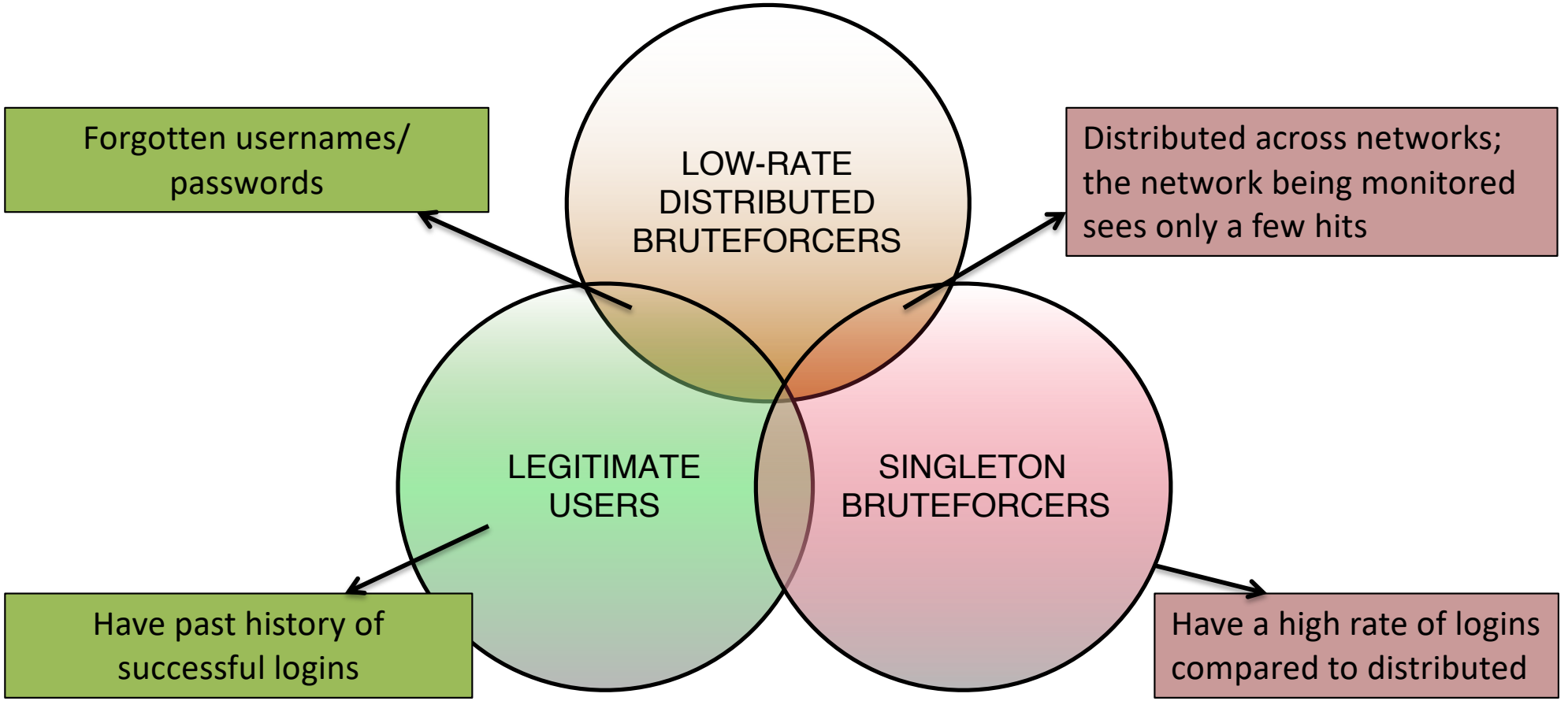
User Populations



Characteristics overlap between legitimate users and bruteforcers

Time span	Jan 2005–Dec 2012
SSH servers	2,243
Valid users	4,364
Distinct valid user/server pairs	10,809
Login attempts	12,917,223
Login successes	8,935,298
Remote clients	154,318
Attempts using passwords	5,354,833
successes	1,416,590
remote clients	119,826
SSH border flows	215,244,481
remote clients seen in flows	140,164
High-rate brute-forcers	7,476
Mean attempts per high-rate brute-forcer	382.84
Mean daily password login attempts	486.13 ($\sigma = 182.95$)
Mean daily users	116.44 ($\sigma = 32.41$)

Table 1: Summary of LBNL syslog and flow data.

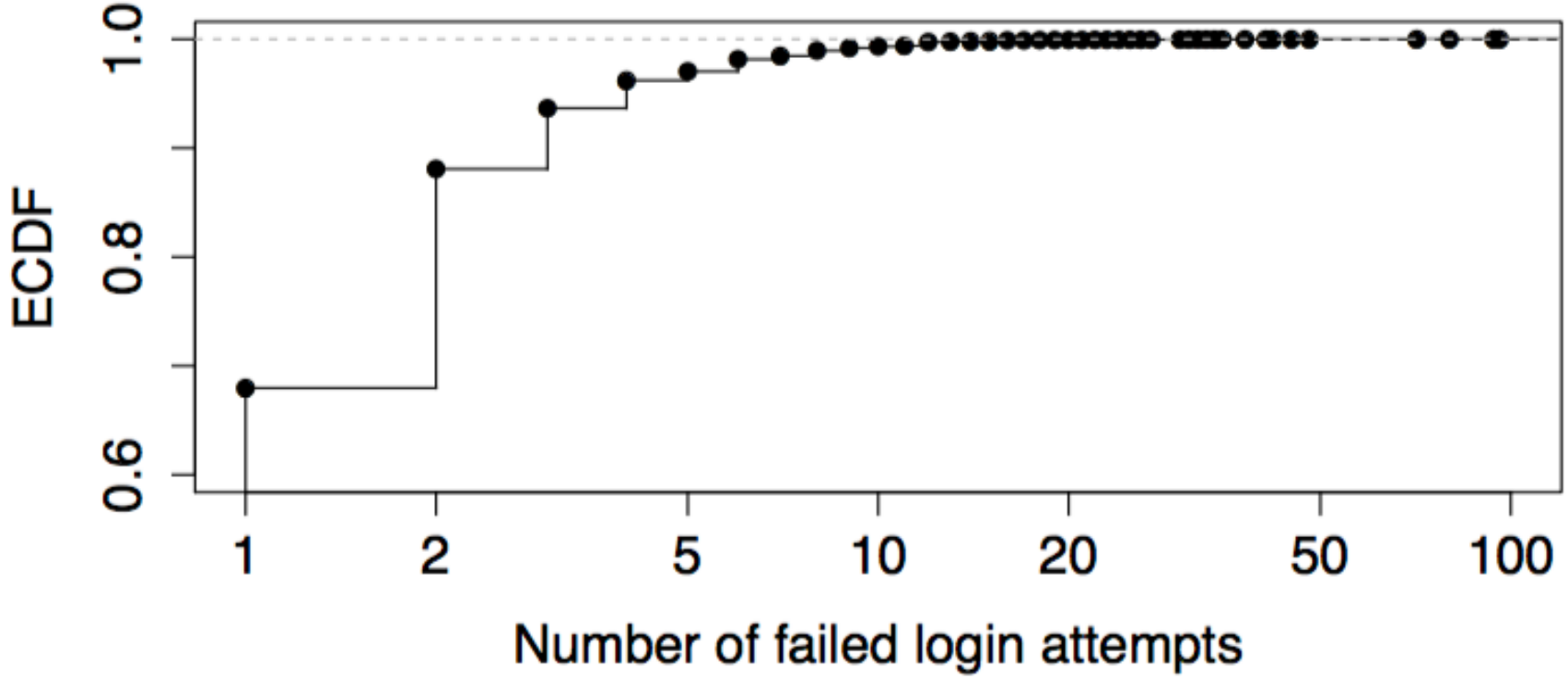
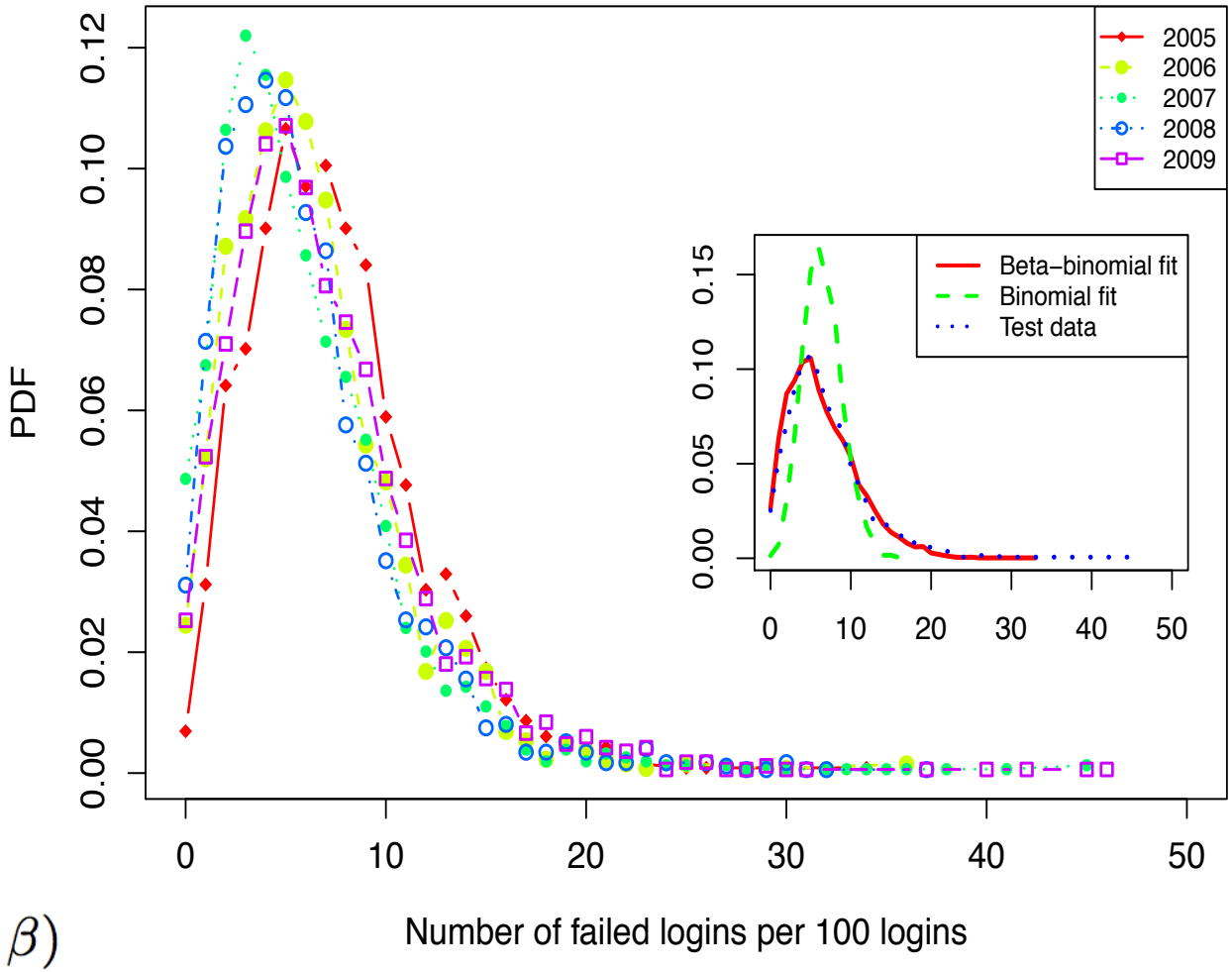


Figure 1: Empirical CDF of the number of failed login attempts per hour until a success for legitimate user login efforts with forgotten or mistyped usernames/passwords.

Aggregate Site Analyzer

- Site-wide parameter
  - *Global Failure Indicator* (GFI)
    - Site-wide number of failed logins per batch of n logins
- GFI well-modeled as Beta–binomial
  - Binomial with beta-prior on probability of success



$$k \sim \binom{n}{k} \frac{\mathrm{Beta}(k+\alpha, n-k+\beta)}{\mathrm{Beta}(\alpha, \beta)}$$

Aggregate Site Analyzer

Monitoring for Change (CUSUM Algorithm)

$$C_0 = 0$$

$$C_n = \max(0, C_{n-1} + X_n - \mu - k)$$

X_n – Random variable (GFI)

μ - Mean under normal behavior

k - Parameter based on magnitude of change to be detected

If we just used this term, we'd have an ordinary random walk around the origin

Aggregate Site Analyzer

Monitoring for Change (CUSUM Algorithm)

$$C_0 = 0$$

$$C_n = \max(0, C_{n-1} + X_n - \mu - k)$$

X_n – Random variable (GFI)

μ - Mean under normal behavior

k - Parameter based on magnitude of change to be detected

Including this term *biases* the random walk negatively …

Aggregate Site Analyzer

Monitoring for Change (CUSUM Algorithm)

$$C_0 = 0$$

$$C_n = \max(0, C_{n-1} + X_n - \mu - k)$$

X_n – Random variable (GFI)

μ - Mean under normal behavior

k - Parameter based on magnitude of change to be detected

… and this term makes sure it never goes below the X axis

Aggregate Site Analyzer

Monitoring for Change (CUSUM Algorithm)

$$C_0 = 0$$

$$C_n = \max(0, C_{n-1} + X_n - \mu - k)$$

X_n – Random variable (GFI)

μ - Mean under normal behavior

k - Parameter based on magnitude of change to be detected

Putting it together, we have a **one-sided random walk** that will make (short) excursions upwards from zero, but always returns …

Aggregate Site Analyzer

Monitoring for Change (CUSUM Algorithm)

$$C_0 = 0$$

$$C_n = \max(0, C_{n-1} + X_n - \mu - k)$$

X_n – Random variable (GFI)

μ - Mean under normal behavior

k - Parameter based on magnitude of change to be detected

… *unless* the process *changes* and **failures become more likely**, such that μ' > μ + k, in which case the random walk *steadily climbs upward*!

Aggregate Site Analyzer

Monitoring for Change (CUSUM Algorithm)

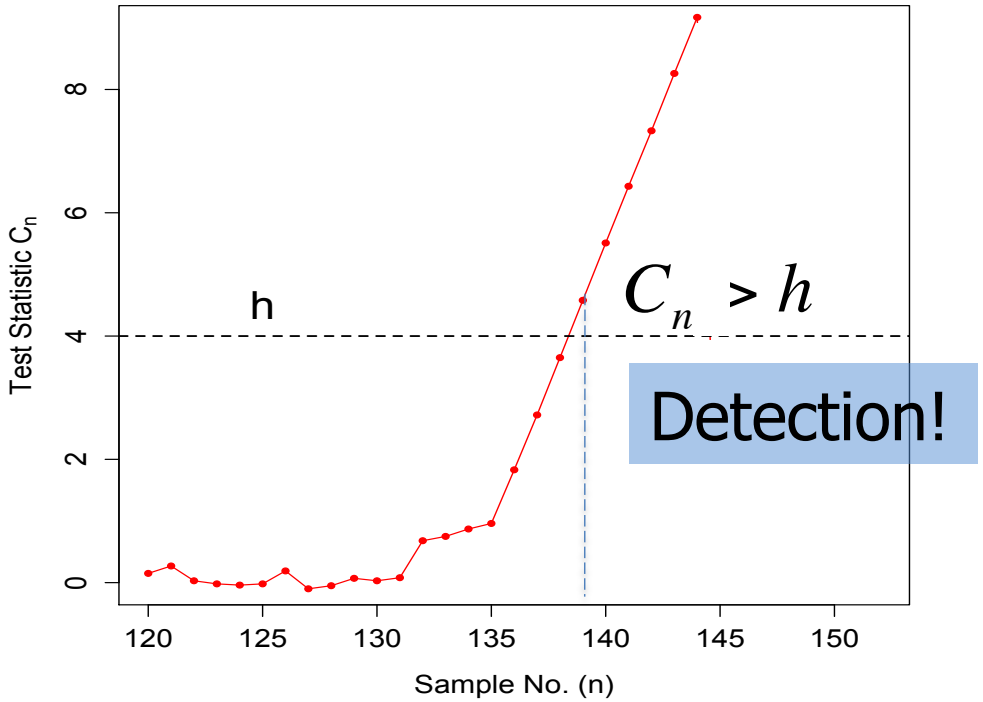
$$C_0 = 0$$

$$C_n = \max(0, C_{n-1} + X_n - \mu - k)$$

X_n – Random variable (GFI)

μ - Mean under normal behavior

k - Parameter based on magnitude of change to be detected



Aggregate Site Analyzer

Monitoring for Change (CUSUM Algorithm)

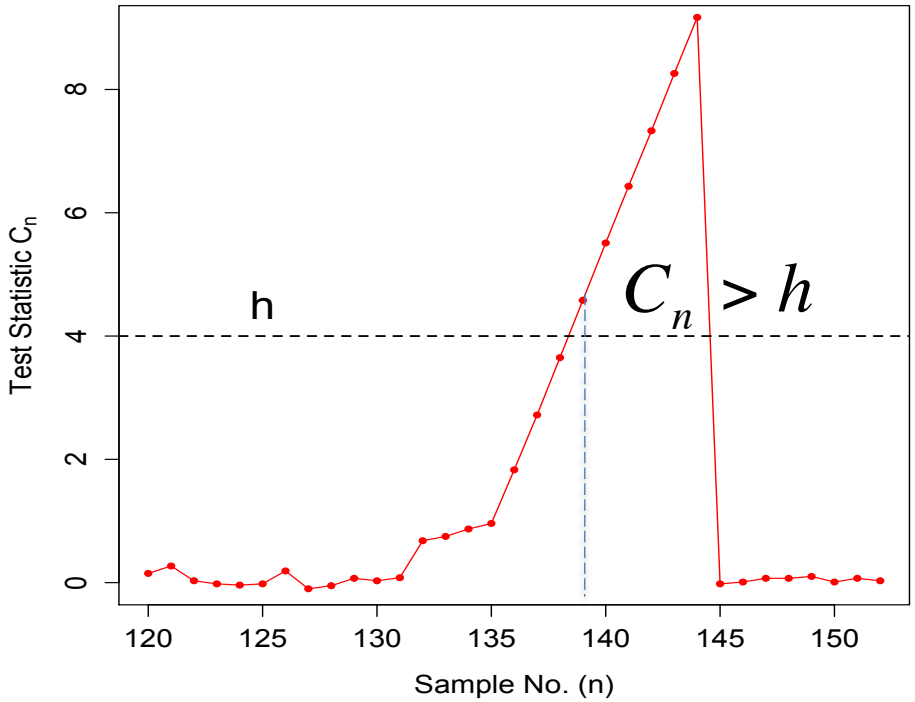
$$C_0 = 0$$

$$C_n = \max(0, C_{n-1} + X_n - \mu - k)$$

X_n – Random variable (GFI)

μ - Mean under normal behavior

k - Parameter based on magnitude of change to be detected



Manually reset to zero upon observing $X_n < \mu$

Aggregate Site Analyzer

Monitoring for Change (CUSUM Algorithm)

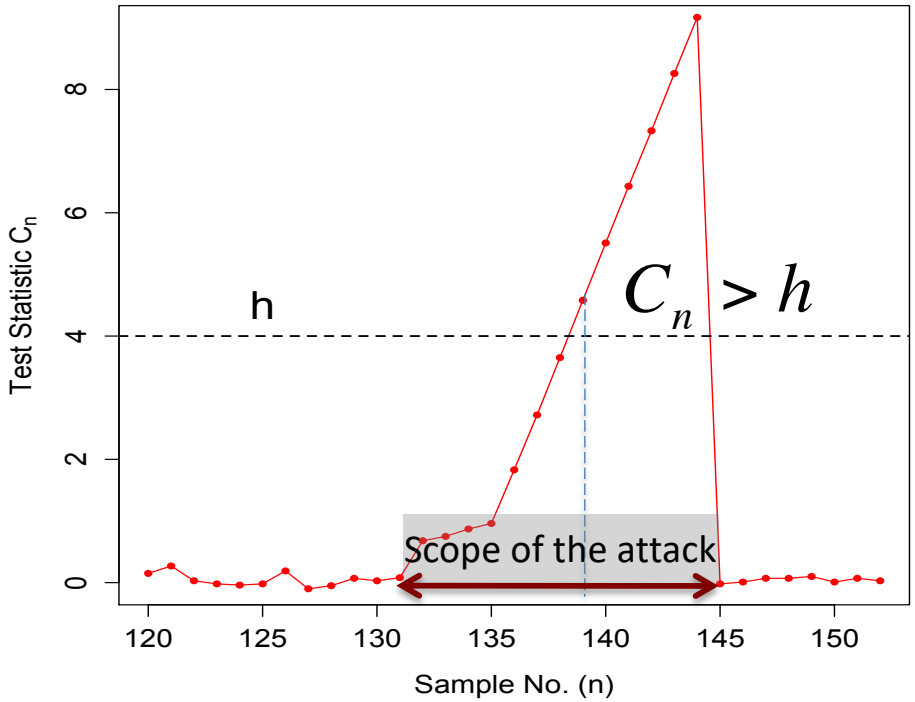
$$C_0 = 0$$

$$C_n = \max(0, C_{n-1} + X_n - \mu - k)$$

X_n – Random variable (GFI)

μ - Mean under normal behavior

k - Parameter based on magnitude of change to be detected



Evaluation

Aggregate Site Analyzer	
Total number of attacks	**99**
Number of false attacks	9

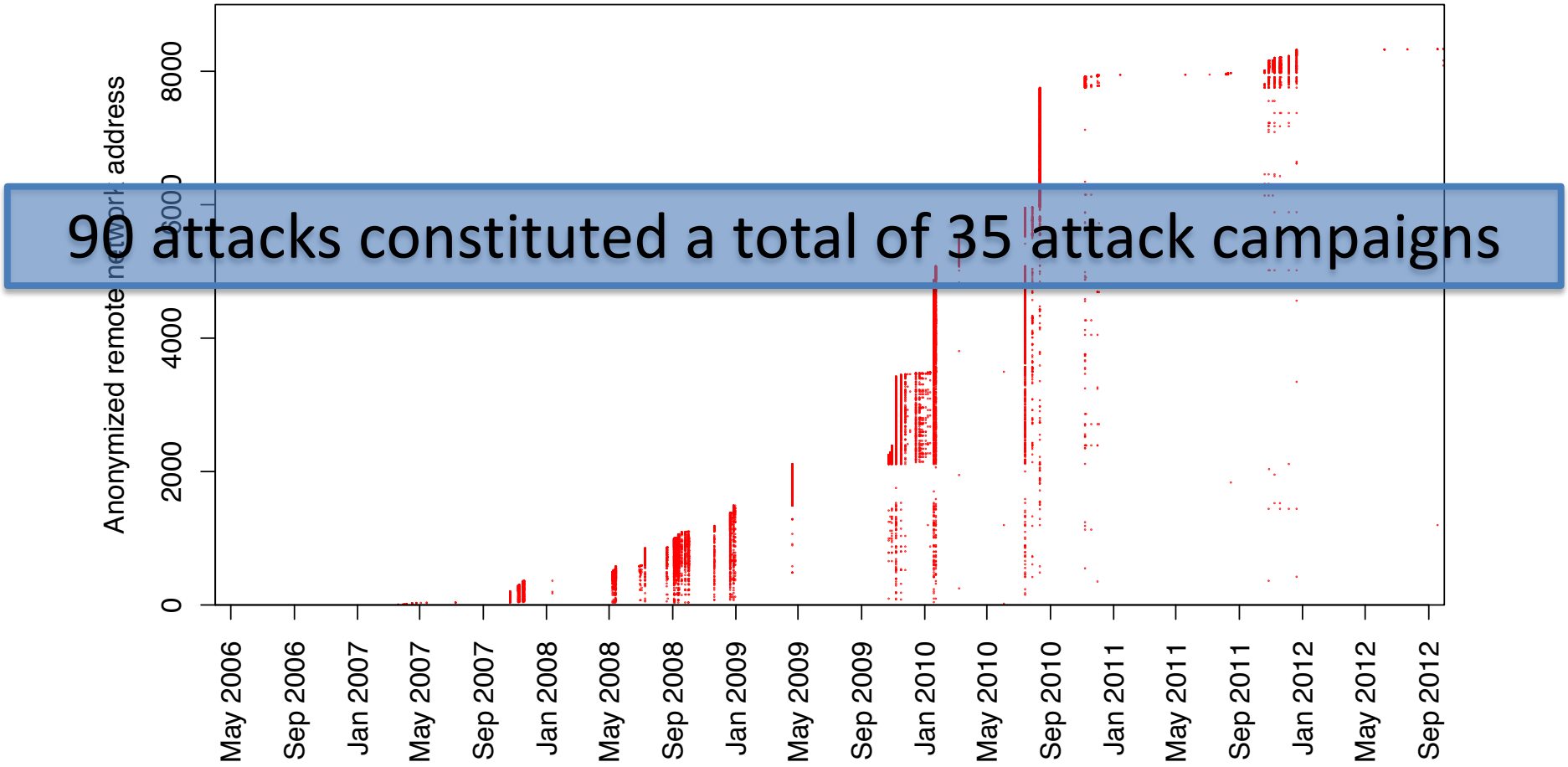
Determined by Attack Participants Classifier

Attack Participants Classifier	
Number of attack hosts	**9,306**
Number of false attack hosts	37

Determined by future successful activity/
Site Incident Database

Characterization of Attacks

Overlap of attack sources over different attacks



90 attacks constituted a total of 35 attack campaigns

ID	Appearances	Attrs.	Aggregate statistics		Per remote avg. hourly characteristics		
			Attack machines	Local machines	Attempts	Locals contacted	Per-Local attempts
1	2007: [Jul 7-9], [Oct 20-23], [Nov 5-9](2), [Nov 13-18](2)	**L,!!**	431	133	74.68	56.10	1.33
2	2008: [Apr 30 - May 7],[May 8-14](3)	**L**	286	140	98.50	54.80	1.79
3	2008: [Jun 28-29], [Jun 30 - Jul 1] [Jul 7-9], [Aug 17-21], [Sep 1-8] (5)	**L**	969	113	293.30	41.70	7.00
4	2008: [Sep 8-13](3)	**L**	378	257	52.50	40.70	1.28
5	2008: [Sep 16-18]	**L,S,T**	88	12	9.00	2.53	3.57
6	2008: [Sep 23-26](2), [Sep 29 - Oct 2](2)	**L**	185	109	48.50	38.38	1.26
7	2008: [Nov 18-19], [Nov 20 - Dec 29](5) 2009: [Apr 7-9]	**L,S**	1,097	22	16.01	8.04	1.99
8	2009: [Oct 22-23], [Oct 27 - Nov 24](5)	**L,S**	1,734	5	5.60	3.70	1.50
9	2010: [Dec 6 - Jan 10](6), [Jan 11-18], [Jan 20-22], [Mar 4-8]	**L**	3,496	44	38.80	21.50	1.80
10	2010: [Jun 16 - Jul 27](2), [Jul 29 - Aug 11]	**L**	7,445	1,494	90.80	34.50	2.70
11	2010: [Nov 1-6] (2), [Nov 7-8], [Nov 27 - Dec 1], [Dec 15-17]	**L,!**	581	98	140.60	45.47	3.09
12	2011: [Oct 11-19], [Oct 25-29](2), [Nov 4-7], [Nov 17-20]	**L**	377	158	33.93	25.25	1.34
13	2010: [Mar 30 - Apr 1]	**R,t**	78	18,815	999.70	118.91	1.33
14	2010: [Apr 23-26]	**R,t**	130	29,924	2325.57	117.97	1.22
15	2010: [May 7-10]	**R,t**	72	9,300	713.05	67.47	1.36
16	2010: [Sep 20-22]	**R,t**	33	5,380	69.05	60.72	1.14
17	2010: [Dec 27-30]	**R,t**	32	3,881	260.59	43.11	1.34
18	2011: [Feb 10-14](2)	**R,t**	108	7,520	40.45	27.21	1.48
19	2011: [May 16-18]	**R,t**	30	1,621	153.23	19.70	2.02
20	2011: [Jul 21-22]	**R,t**	20	2,556	388.25	38.13	1.18
21	2011: [Aug 2-6]	**R,t**	45	9,465	315.12	21.66	2.41
22	2011: [Aug 7-9]	**R,t**	48	6,516	444.16	17.60	2.18
23	2011: [Aug 17-21](2)	**R,t**	22	3,279	33.07	16.40	2.02
24	2011: [Nov 2-4]	**R**	31	3,446	273.80	20.08	1.02
25	2011: [Nov 30 - Dec 5]	**R**	181	10,467	829.68	18.31	1.03
26	2011: [Dec 18-20]	**R**	258	961	1099.85	14.00	1.02
27	2012: [Jul 20-21]	**R,t**	2	53,219	20,844	11,749	1.06
28	2012: [Aug 27 - Sep 2]	**R,t**	10	1,912	20.84	14.38	1.23
29	2012: [Sep 26-29]	**R**	6	1,971	72.30	13.05	1.59
30	2012: [Oct 8 - Nov 1](4)	**R,S**	190	19,639	5.27	4.97	1.06
31	2012: [Nov 16-18]	**R,t**	3	493	38.36	12.22	2.99
32	2012: [Nov 30 - Dec 2]	**R,t**	3	344	133.00	68.80	1.93
33	2008: [Jan 9-12]	**X,t**	17	63,015	2,846.44	1,761.69	1.61
34	2011: [Apr 8-26]	**X,t**	67	19,158	591.34	87.41	6.76
35	2012: [Dec 14-17]	**X,t**	13	45,738	1,490.26	1,430.67	1.04

Attack Campaign Stealthiness

DETECTION COMPARISON

- Point-wise Host detector (0/35)

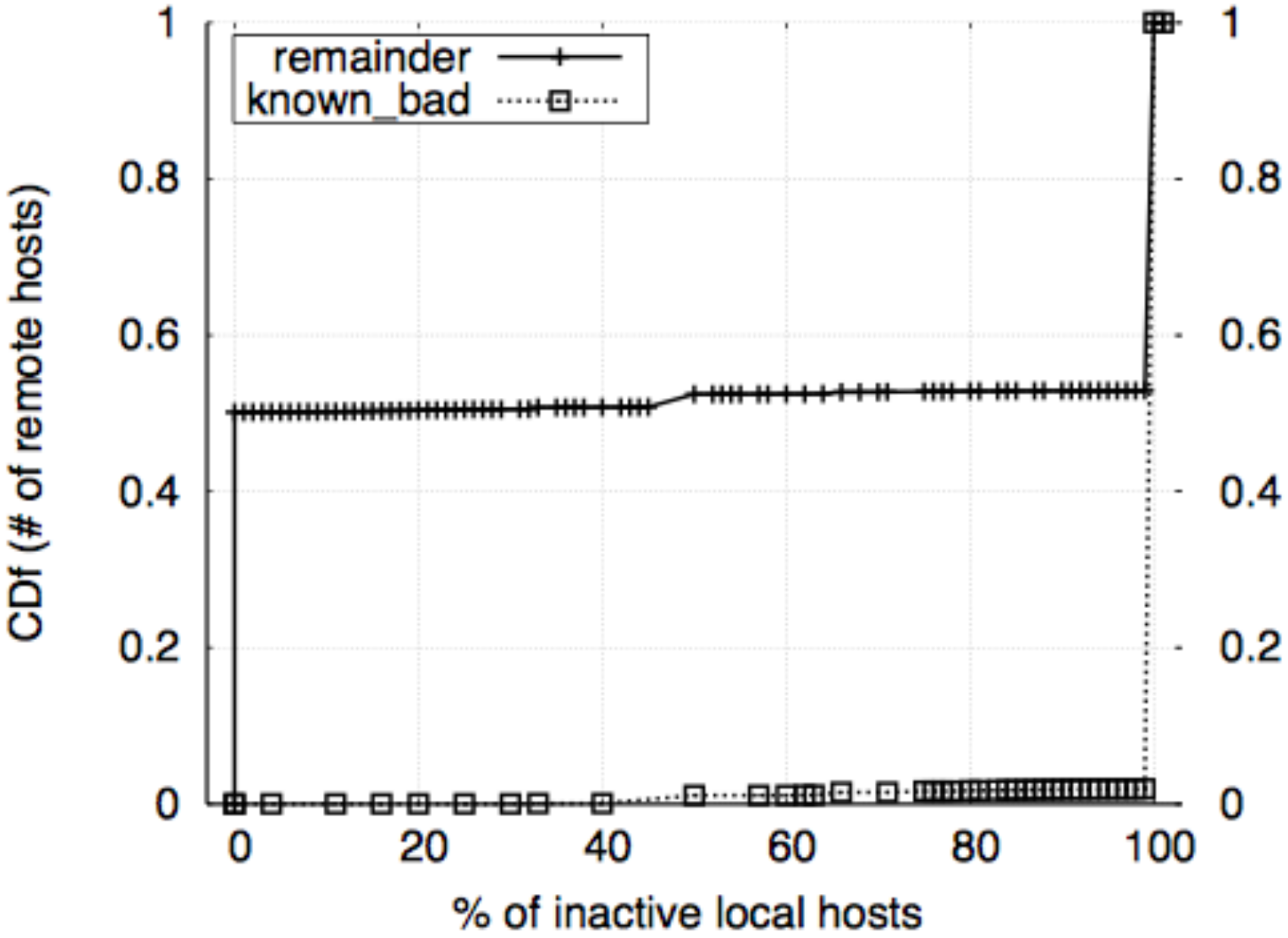
On average 2 attempts per local machine per hour

Two of the campaigns succeeded in breaking-in; one undetected by the site
One stealthy attack specifically targeted LBNL (valid usernames)

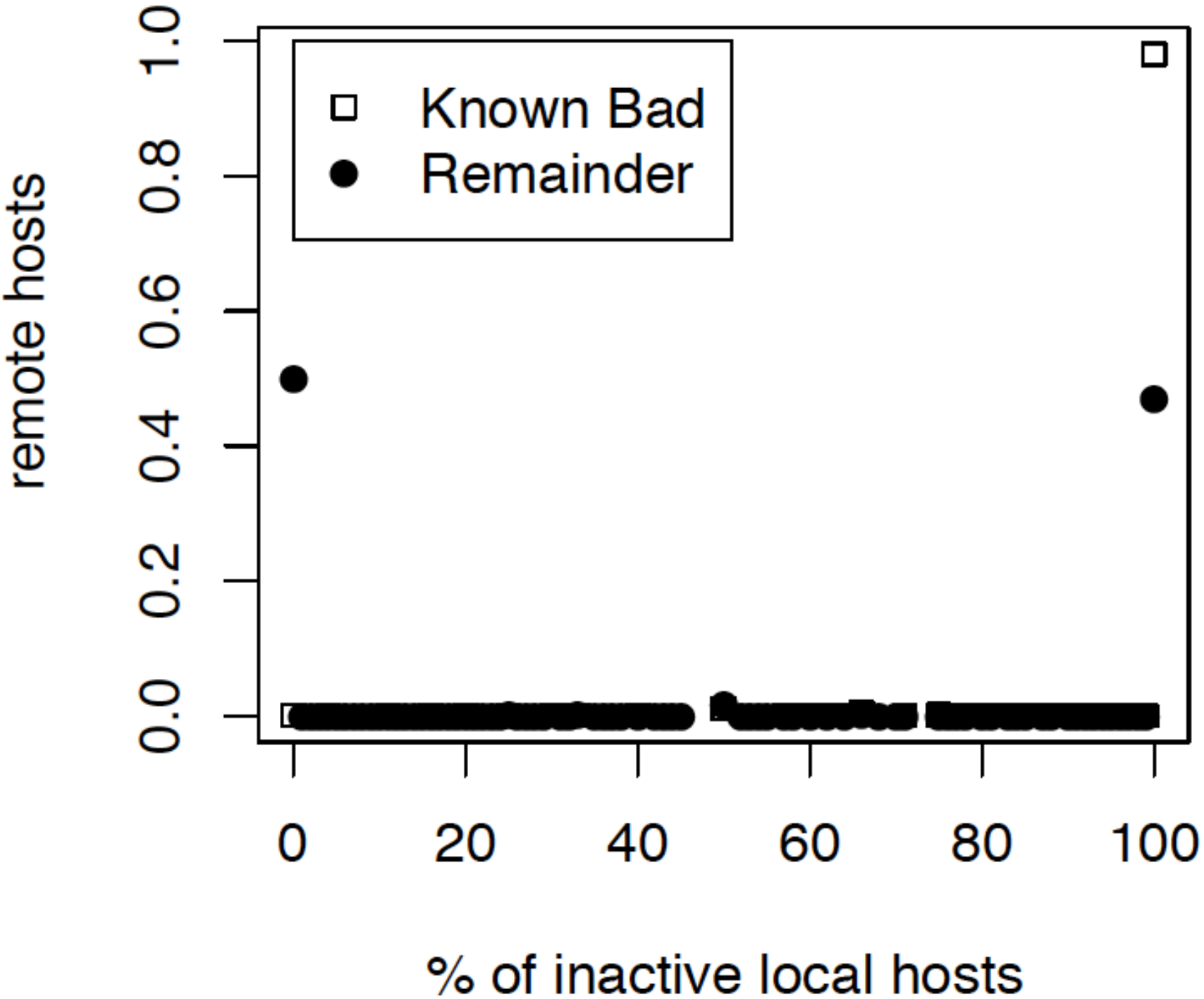
(31/35 – Partially detectable)

High-rate hourly activity in total number of failed attempts /
# of local hosts contacted

- Undetectable by any point-wise detector (4/35)


remainder
known_bad
CDf (# of remote hosts)
% of inactive local hosts
0
0.2
0.4
0.6
0.8
1
0
20
40
60
80
100

(a) LBL


Known Bad
Remainder
remote hosts
0.0
0.2
0.4
0.6
0.8
1.0
0
20
40
60
80
100
% of inactive local hosts

Timing Analysis of Keystrokes and Timing Attacks on SSH*

Dawn Xiaodong Song David Wagner Xuqing Tian
University of California, Berkeley

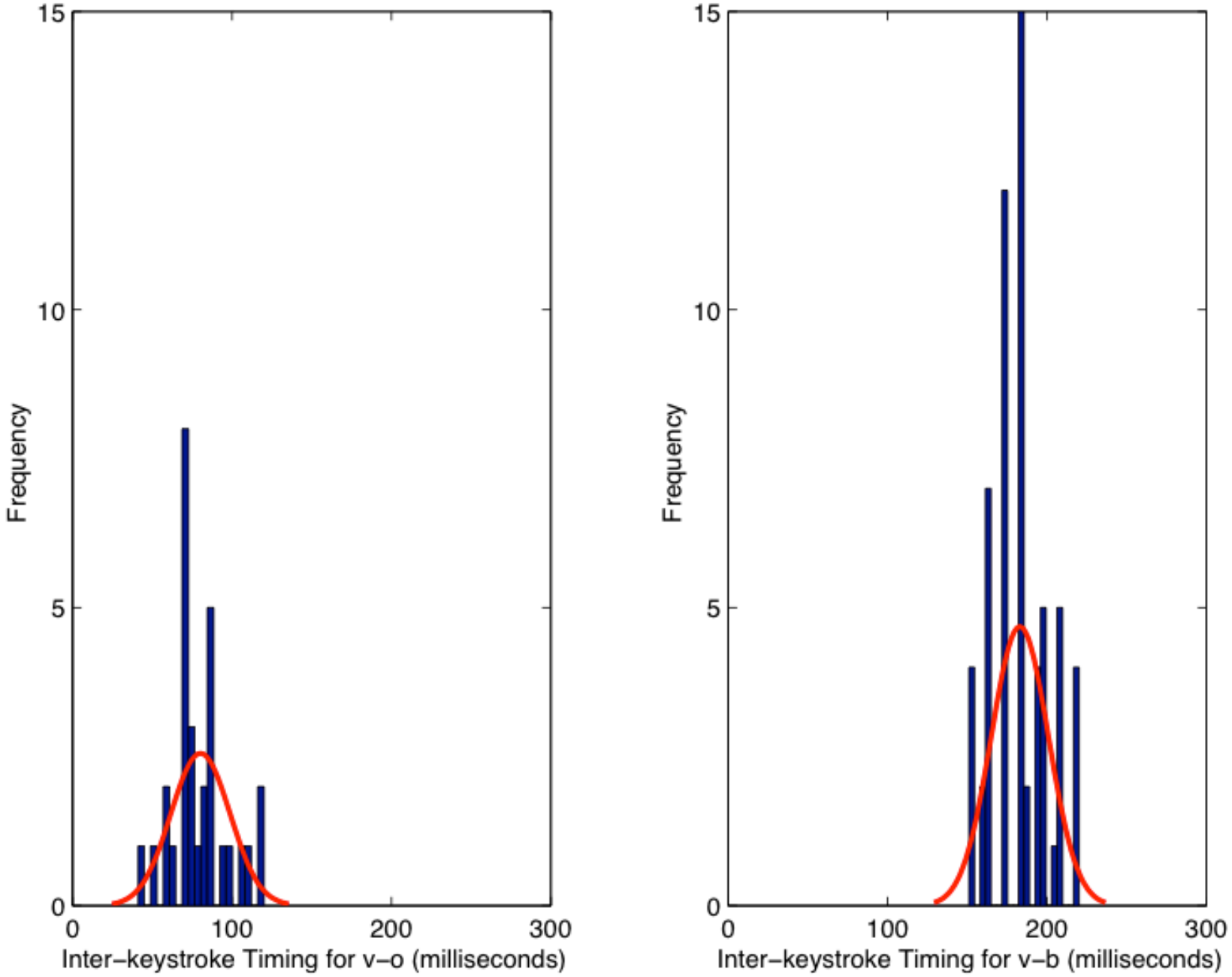


Figure 3: The distribution of inter-keystroke timings for two sample character pairs.

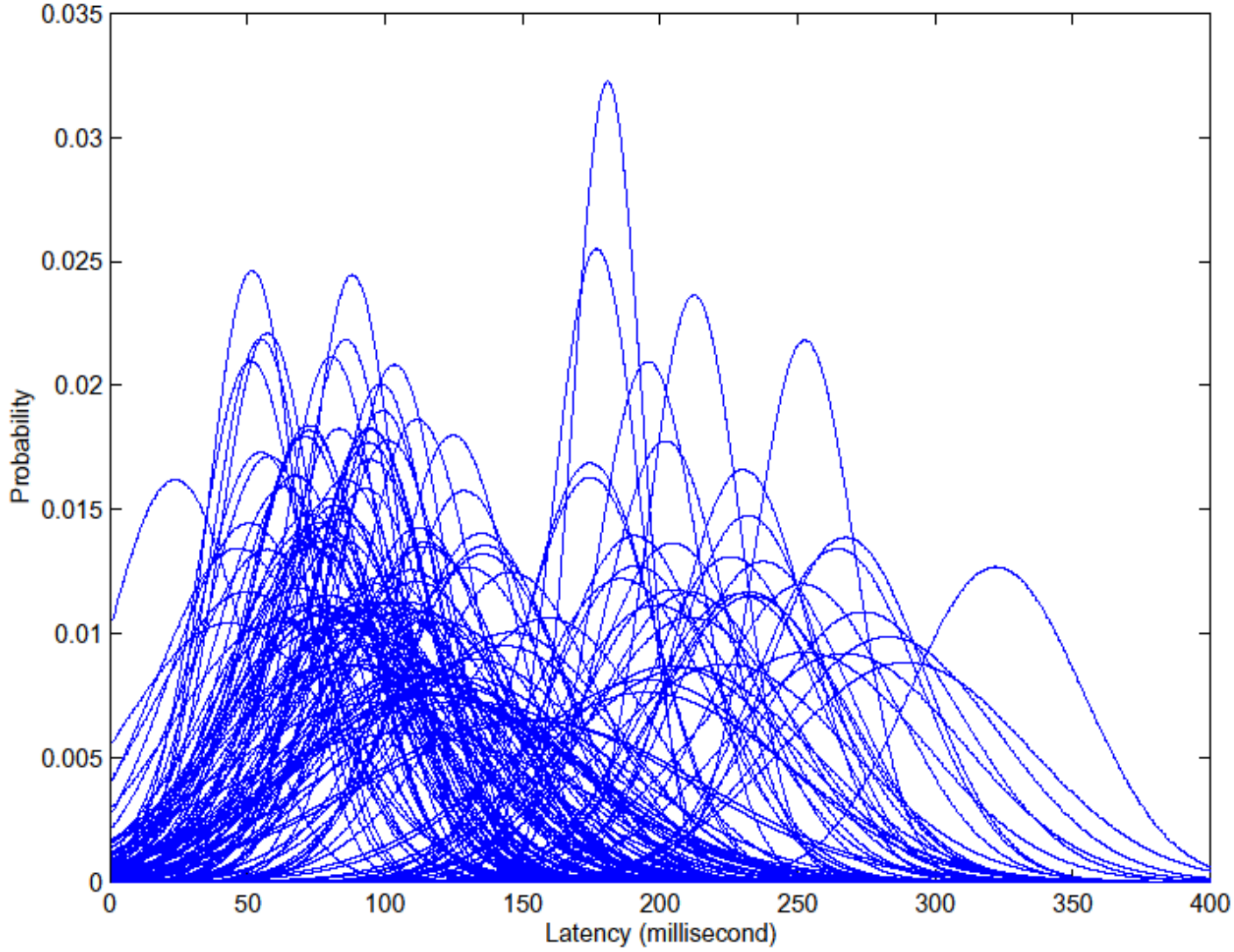


Figure 5: Estimated Gaussian distributions of all 142 character pairs collected from a user.

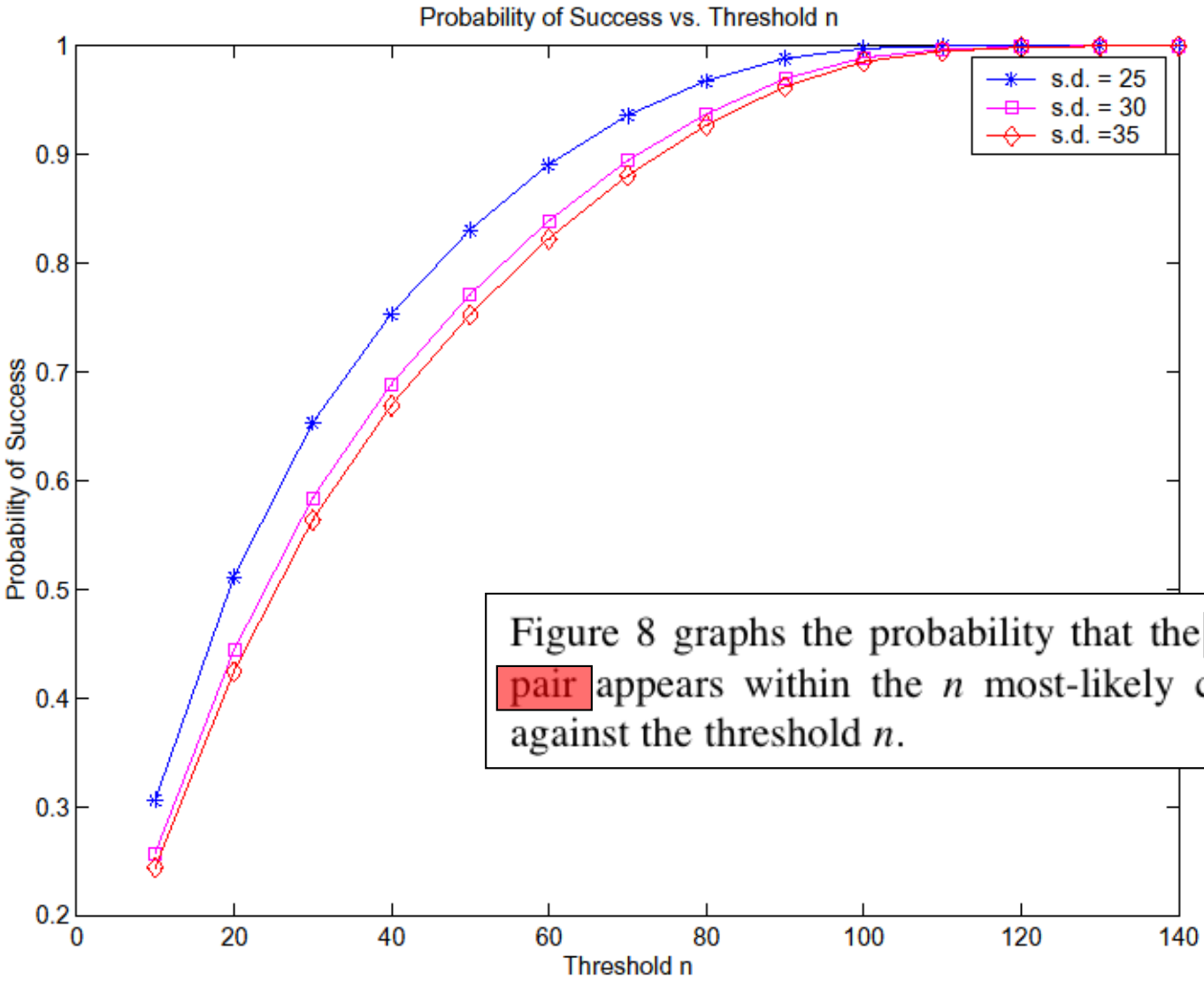


Figure 8 graphs the probability that the real character pair appears within the n most-likely character pairs against the threshold n.

Figure 8: The probability that the n-Viterbi algorithm outputs the correct password before the first n guesses, graphed as a function of n.

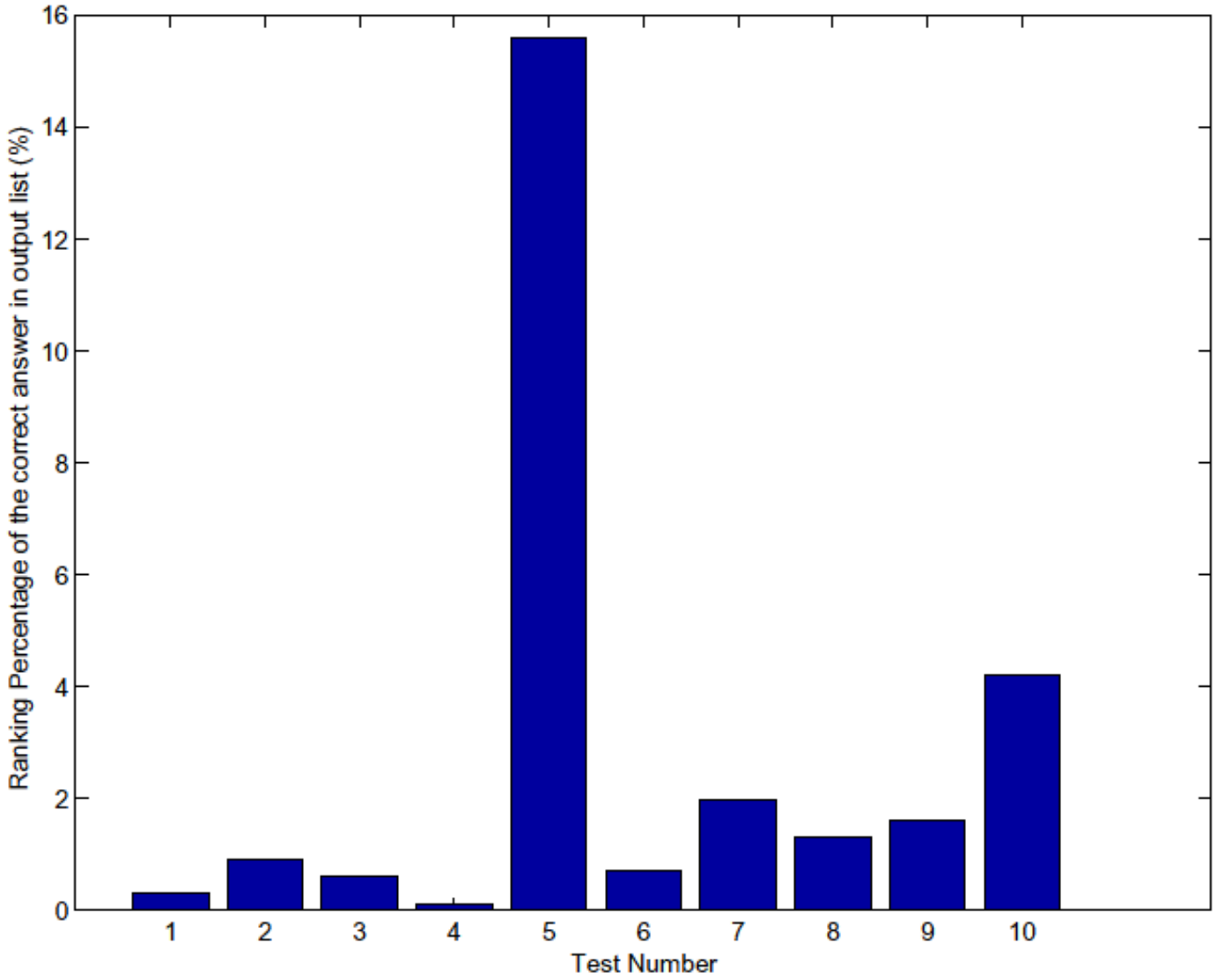
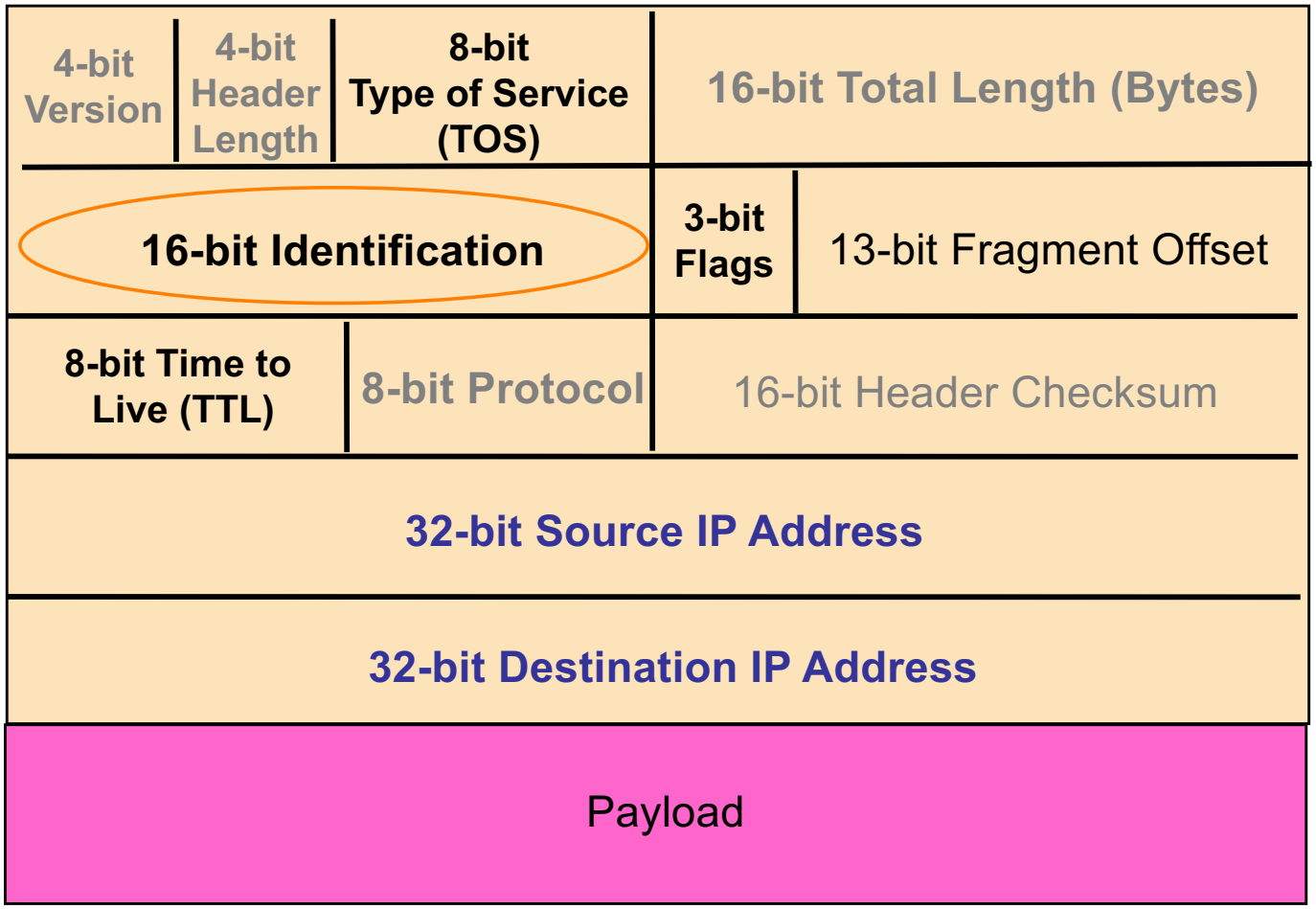


Figure 10: The percentage of the password space tried by Herbivore in 10 tests before finding the right password.

Training Set	Test Set	Test Cases				
		Password 1	Password 2	Password 3	Password 4	Password 5
User 1	User 1	15.6%	0.7%	2.0%	1.3%	1.6%
User 1	User 2	62.3%	15.2%	7.0%	14.8%	0.3%
User 1	User 3	6.4%	N/A	1.8%	3.1%	4.2%
User 1	User 4	1.9%	31.4%	1.1%	0.1%	28.8%
User 2	User 1	4.9%	1.3%	1.6%	12.3%	3.1%
User 2	User 2	30.8%	15.0%	2.8%	3.7%	2.9%
User 2	User 3	4.7%	N/A	5.3%	6.7%	38.4%
User 2	User 4	0.7%	16.8%	3.9%	0.6%	5.4%

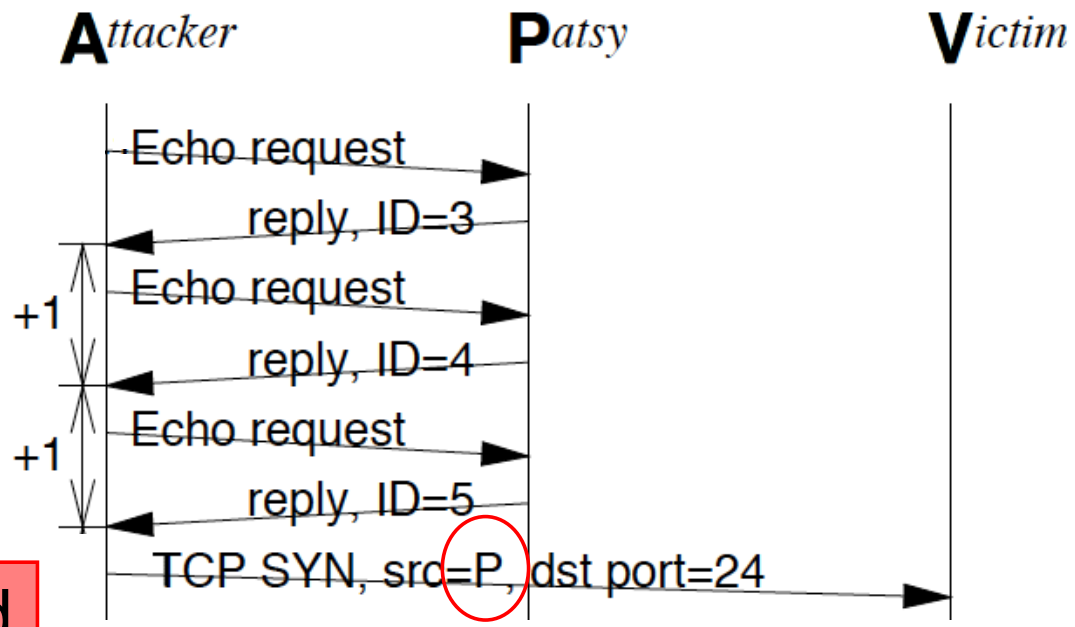
Table 1: Success rates for password inference with multiple users. The numbers are the percentage of the search space the attacker has to search before he finds the right password.

IP Header Side Channel

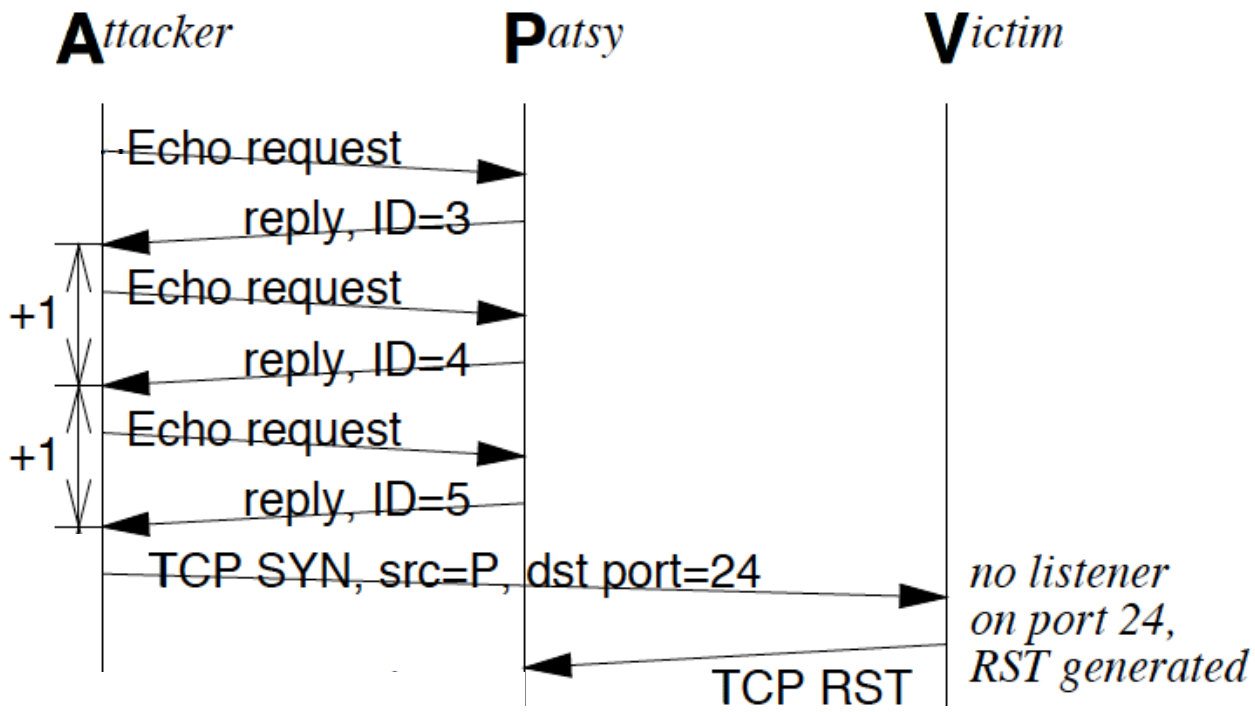


ID field is supposed to be unique per IP packet.

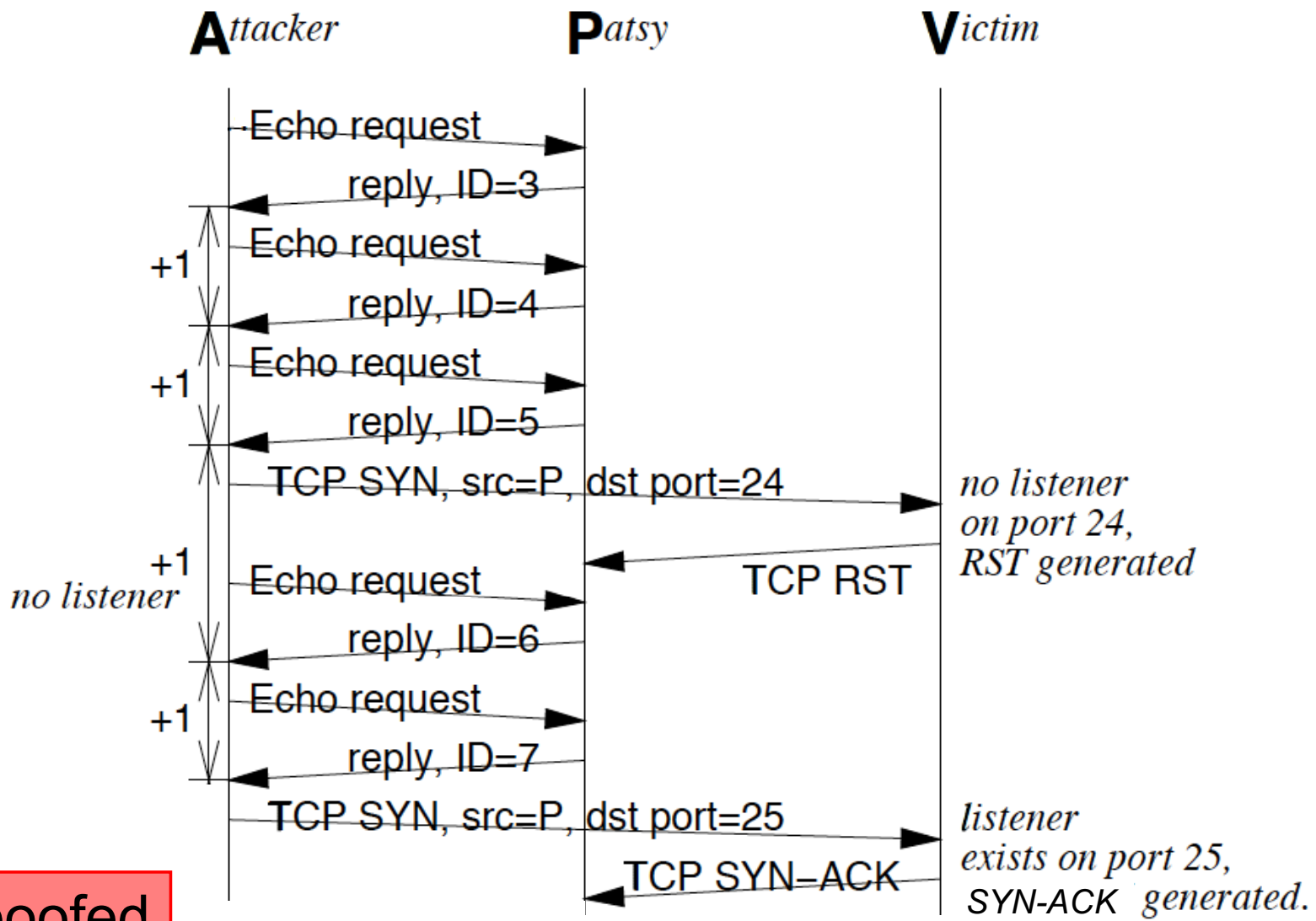
One easy way to do this: **increment** it each time system sends a new packet.


Attacker
Patsy
Victim
Echo request
reply, ID=3
Echo request
+1
reply, ID=4
Echo request
+1
reply, ID=5
TCP SYN, src=P, dst port=24

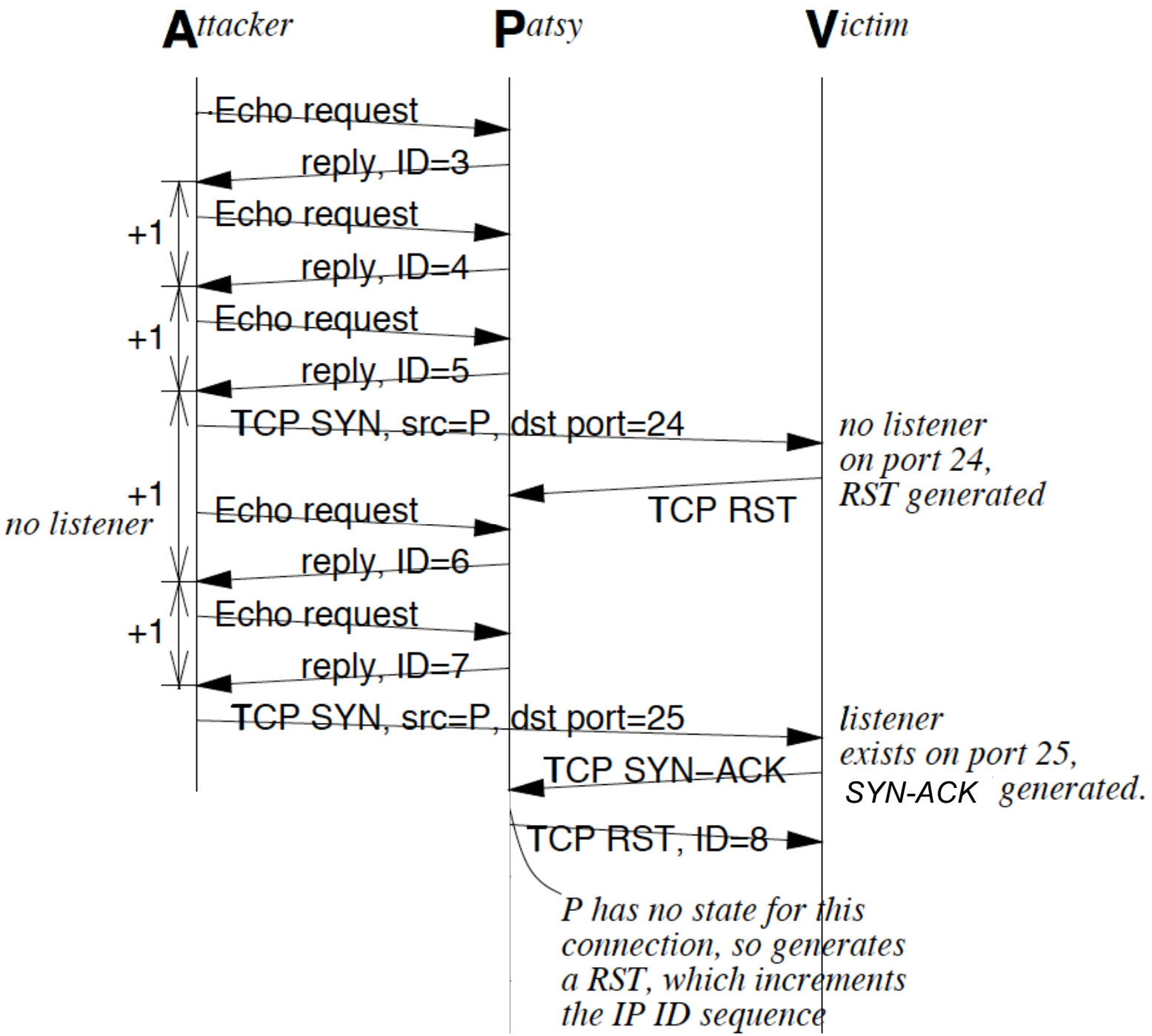
Spoofed

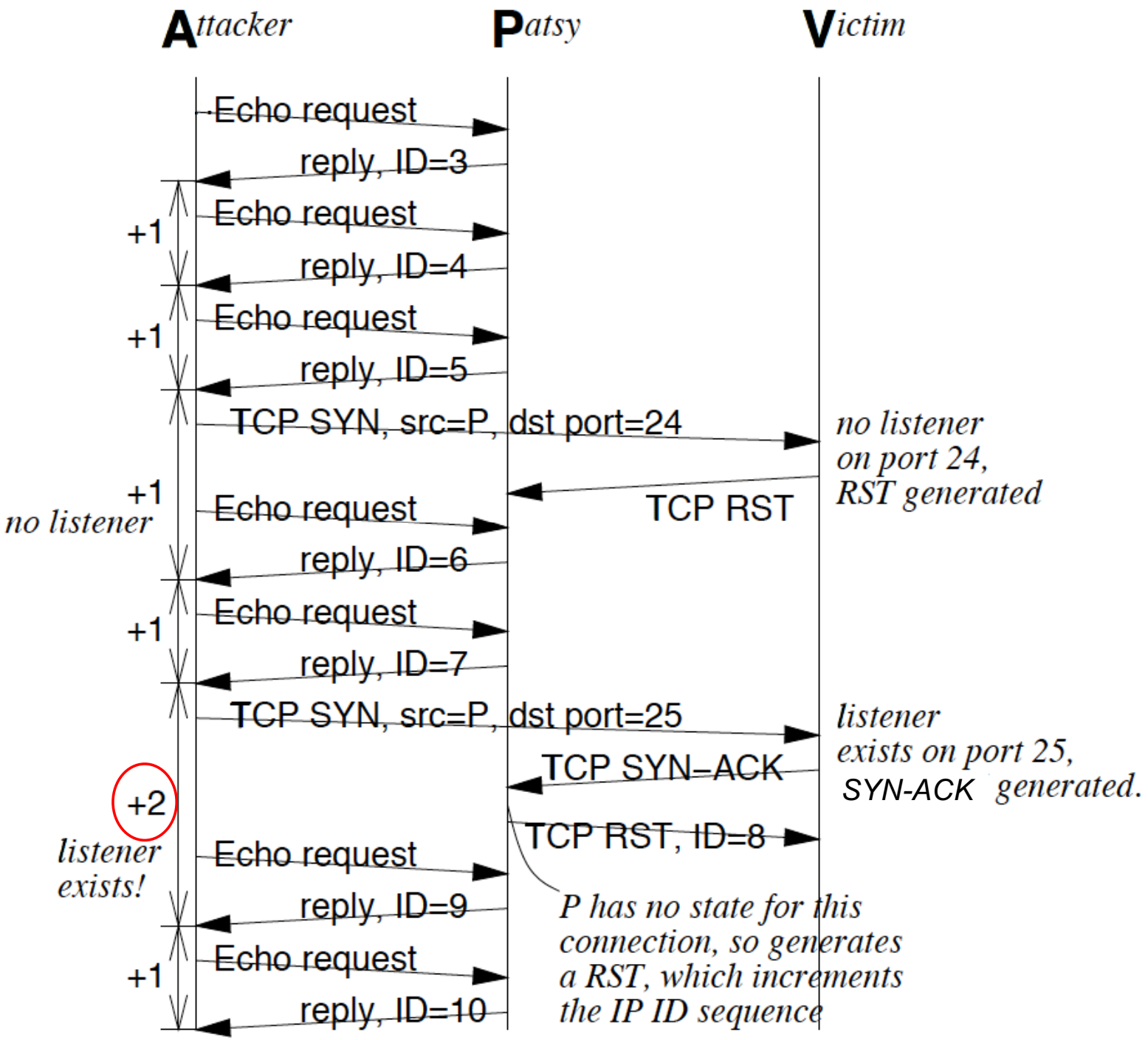


Upon receiving RST, Patsy ignores it and does **nothing**, per TCP spec.


Attacker
Patsy
Victim
Echo request
reply, ID=3
+1
Echo request
reply, ID=4
+1
Echo request
reply, ID=5
TCP SYN, src=P, dst port=24
no listener
on port 24,
RST generated
+1
no listener
Echo request
TCP RST
reply, ID=6
+1
Echo request
reply, ID=7
TCP SYN, src=P, dst port=25
listener
exists on port 25,
TCP SYN–ACK
SYN-ACK generated.

Spoofed


Attacker
Patsy
Victim
Echo request
reply, ID=3
Echo request
+1
reply, ID=4
Echo request
+1
reply, ID=5
TCP SYN, src=P, dst port=24
no listener
on port 24,
RST generated
TCP RST
+1
no listener
Echo request
reply, ID=6
Echo request
+1
reply, ID=7
TCP SYN, src=P, dst port=25
listener
exists on port 25,
SYN-ACK generated.
TCP SYN–ACK
TCP RST, ID=8
P has no state for this
connection, so generates
a RST, which increments
the IP ID sequence


Attacker
Patsy
Victim
Echo request
reply, ID=3
+1
Echo request
reply, ID=4
+1
Echo request
reply, ID=5
TCP SYN, src=P, dst port=24
no listener on port 24, RST generated
TCP RST
+1
no listener
Echo request
reply, ID=6
+1
Echo request
reply, ID=7
TCP SYN, src=P, dst port=25
listener exists on port 25, SYN-ACK generated.
TCP SYN–ACK
+2
listener exists!
TCP RST, ID=8
Echo request
reply, ID=9
P has no state for this connection, so generates a RST, which increments the IP ID sequence
+1
Echo request
reply, ID=10