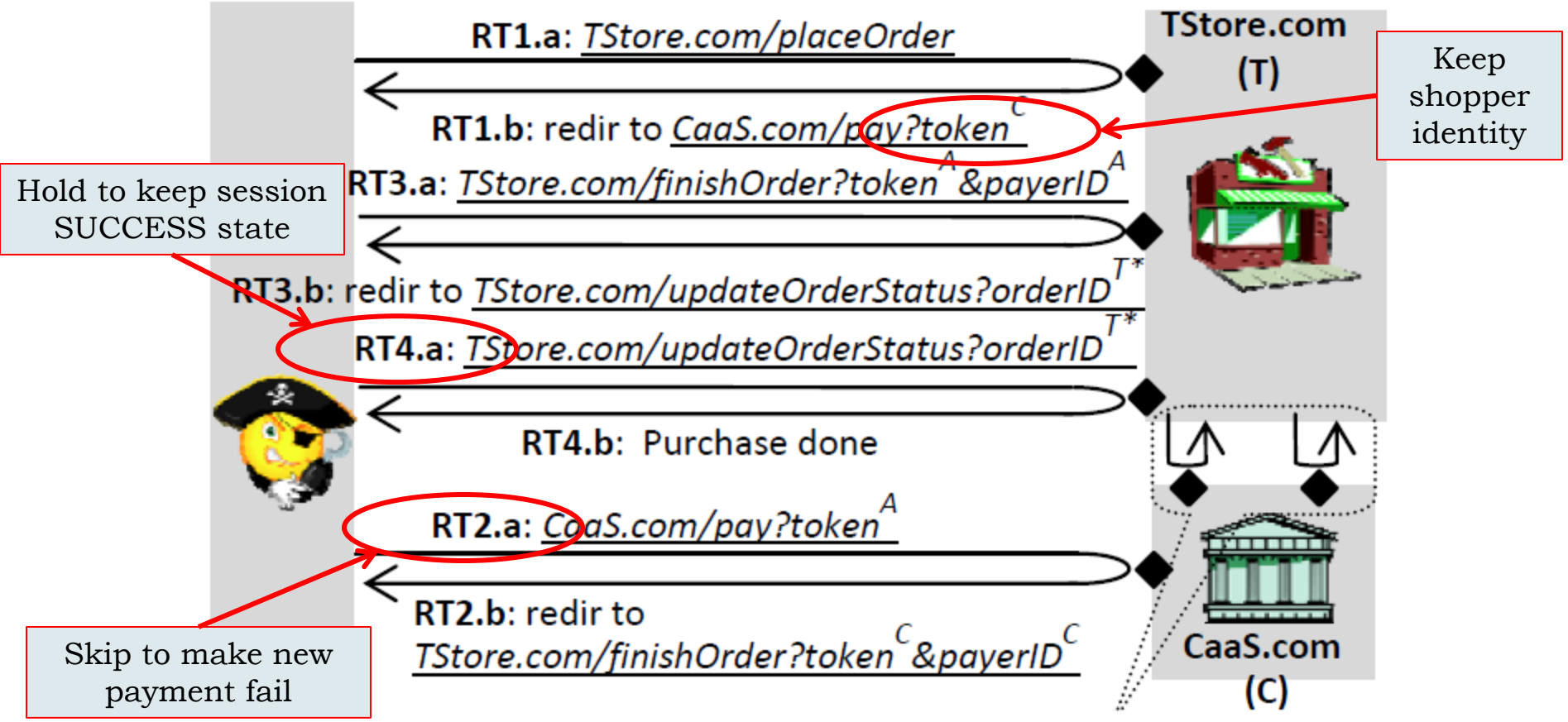
NopCommerce with Amazon SimplePay

- Shop in one store but pay to a malicious store by faking the message to CaaS with attacker signature



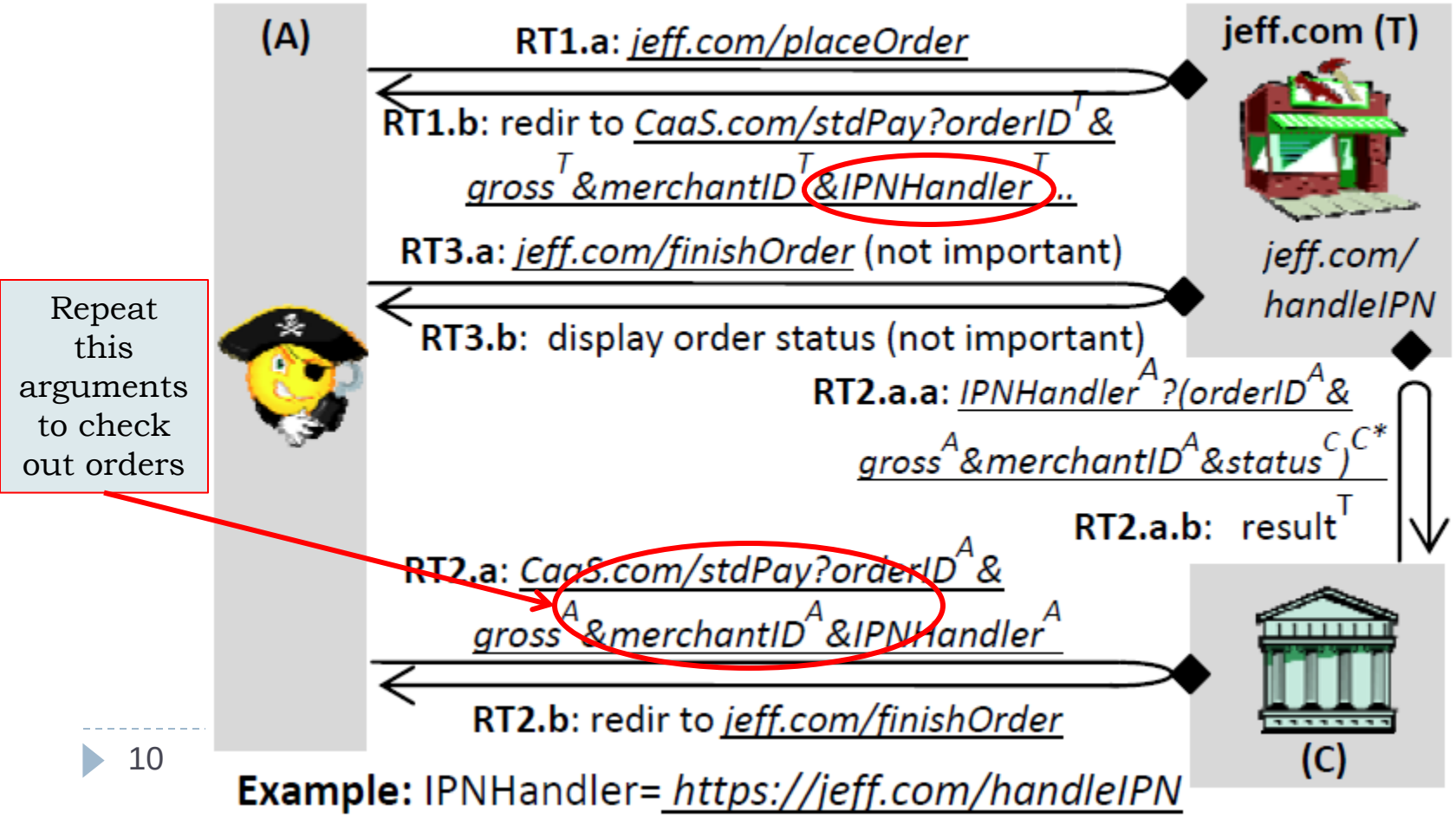
Interspire with PayPal Express

- *updateOrderStatus* based on session state
- Keep old session *SUCCESS* state to check out new items

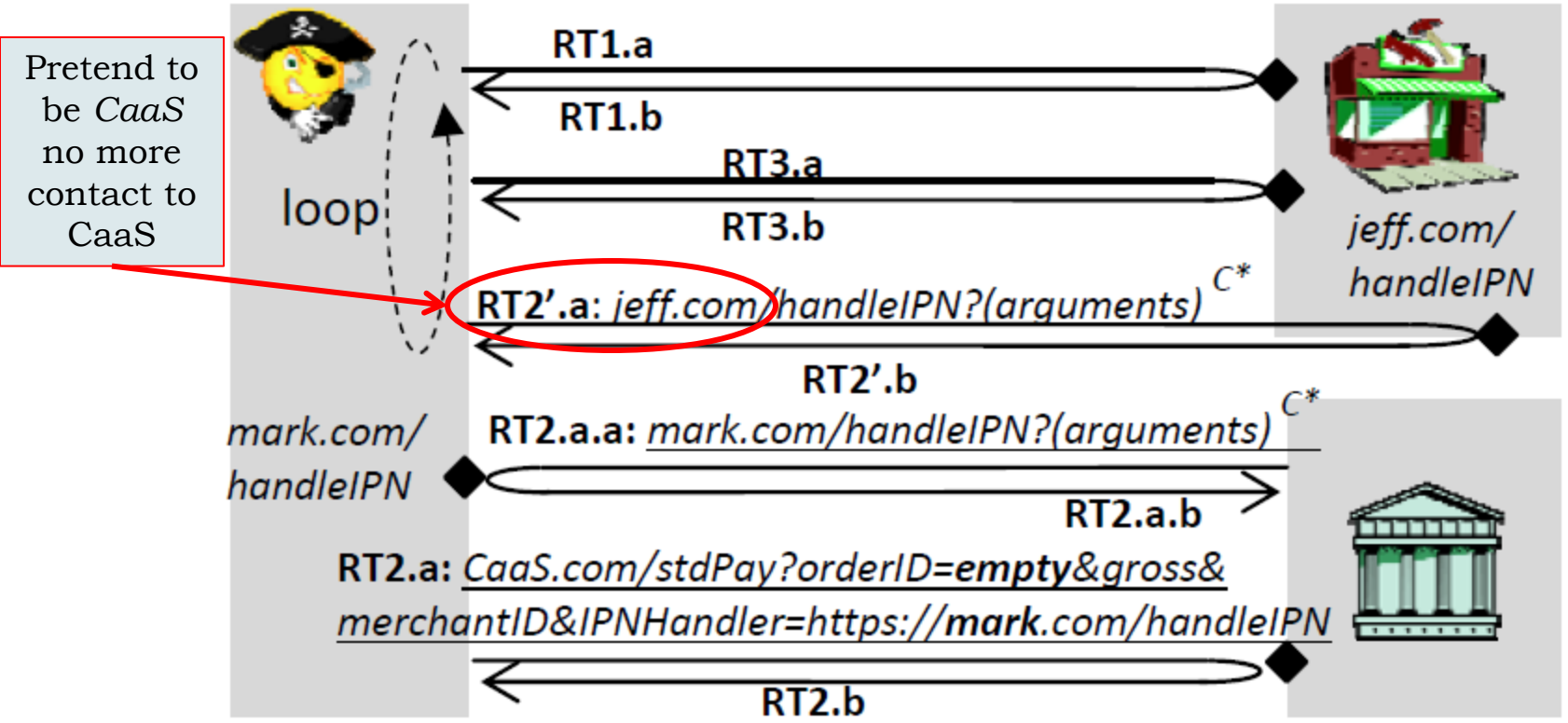


Interspire with PayPal Standard

- CaaS sends IPN to merchant right after shopper makes payment using URL in *IPNHandler*
- Replay IPN message to check out orders

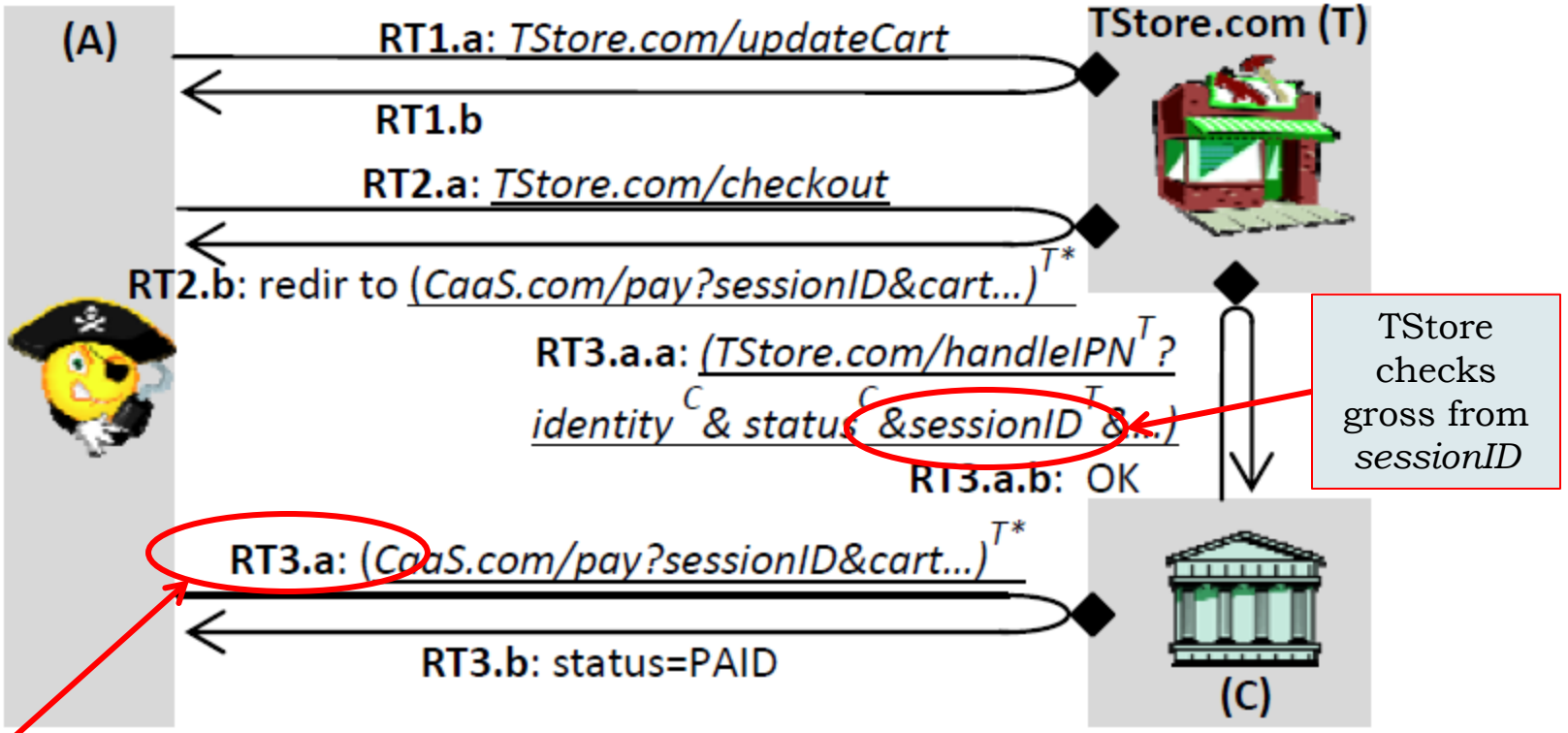


Interspire with PayPal Standard (cont.)



Interspire with Google Checkout

- Adding item into cart after checkout button is clicked



Amazon Payment SDK flaw

- Amazon Payments offers a signature verification API
- URLs signed by Amazon Payments,

(https://merchant/someAPI?arg1&arg2&…&argN&**certificateURL=https://fps.amazonaws.com/certs/090909/PKICert.pem**)[C*]

- If the message is signed by Amazon, then the certificate is an Amazon certificate, and vice versa
- Only verify the signature using the certificate referenced by certificateURL, without verifying the certificate itself

(https://merchant/someAPI?arg1&arg2&…&argN&**certificateURL=https://cert.foo.com/PKICert.pem**)[A*]

Systematic validation

- Store on authors' server
- Against authors' store on Interspire's popular hosting service i.e. BigCommerce
- Against real stores powered by NopCommerce and Interspire
- Similar attacks against stores running closed-source software

Responsible experiments

- No intrusion
- No monetary loss to the stores
- Communicated full details to affected parties
- No negative opinions on the tests,
- Responsible research efforts were appreciated by most of the organizations

Also in the paper

- Attacker anonymity
  - Tor, Anonymizer
- Complexity analysis of checkout logic
  - Subset of Interspire's logic
  - Express in C language
  - Verification condition based on payment-completion invariant
  - Do theorem proving

Conclusions

- Multi-party web apps are fundamentally more complicated than traditional web apps
- CaaS-based stores are under imminent threats
- 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

Discussion Question

- **1.** One of the root problems of CaaS-based web-store is the intermediate role of shopper when web store and CaaS allow shopper contact to each of two parties to setup agreement. In principle, this scheme exhibits man-in-the-middle attack as presented in the paper. Why do you think it's important for online payment service to keep intermediate role of shopper? In other words, can we eliminate such role of shopper and allow more direct contact between CaaS and web-store to setup agreement?
- **2.** The paper has listed many possible attacks based on logic flaws exploited in the CaaS-based web-store. However it seems that those flaws can be fixed very quickly and simply. Does this make you think the flaws mentioned in the store web applications can solve the problem.
- **3.** While authors did mentioned some possible solutions but those are all too abstract. Could you think of any more practical approach to handle the problem? Try to make it as concrete as possible. In other hands, you can evaluate the significance of the problem by asking the question could current system can detect such kind of cheat of shopper and recover to avoid damage.
- **4.** The authors generalized the problem to other fields of internet security like social network (e.g. Facebook, Linkedln)and web authentication service - so called 3rd authentication (e.g. Google, Yahoo). Could you give example how these web service integrations could encounter the similar problems as those mentioned in the paper.