Locating Prefix Hijackers using LOCK

Tongqing Qiu[+], Lusheng Ji[*], Dan Pei[*]

Jia Wang[*], Jun (Jim) Xu[+], Hitesh Ballani[++]

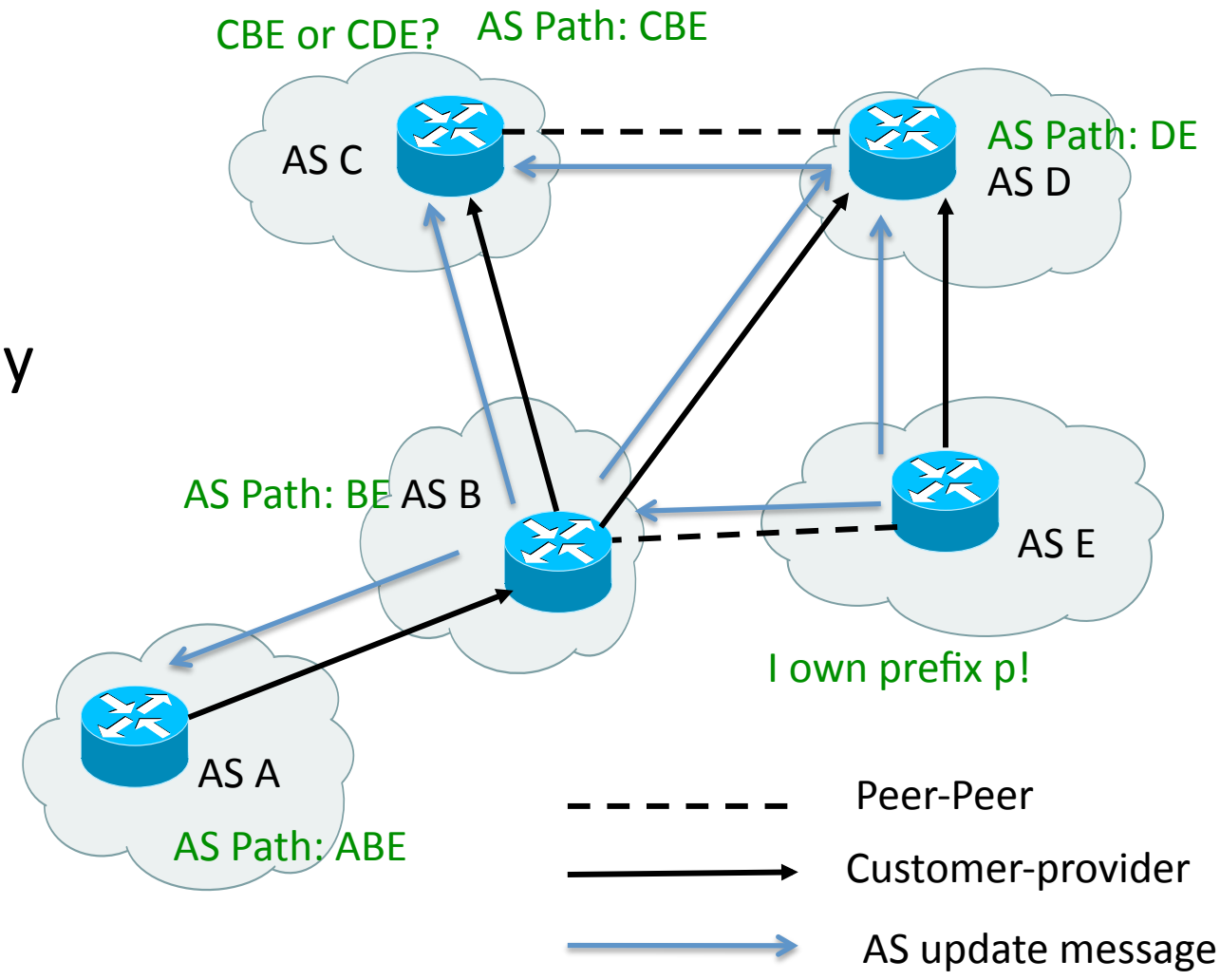
+ College of Computing, Georgia Tech
* AT&T Lab – Research
++ Department of Computer Science, Cornell University

Outline

- Background & Motivation
- System Architecture
- Basic algorithm and improvements
- Evaluation
- Conclusion

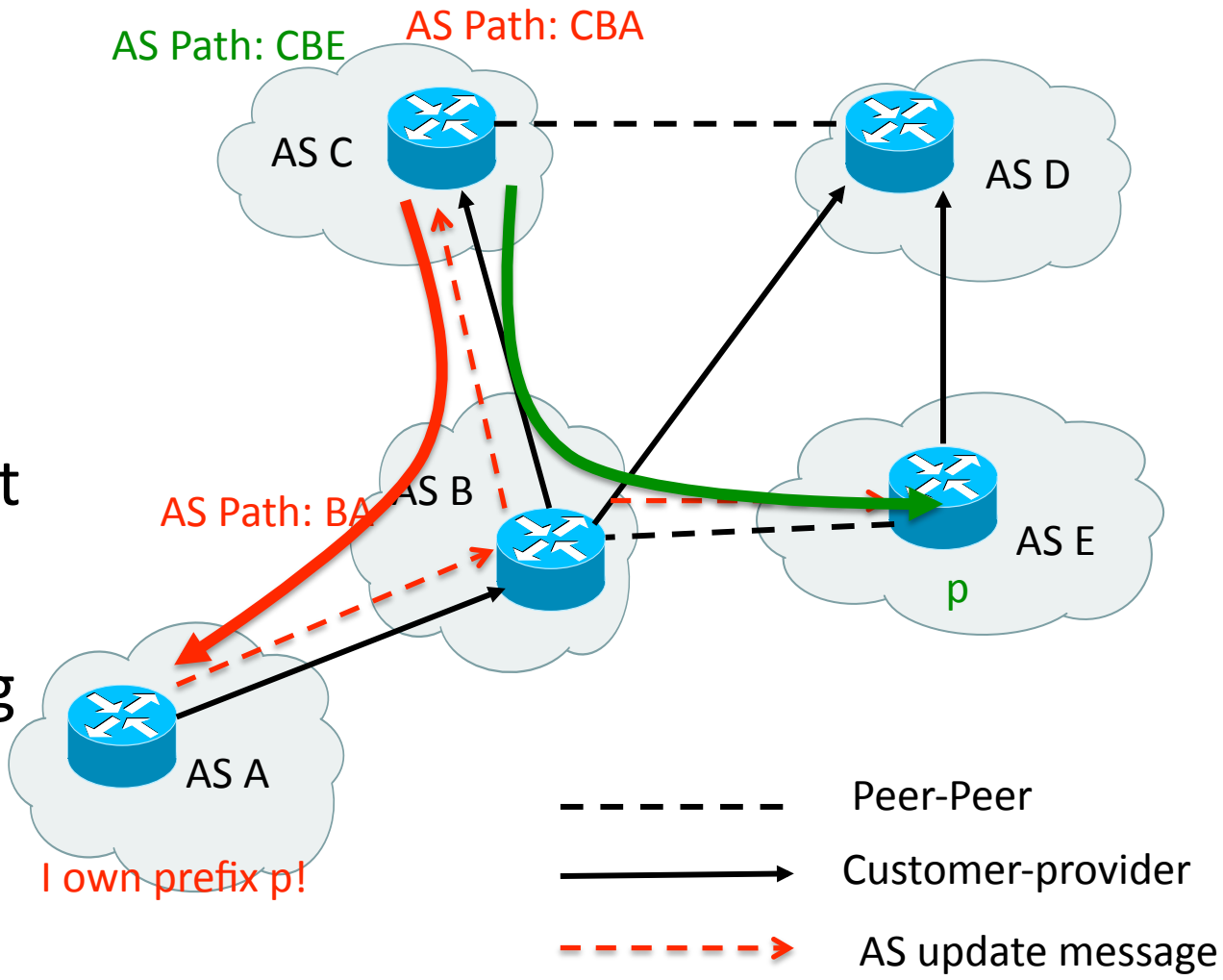
Background

- Autonomous System (AS)
- Border Gateway Protocol (BGP)
- Profit-driven policy



Background (cont.)

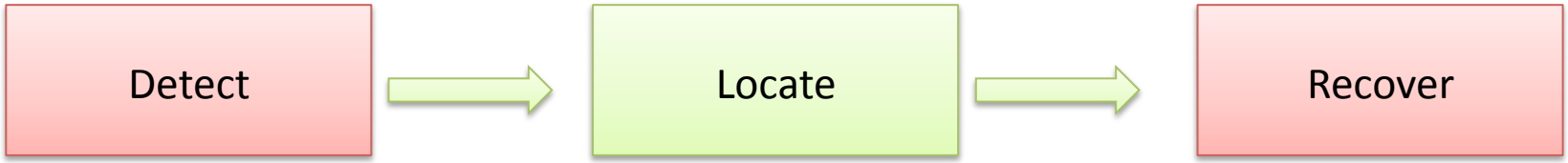
- BGP lacks authentication
- Fabricated AS announcement
- Prefix hijacking
  - blackholing
  - imposture
  - interception



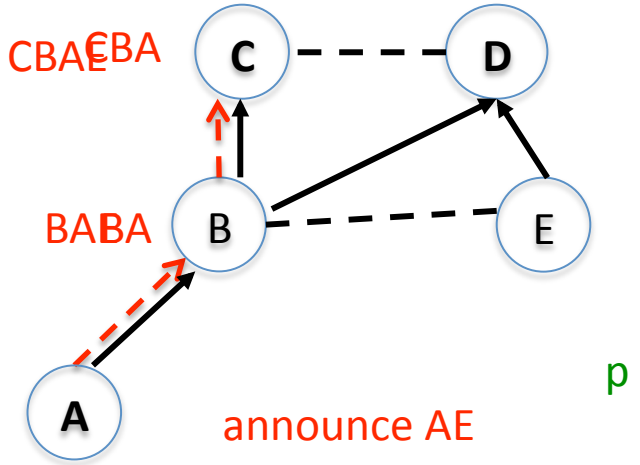
State of Art

- Proactive
  - Prevent the happenings of hijacks
    - e.g. [Kent et al. JSAC 00] [Aiello *et al.* CCS 03], [Subramanian *et al.* NSDI 04], [Karlin *et al.* ICNP 06], etc.
  - Deployment issues:
    - Routing infrastructure modification
    - Difficulties of incremental deployment
    - PKI requirement
- Reactive
  - Detection
    - e.g. [Lad et al. Usenix Secuirty 06], [Ballani *et al.* Sigcomm 07], [Zheng *et al.* Sigcomm 07], [Hu *et al.* IEEE S&P 07], [Zhang *et al.* Sigcomm *08*], etc.
  - Recovery
    - e.g. [Zhang *et al.* CoNext 07]

A Complete and Automated Solution?

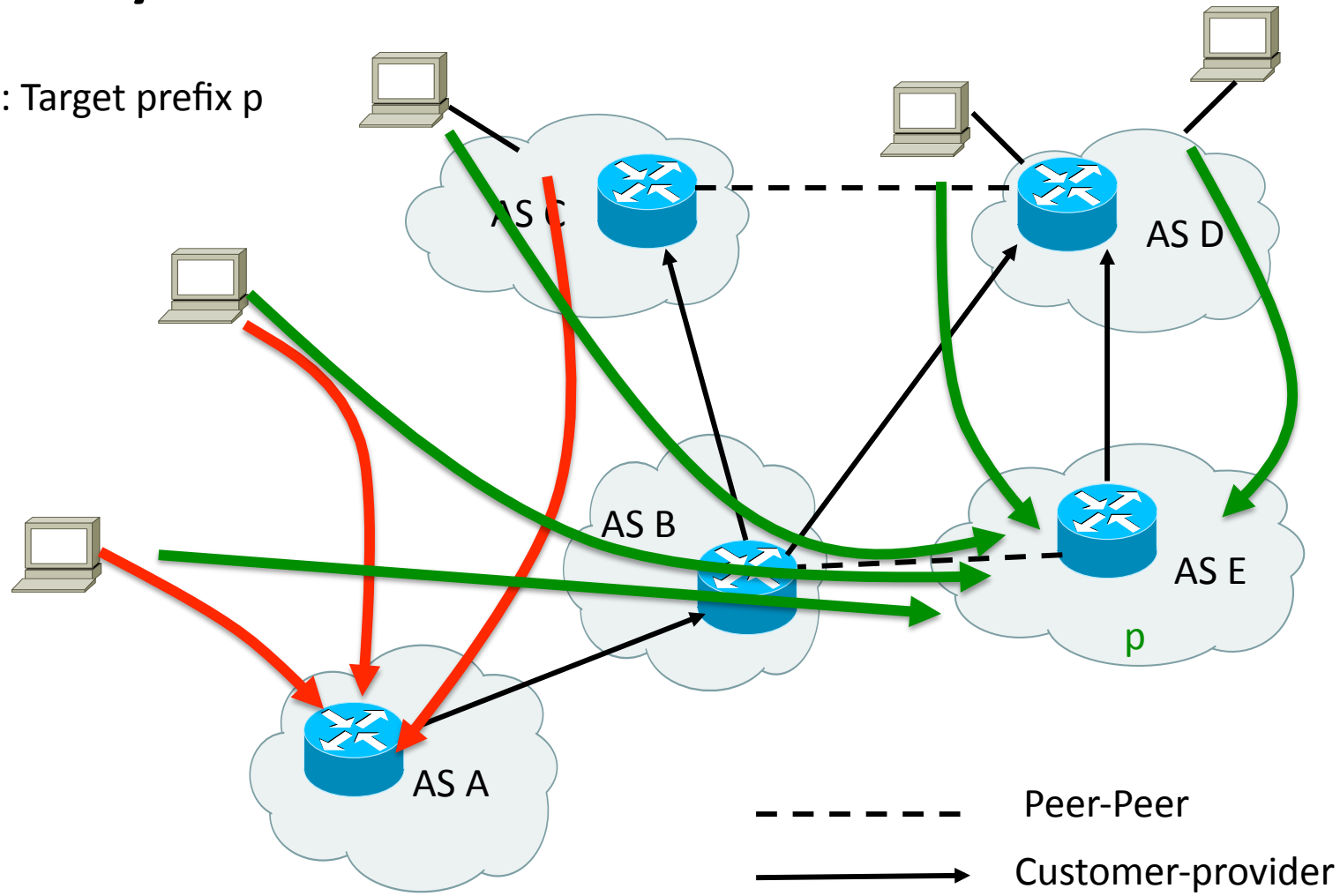


- Locating is important
  - Provide key information for recovery/mitigation
- Locating is not trivial
  - Current practice
    - Indentify newly appeared origin AS of prefix *p*



System Architecture of LOCK

Input: Target prefix p



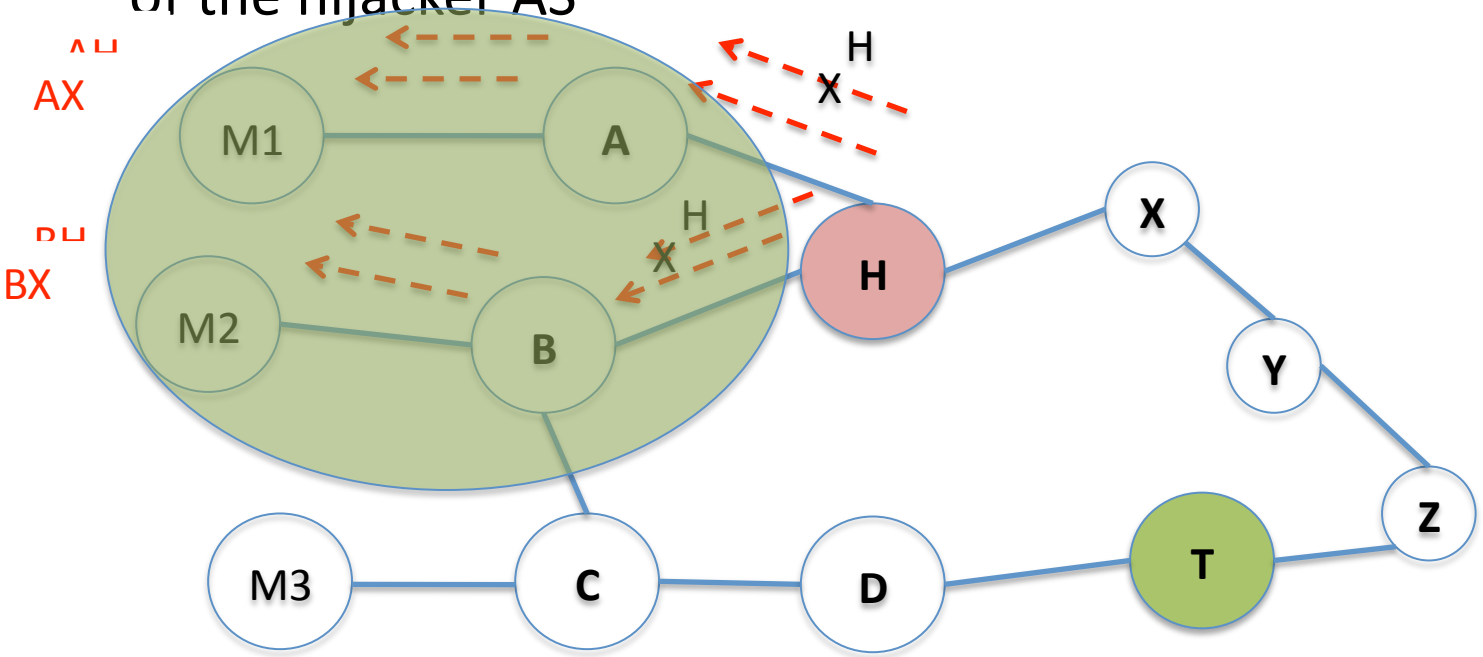
Output: A is the hijacker!

Key Components of LOCK

- Monitor Selection (from candidates)
  - Maximize the likelihood of observing hijacking events on the target prefix
  - Maximize the diversity of paths from monitors to the target prefix
- Locating Scheme
  - Using AS path information
  - Infer the hijacker location (how?)

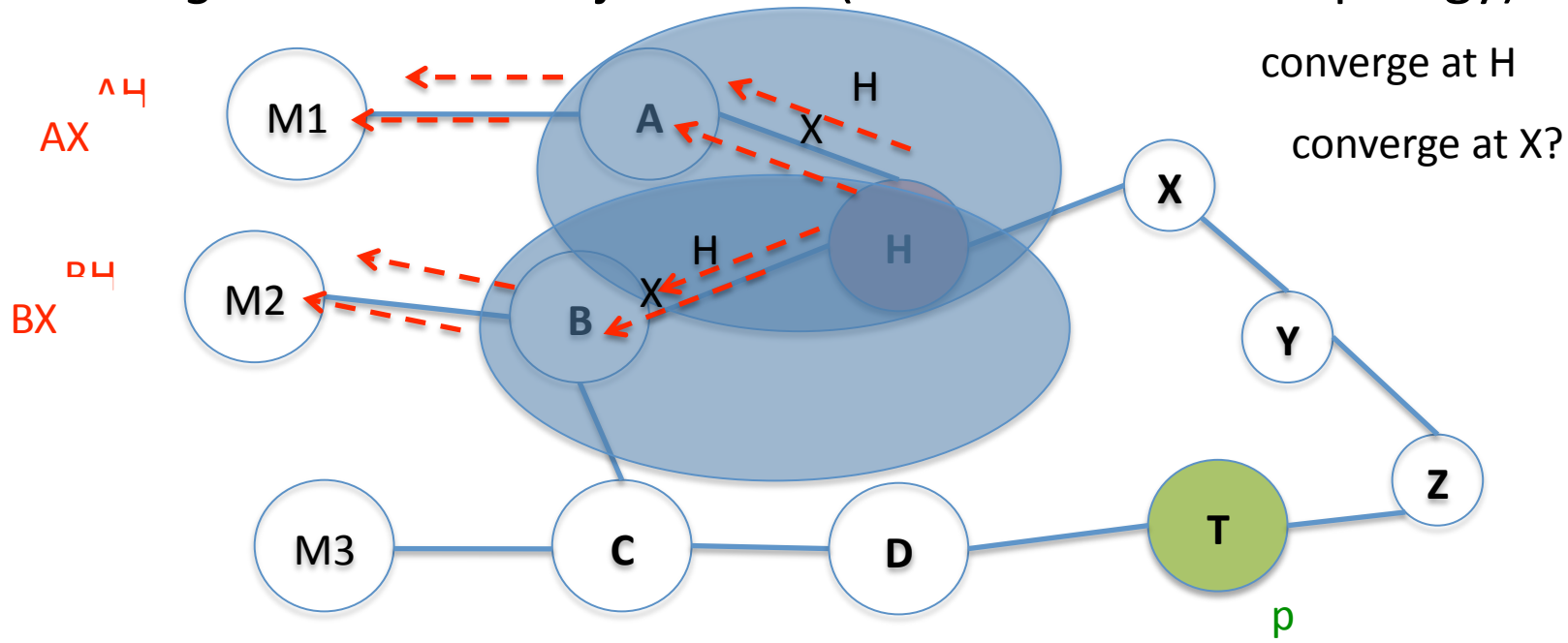
Two key observations

- Countermeasure ability
  - The hijacker cannot manipulate the portion of AS path from a polluted vantage point to the upstream neighbor AS of the hijacker AS



Two key observations

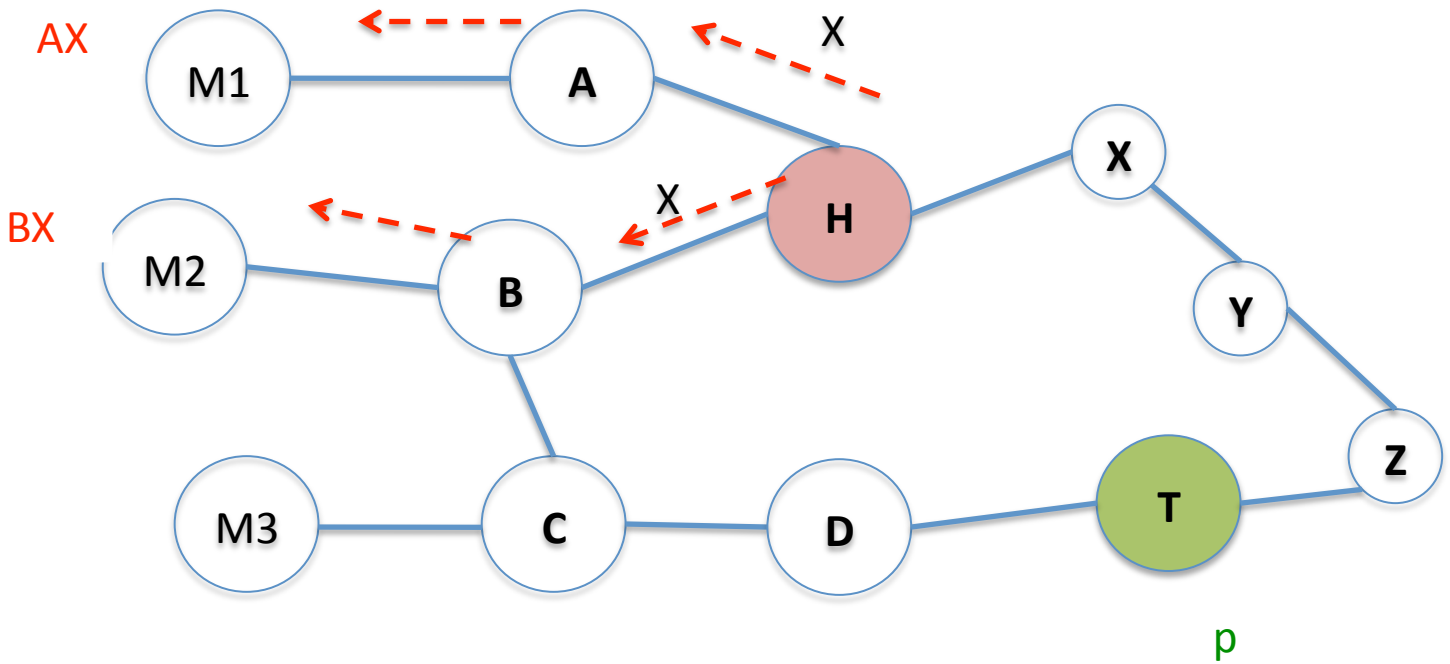
- Convergence: The trustworthy portion of polluted AS paths from multiple vantage points to a hijacked victim AS prefix *converge around* the hijacker AS (based on real AS topology).



Basic Locating Algorithm

- Indentifying hijacker search space
  - *Neighborset of one AS*: ASes one-hop away (include itself)
  - Based on existing AS topology
  - The union of *neighborset* of all ASes on all polluted paths (why?)
  - The hijacker should be in the space (based on observation 1)
- Ranking all ASes in the search space
  - Based on observation 2
  - The more frequently an AS appears, the higher its ranking is
  - Tie breaker: The closer an AS to the monitors, the higher its ranking is

Basic Locating Algorithm Example



Monitors	Polluted AS PATH	Neighbor Set	Hijacker List
M1	A X	(A H) (H X Y)	H > (4 times) X > Y > (2 times) A = B > C (once)
M2	B X	(B H C) (H X Y)	

Improvements

- Search space of basic algorithm
  - Trim the suspect list
- Improvement I: AS relationship
  - Basic algorithm neighborset
  - Valley free
  - Trim the neighorset on “trustworthy” ASes
- Improvement II: excluding “innocent” ASes
- Two improvements may introduce false negative

Evaluation

- Three sets of experiments:
  - Simulating synthetic prefix hijacking events
  - Reconstructed previous known hijacking events
  - Real prefix hijacking events

Simulating Synthetic Prefix Hijacking Events

- Hijacker *h* and source *s* from 73 Planetlab nodes
  - http://www.planet-lab.org/
- 451 Target prefix *t*
  - Multiple Origin ASes (MOAS) prefix
  - Single Origin Ases with large traffic
  - Popular website (based on Alexa ranking)
- Emulate all possible hijacking events
  - Based on the combination of (*s, h, t*)
  - Imposture, interception, and malicious (countermeasure) cases
- Monitor selection
  - From Planetlab nodes
  - Based on the target prefix

Effectiveness and Improvement

Algorithms	All monitors					
	Imposture		Interception		Malicious	
	Accuracy	FNR	Accuracy	FNR	Accuracy	FNR
B	88.7%	0.00%	86.3%	0.00%	85.4%	0.00%
B+I1	89.8%	0.03%	90.3%	0.17%	88.6%	0.14%
B+I2	91.3%	0.09%	93.1%	0.16%	90.4%	0.10%
B+I1+I2	94.2%	0.09%	94.3%	0.24%	93.1%	0.18%

- The accuracy of basic algorithm is 85%+
- Combine both improvements, the accuracy is up to 94.3%
- False negative ratio is relatively low.

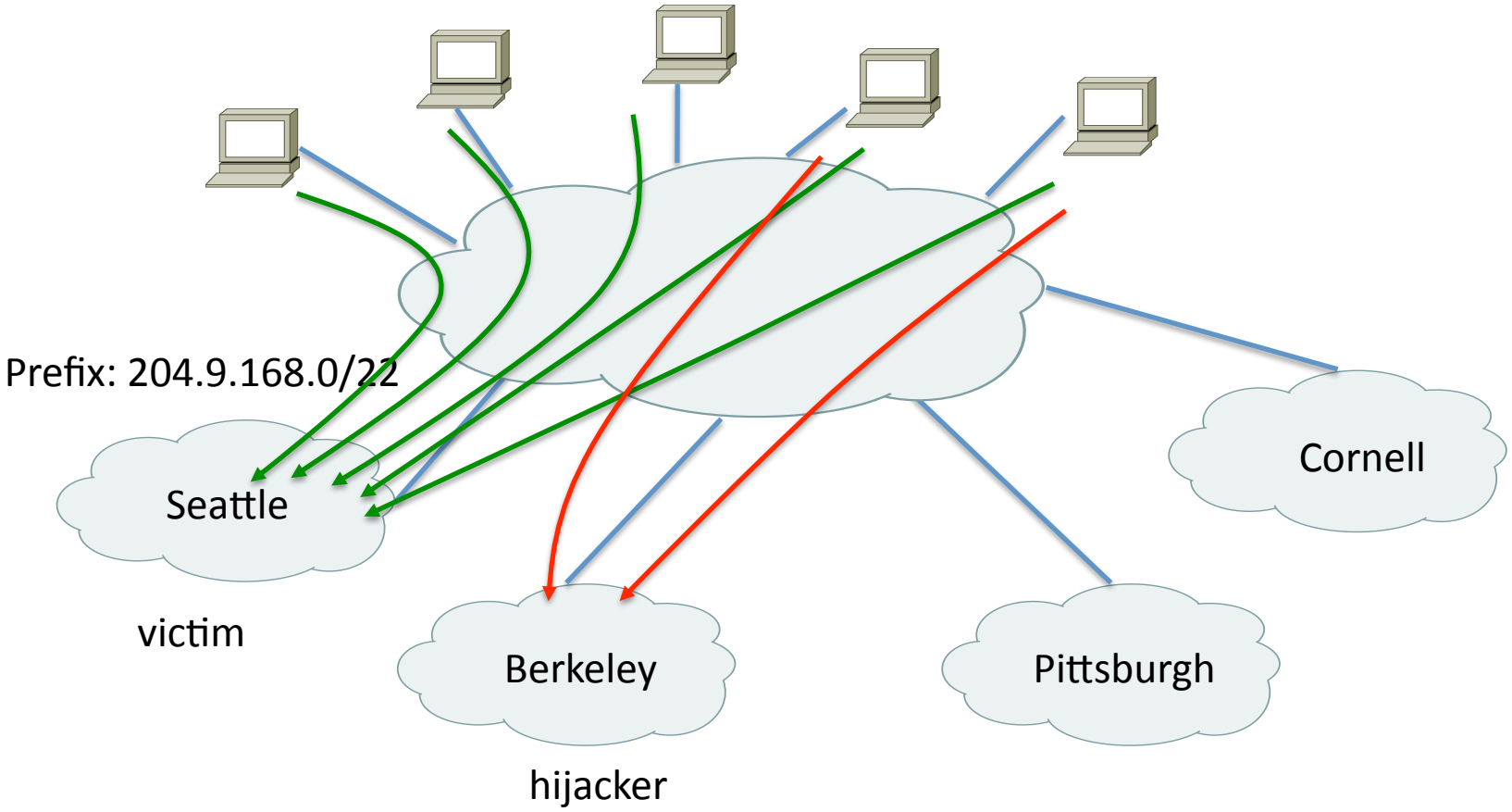
Reconstruct Previously-known Hijacking Events

7 hijacking events

Locate all hijackers

Victim AS	Hijacker AS	Date	#monitors
3691	6461	March 15, 2008	16
36561 (YouTube)	17557	February 24, 2008	9
11643 (eBay)	10139	November 30, 2007	7
4678	17606	January 15, 2007	8
7018	31604	January 13, 2007	13
1299	9930	September 7, 2006	5
701, 1239	23520	June 7, 2006	12

Real Hijacking Events



Real Hijacking Events (cont.)

Victim Site	Hijacker Site	Launch Time (EST)	Response Time (minutes)	Required monitors
Cornell	Berkeley	May 2 12:01:31	13	12
	Seattle	May 2 16:12:47	7	10
	Pittsburgh	May 2 17:34:39	9	9
Pittsburgh	Cornell	May 2 19:32:09	13	14
	Berkeley	May 2 22:50:25	11	15
	Seattle	May 3 02:26:26	12	15
Seattle	Cornell	May 3 11:20:42	9	8
	Pittsburgh	May 3 13:03:10	12	12
	Berkeley	May 3 19:16:16	8	18
Berkeley	Seattle	May 3 22:35:07	13	14
	Pittsburgh	May 4 00:01:01	12	16
	Cornell	May 4 11:19:20	11	10

Conclusion

- LOCK to locate prefix hijacker ASes
  - First study of hijacker location problem
  - Locate the hijacker even when countermeasures are engaged
  - Extensively evaluation illustrates high location accuracy

Acknowledgement

- Authors Tongqing Qiu and Jun (Jim) Xu would like to acknowledge the generous support from the NSF CyberTrust program (specifically CNS 0716423)

- Thanks You!
- Questions