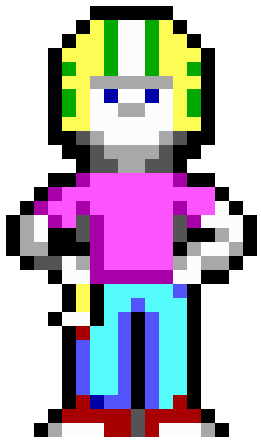
Outbound SPIT Filter with Optimal Performance Guarantees

T. Jung, **S. Martin**, D. Ernst, G. Leduc

Research Unit in Networking, University of Liège

AIMS 2012, Luxembourg, 7th June

3 AM, the (VoIP) phone rings ...



“*we have special pills offers for you. send DEAL on 1337*”

SPIT: SPam over Internet Telephony

is	...	so	...
	automated		CAPTCHA filters
	network aggressive		rate-limiters, blacklisting
	specific software		signature-based filter
	undesired		user reporting

... can handle it.

- Progressive Multi Gray-Leveling (Shin, 2006)
- NEC's SEAL framework (Niccolini, 2006)
- Decision Tree Application-Layer Firewall (Nassar, 2011)

SPIT: SPam over Internet Telephony

is	...	so	...
	automated		CAPTCHA filters
	network aggressive		rate-limiters, blacklisting
	specific software		signature-based filter
	undesired		user reporting

... can handle it.

- Progressive Multi Gray-Leveling (Shin, 2006)
- NEC's SEAL framework (Niccolini, 2006)
- Decision Tree Application-Layer Firewall (Nassar, 2011)

SPIT: SPam over Internet Telephony

yet, we're far from being done

but ...	is ...
CAPTCHA filters	intrusive
rate-limiters	useless against botnets
signature-based filter	easily altered
user reporting	too late ?

... and has high operation cost.

Can't we build something that

- can automatically meet business constraints ?
- require less expertise when updated ?

SPIT: SPam over Internet Telephony

yet, we're far from being done

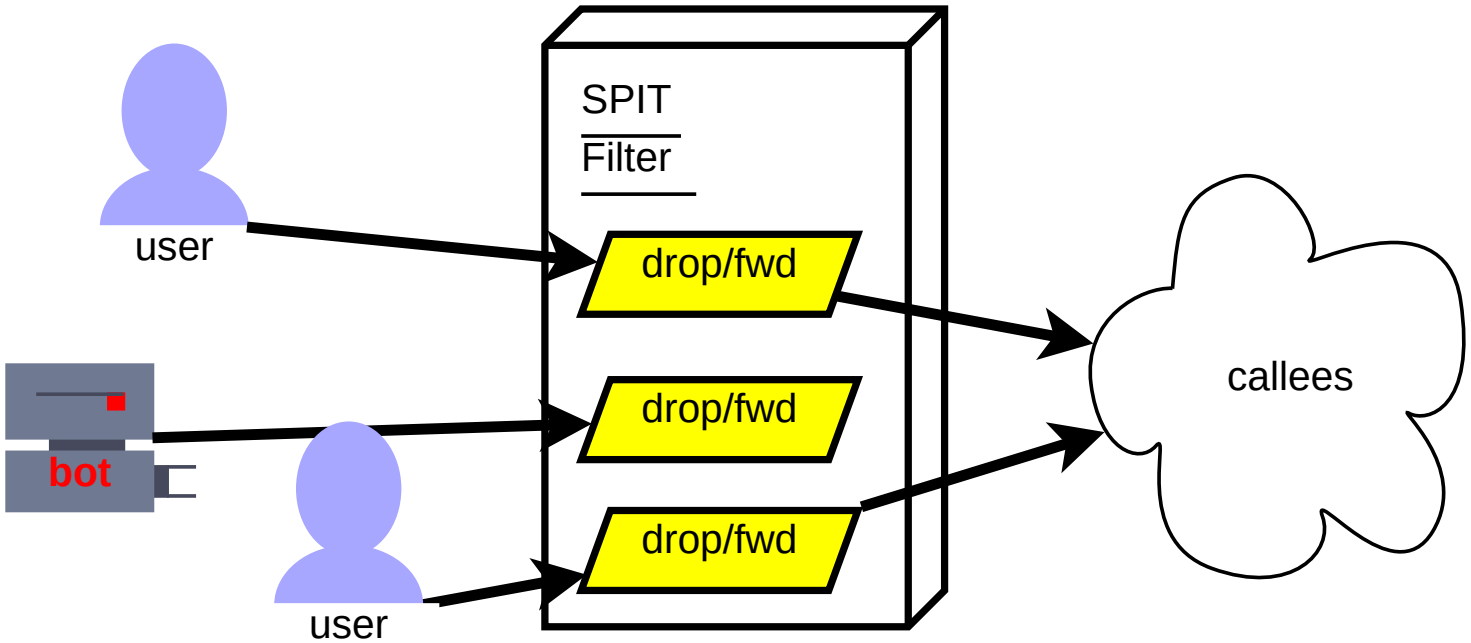
but	...	is	...
	CAPTCHA filters		intrusive
	rate-limiters		useless against botnets
	signature-based filter		easily altered
	user reporting		too late ?

... and has high operation cost.

Can't we build something that

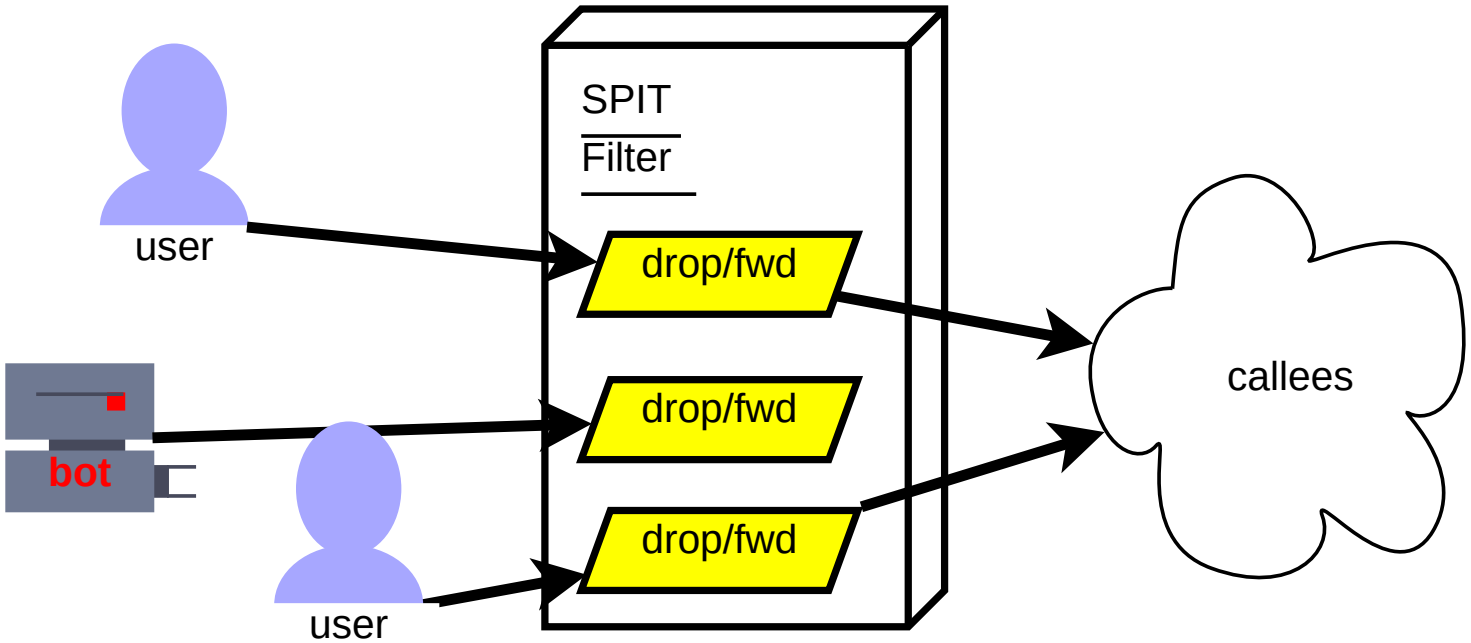
- can automatically meet business constraints ?
- require less expertise when updated ?

Problem Statement



- at call setup, we only observe SIP and IP headers
- merely allow us to recognize sources
- not sufficient to drive the decision w/ required accuracy.
- any hope without explicit user feedback ?

Problem Statement



- at call setup, we only observe SIP and IP headers
- merely allow us to recognize sources
- not sufficient to drive the decision w/ required accuracy.
- any hope without explicit user feedback ?

There's much more info waiting

D. Putz, "Spam on the Phone", 2009

	headers features	call features
	content of INVITE packet	how user reacts to the call
	+ readily available - easy to forge	- during/ after the call + hard to alter
ex:	codec used, user agent, source address	time-to-speech call duration, double-talk ratio
avl:	at call setup time	during / at the end of the call

- "*After third time they all started to just hang up*"
- Some *exploration* of sources would be required,

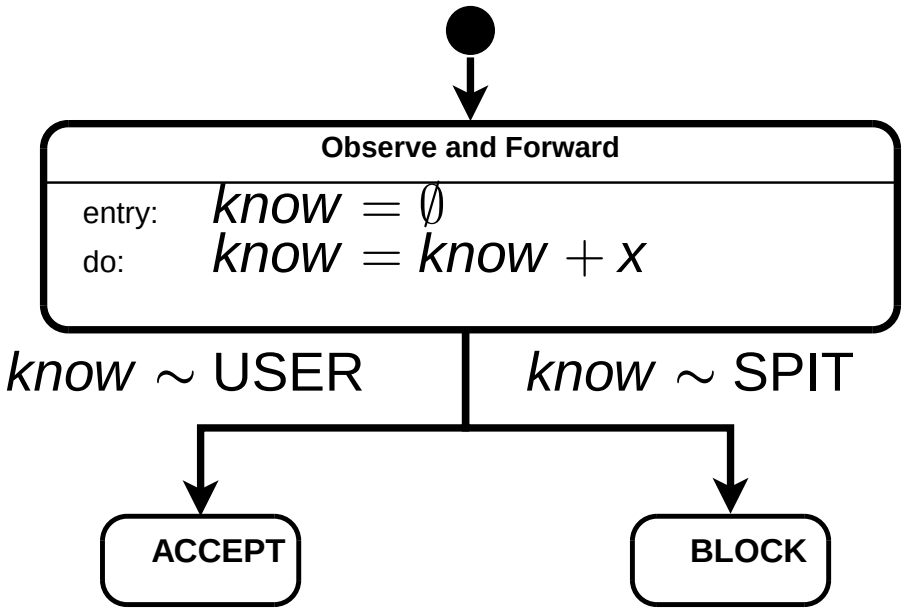
There's much more info waiting

D. Putz, "Spam on the Phone", 2009

	headers features	call features
	content of INVITE packet	how user reacts to the call
	+ readily available - easy to forge	- during/ after the call + hard to alter
ex:	codec used, user agent, source address	time-to-speech call duration, double-talk ratio
avl:	at call setup time	during / at the end of the call

- "*After third time they all started to just hang up*"
- Some *exploration* of sources would be required,

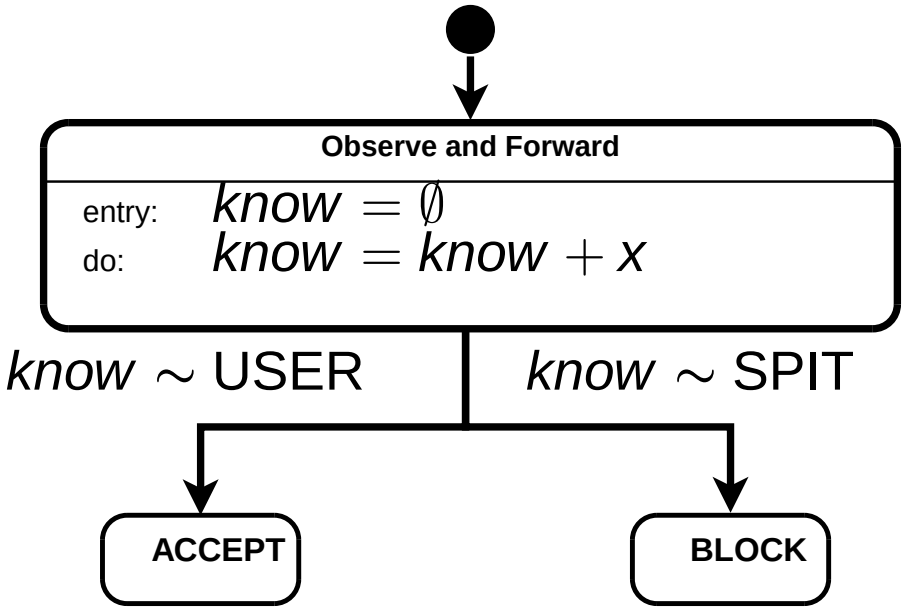
Towards an autonomic filter



- per-source behaviour
- observation x for every call
- compare against a model of SPIT and model of USER calls

- switch to BLOCK or ACCEPT state as soon as confidence is high enough.
- what is *confidence*? when is it *high enough*?

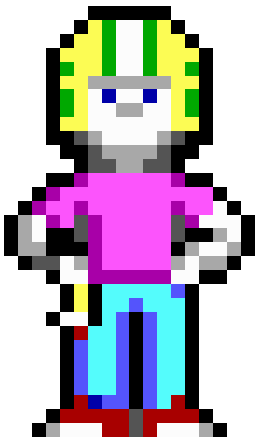
Towards an autonomic filter



- per-source behaviour
- observation x for every call
- compare against a model of SPIT and model of USER calls

- switch to BLOCK or ACCEPT state as soon as confidence is high enough.
- what is *confidence*? when is it *high enough*?

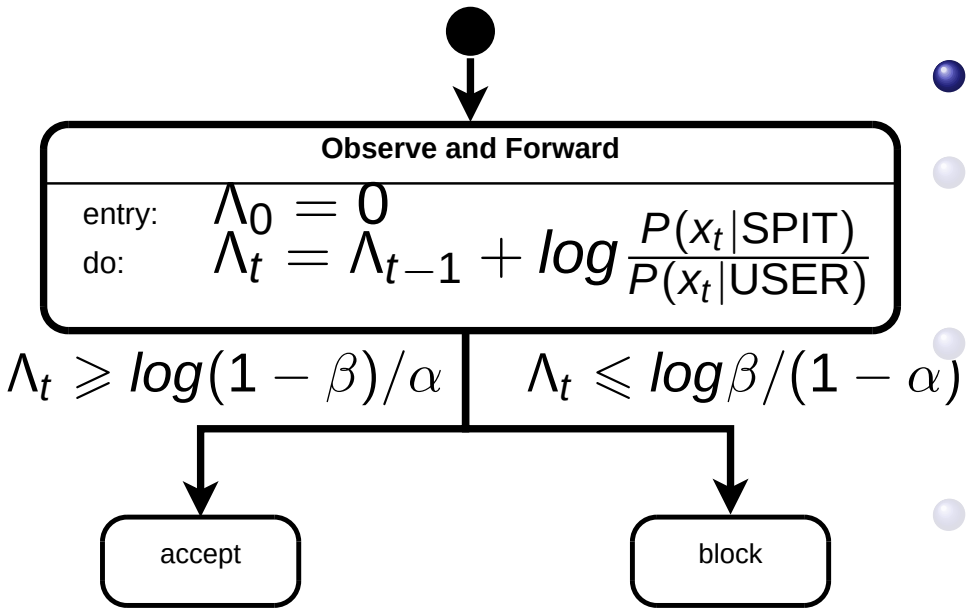
We need guarantees and optimality



Is there any theoretical tool around to offer that ?

Sequential Probability Ratio Testing

[A. Wald '45] to the rescue – P(feature) as model



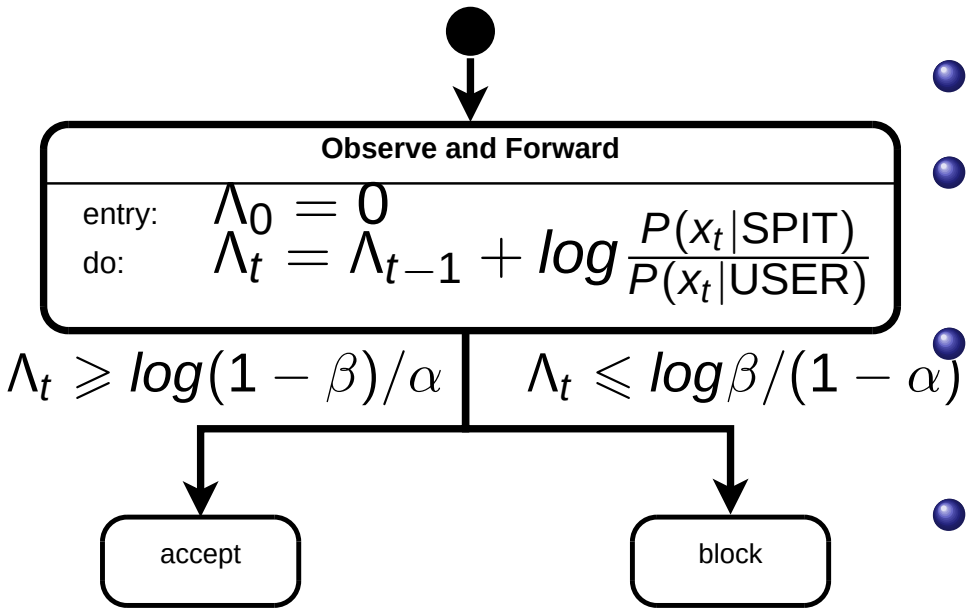
- $P(x|\text{SPIT})$ model
- $P(x|\text{USER})$ model
- small per-invite computation
- small state: one float per source.
- bounded # samples (T_0 aka *Stopping Time*)

Where

- $\alpha =$ tolerated *P*robability (accepting from a SPIT source).
- $\beta =$ tolerated *P*robability (blocking from a USER source).

Sequential Probability Ratio Testing

[A. Wald '45] to the rescue – P(feature) as model



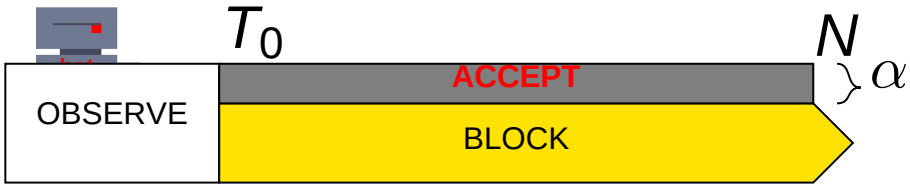
- $P(x|\text{SPIT})$ model
- $P(x|\text{USER})$ model
- small per-invite computation
- small state: one float per source.
- bounded # samples (T_0 aka *Stopping Time*)

Where

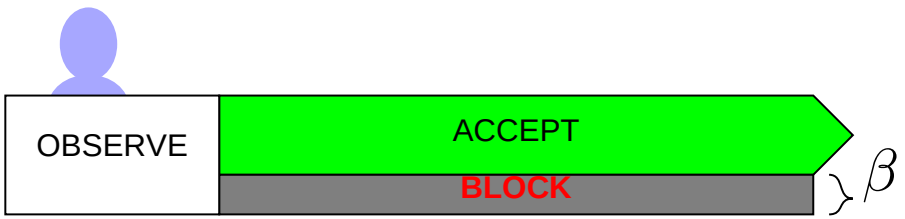
- $\alpha =$ tolerated *P*robability (accepting from a SPIT source).
- $\beta =$ tolerated *P*robability (blocking from a USER source).

Expected Loss – Turning SPRT autonomic

or "how to avoid defining α and β by hand"



$$c_S = c_A T_0 + \alpha c_A (N - T_0)$$



$$c_U = \beta c_B (N - T_0)$$

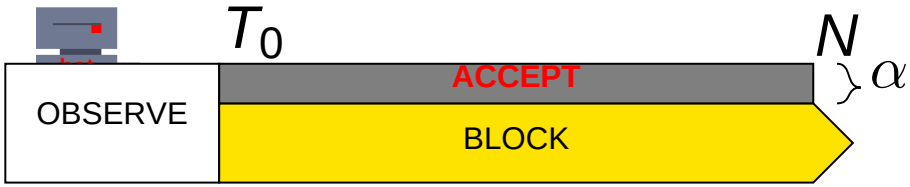
- No tuning: just costs!
- c_A = co$t/accepted SPIT call
- c_B = co$st/blocked USER call
- N = autonomic horizon = # calls before human fix

$$E(Loss) = P(\text{SPIT}) c_S + P(\text{USER}) c_U = f(\alpha, \beta) \qquad (1)$$

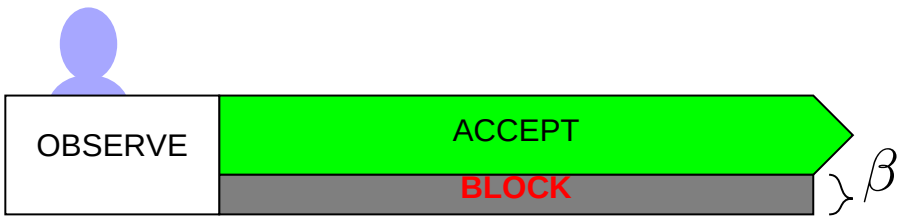
- defines cummulated error over N calls
- minimizing *Loss* provides optimal $\alpha^\star$ and $\beta^\star$ for the filter
- thanks to the *stopping time* guarantee of SPRT

Expected Loss – Turning SPRT autonomic

or "how to avoid defining α and β by hand"



$$c_S = c_A T_0 + \alpha c_A (N - T_0)$$



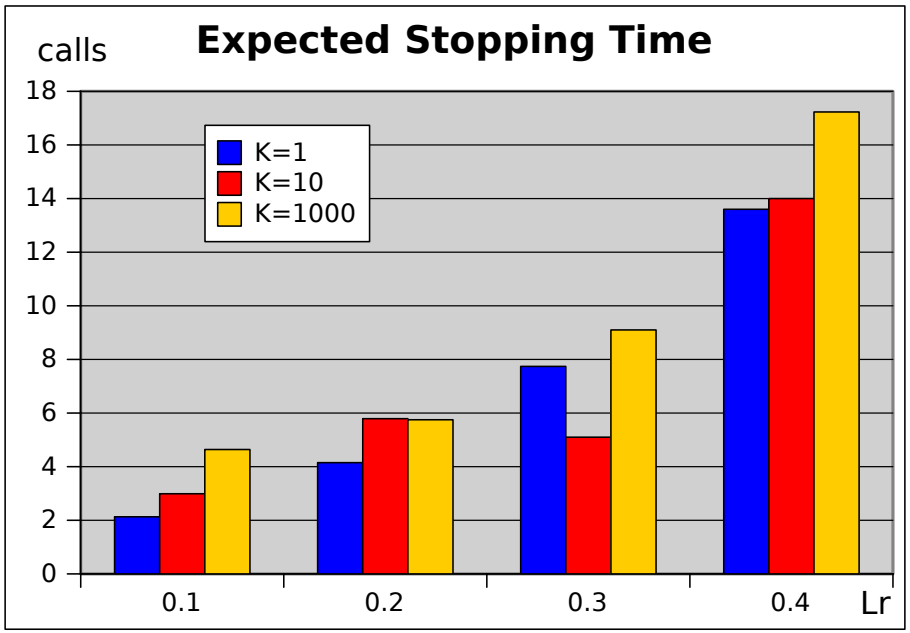
$$c_U = \beta c_B (N - T_0)$$

- No tuning: just costs!
- c_A = co$t/accepted SPIT call
- c_B = co$st/blocked USER call
- N = autonomic horizon = # calls before human fix

$$E(Loss) = P(\text{SPIT}) c_S + P(\text{USER}) c_U = f(\alpha, \beta) \qquad (1)$$

- defines cummulated error over N calls
- minimizing *Loss* provides optimal $\alpha^\star$ and $\beta^\star$ for the filter
- thanks to the *stopping time* guarantee of SPRT

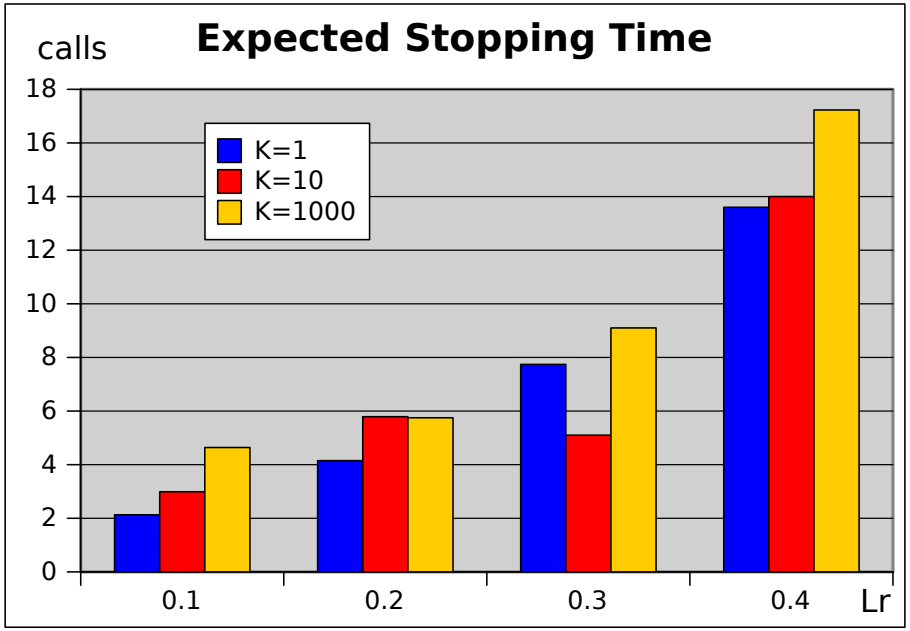
Example : exponential call duration



- $P(x|\text{USER}) = EXP(\lambda_U)$;
- $P(x|\text{SPIT}) = EXP(\lambda_S)$;
- compute $\alpha^\star$ and $\beta^\star$;
- and that's *guaranteed* (worst case) bounds.

- relative costs: $c_B = k c_A$;
- from Kullback-Leibler numbers, performance is mostly dependent on the $Lr = \lambda_U / \lambda_S$ ratio.

Example : exponential call duration

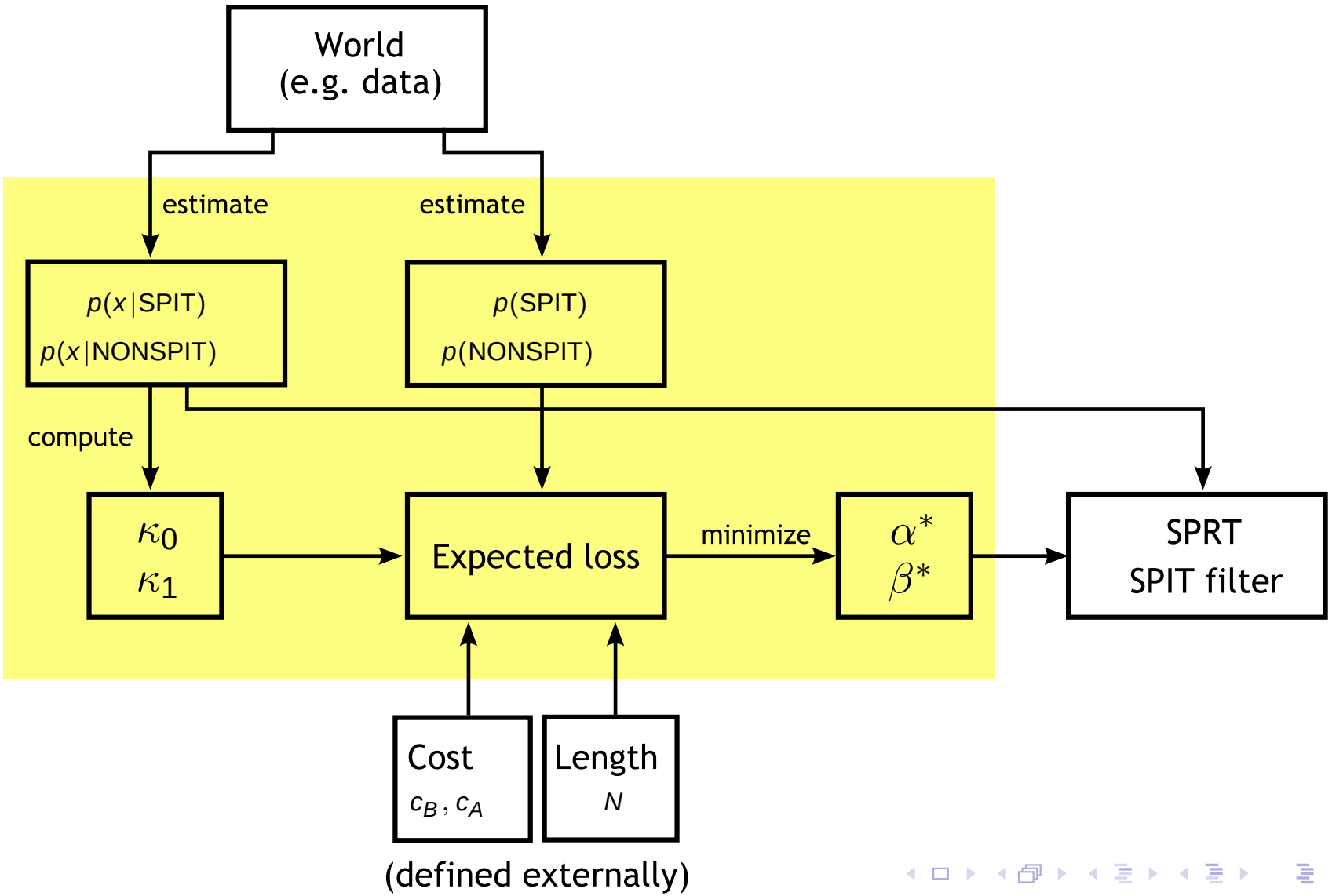


- $P(x|\text{USER}) = EXP(\lambda_U)$;
- $P(x|\text{SPIT}) = EXP(\lambda_S)$;
- compute $\alpha^\star$ and $\beta^\star$;
- and that's *guaranteed* (worst case) bounds.

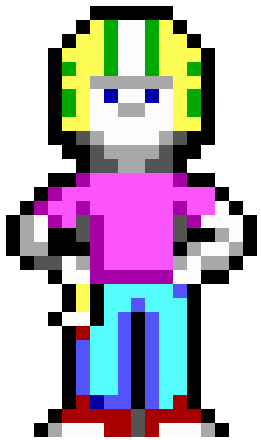
- relative costs: $c_B = k c_A$;
- from Kullback-Leibler numbers, performance is mostly dependent on the $Lr = \lambda_U/\lambda_S$ ratio.

summary

yellow box = automatic re-computation

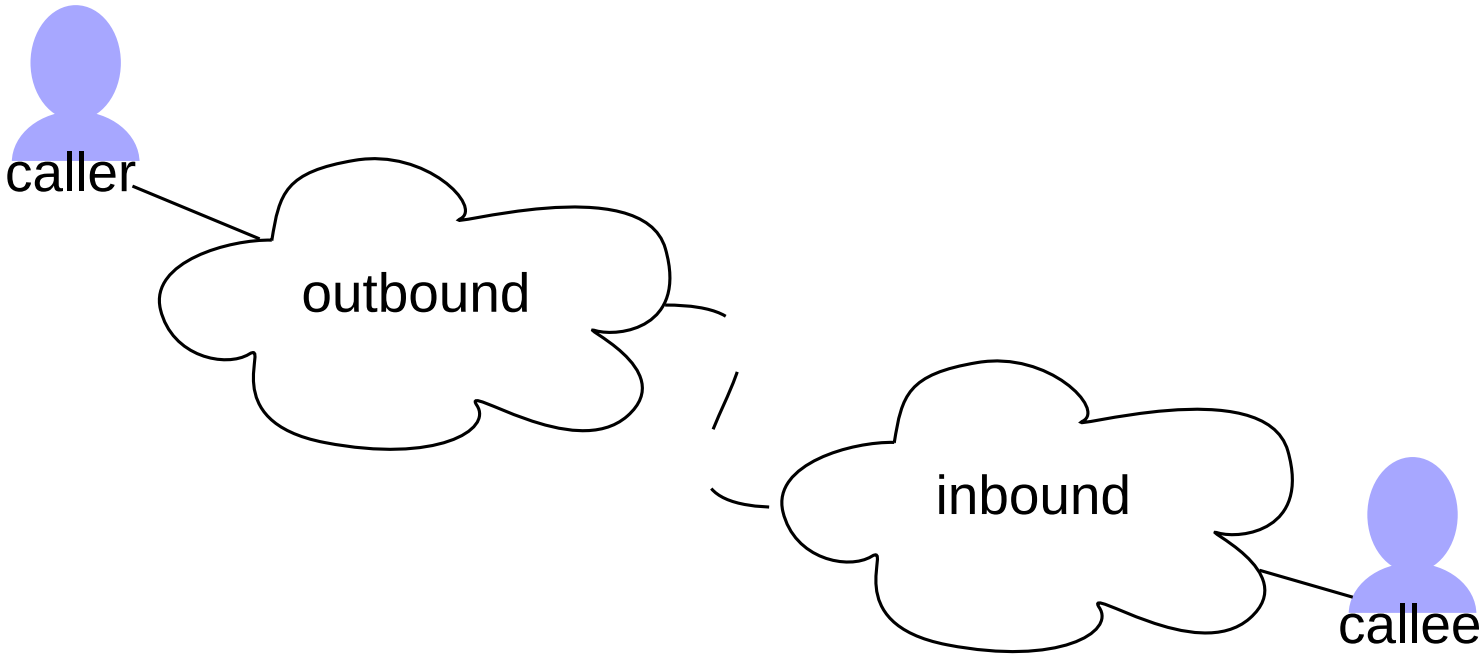


Welcome to the Real World



What should be done to *apply* this ?

What defines the “source” of a call ?

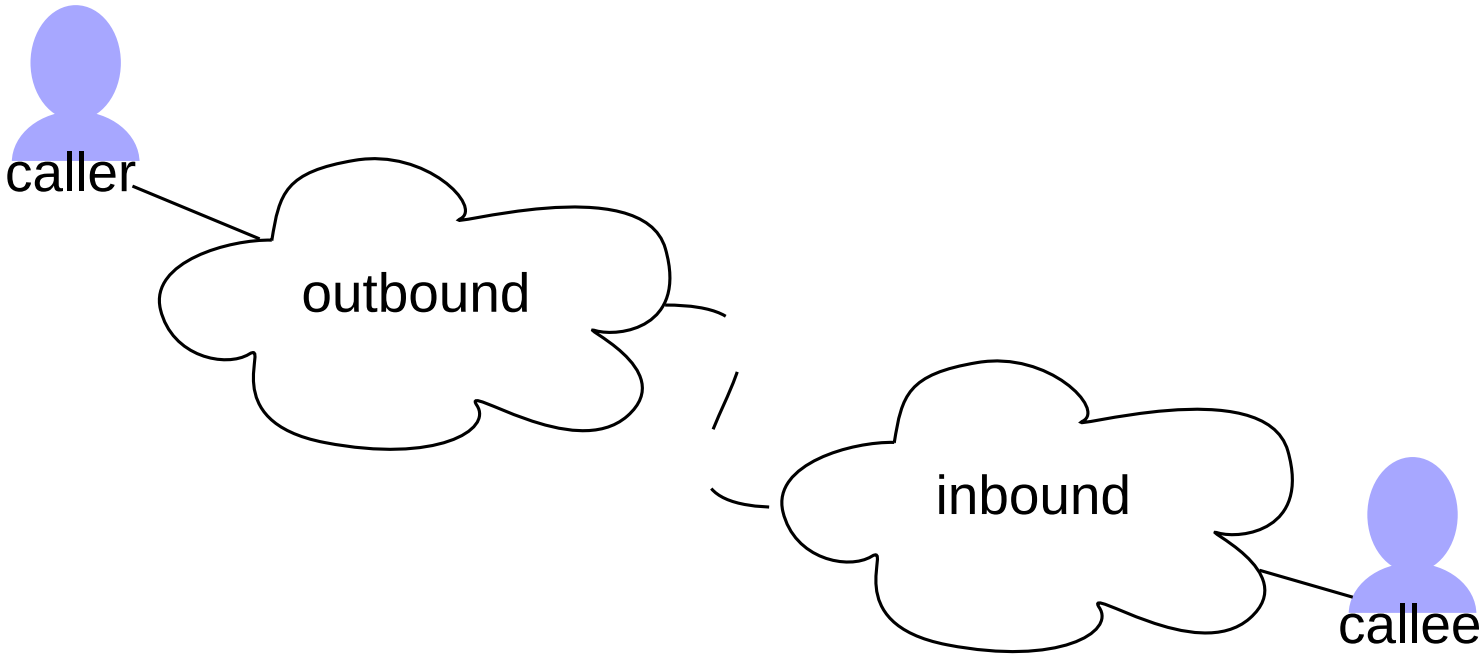


user PIN code
device UID

IP address <-> mobility, NAT
SIP address <-> generator
agent signature <-> generator

- Applying to inbound filtering is still an open question.
- Applying to outbound filtering can save ISP’s reputation.

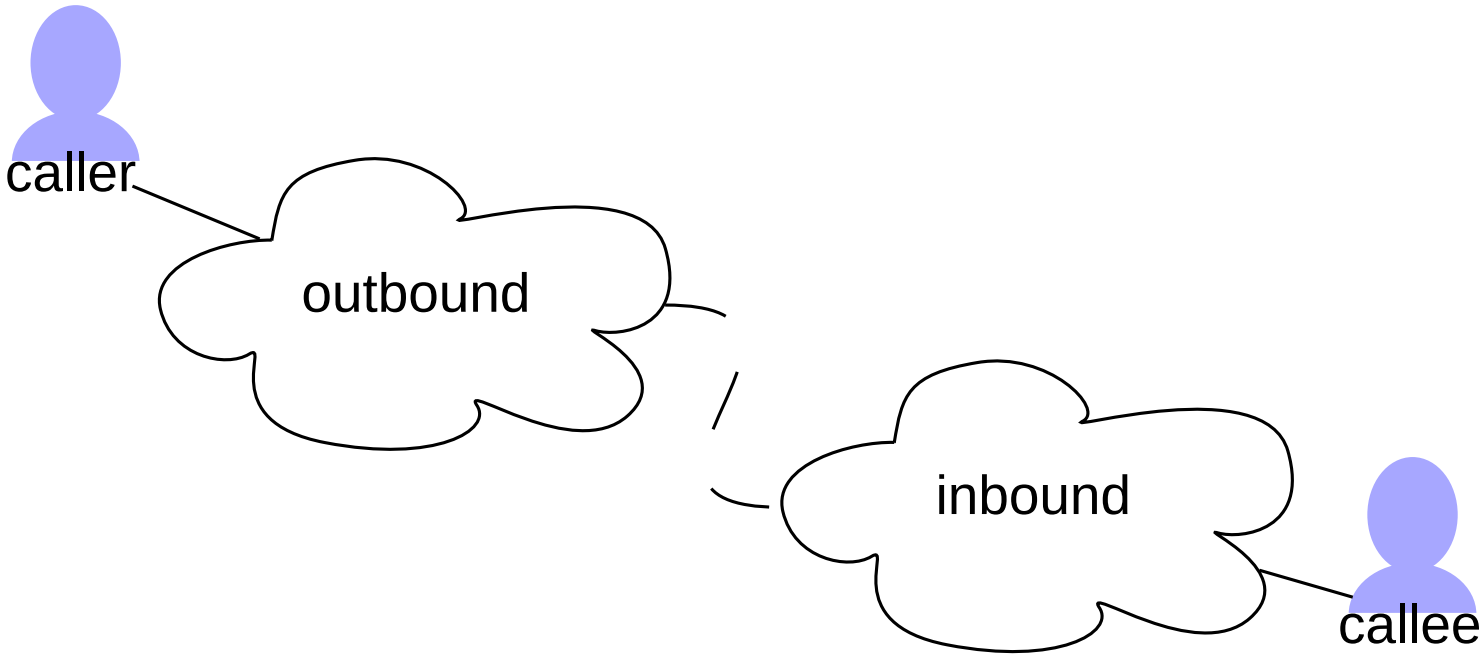
What defines the “source” of a call ?



user PIN code	IP address	<->	mobility, NAT
device UID	SIP address	<->	generator
	agent signature	<->	generator

- Applying to inbound filtering is still an open question.
- Applying to outbound filtering can save ISP’s reputation.

What defines the “source” of a call ?

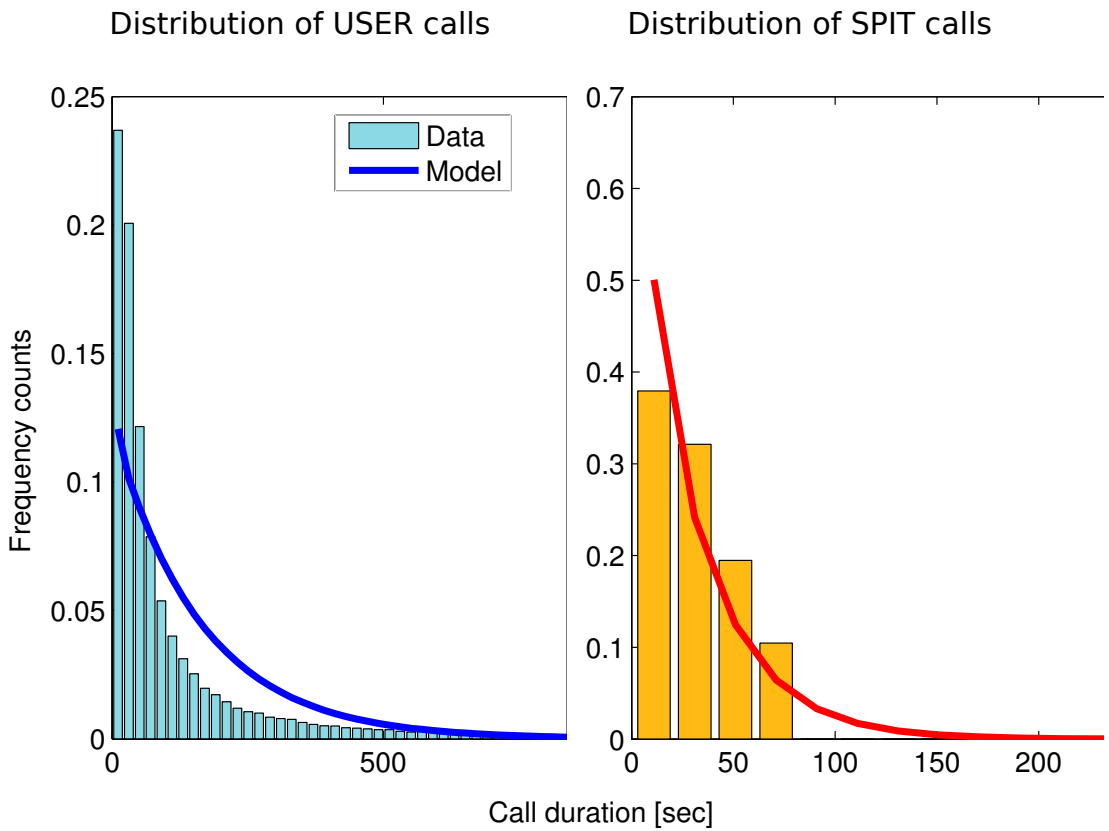


user PIN code	IP address	<->	mobility, NAT
device UID	SIP address	<->	generator
	agent signature	<->	generator

- Applying to inbound filtering is still an open question.
- Applying to outbound filtering can save ISP’s reputation.

Estimating Distributions

Example crafted from the “Reality” dataset, Eagle et al., MIT, 2009



- simple model: max-likelihood fitting to exponential
- sample 20% of the “short” calls to build the SPIT dataset.
- exponential law underestimate short calls in USER set

results (# SPIT sources = # USER sources)

		$c_1 = c_0$	$c_1 = 5c_0$	$c_1 = 100c_0$
N=10	α^*	1.00e-06	1.00e-06	1.00e-01
	β^*	1.00e-01	1.65e-02	8.60e-05
	$T/E[T]$	4.1/3.3	5.6/4.	6.9/4.
	RErr(SPIT)	0.030%	0.788%	70.421%
	Rerr(USER)	33.439%	15.883%	0.671%
N=15	$T/E[T]$	4.5/3.4	6.5/4.8	7.9/5.0
	RErr(SPIT)	0.000%	0.006%	6.593%
	RErr(USER)	33.480%	14.569%	1.642%
N=20	α^*	1.00e-06	1.00e-06	1.00e-01
	β^*	6.35e-02	6.60e-03	4.35e-05
	$T/E[T]$	5.0/3.6	7.1/5.0	8.1/5.1
	RErr(SPIT)	0.000%	0.000%	0.163%
	RErr(USER)	30.660%	13.134%	1.940%
N=100	α^*	1.00e-06	1.00e-06	1.00e-06
	β^*	1.13e-02	1.13e-03	1.13e-05
	$T/E[T]$	6.9/4.7	8.6/6.0	11.5/8.5
	RErr(SPIT)	0.000%	0.000%	0.000%
	RErr(USER)	16.886%	7.659%	1.458%

- simulation stopping times = 140% theoretical bound
- poor estimation affects accuracy: 2% USER calls blocked.

results (# SPIT sources = # USER sources)

		$c_1 = c_0$	$c_1 = 5c_0$	$c_1 = 100c_0$
N=10	α^*	1.00e-06	1.00e-06	1.00e-01
	β^*	1.00e-01	1.65e-02	8.60e-05
	$T/E[T]$	4.1/3.3	5.6/4.	6.9/4.
	RErr(SPIT)	0.030%	0.788%	70.421%
	Rerr(USER)	33.439%	15.883%	0.671%
N=15	$T/E[T]$	4.5/3.4	6.5/4.8	7.9/5.0
	RErr(SPIT)	0.000%	0.006%	6.593%
	RErr(USER)	33.480%	14.569%	1.642%
N=20	α^*	1.00e-06	1.00e-06	1.00e-01
	β^*	6.35e-02	6.60e-03	4.35e-05
	$T/E[T]$	5.0/3.6	7.1/5.0	8.1/5.1
	RErr(SPIT)	0.000%	0.000%	0.163%
	RErr(USER)	30.660%	13.134%	1.940%
N=100	α^*	1.00e-06	1.00e-06	1.00e-06
	β^*	1.13e-02	1.13e-03	1.13e-05
	$T/E[T]$	6.9/4.7	8.6/6.0	11.5/8.5
	RErr(SPIT)	0.000%	0.000%	0.000%
	RErr(USER)	16.886%	7.659%	1.458%

- simulation stopping times = 140% theoretical bound
- poor estimation affects accuracy: 2% USER calls blocked.

results (# SPIT sources = # USER sources)

		$c_1 = c_0$	$c_1 = 5c_0$	$c_1 = 100c_0$
N=10	α^*	1.00e-06	1.00e-06	1.00e-01
	β^*	1.00e-01	1.65e-02	8.60e-05
	$T/E[T]$	4.1/3.3	5.6/4.	6.9/4.
	RErr(SPIT)	0.030%	0.788%	70.421%
	Rerr(USER)	33.439%	15.883%	0.671%
N=15	$T/E[T]$	4.5/3.4	6.5/4.8	7.9/5.0
	RErr(SPIT)	0.000%	0.006%	6.593%
	RErr(USER)	33.480%	14.569%	1.642%
N=20	α^*	1.00e-06	1.00e-06	1.00e-01
	β^*	6.35e-02	6.60e-03	4.35e-05
	$T/E[T]$	5.0/3.6	7.1/5.0	8.1/5.1
	RErr(SPIT)	0.000%	0.000%	0.163%
	RErr(USER)	30.660%	13.134%	1.940%
N=100	α^*	1.00e-06	1.00e-06	1.00e-06
	β^*	1.13e-02	1.13e-03	1.13e-05
	$T/E[T]$	6.9/4.7	8.6/6.0	11.5/8.5
	RErr(SPIT)	0.000%	0.000%	0.000%
	RErr(USER)	16.886%	7.659%	1.458%

- simulation stopping times = 140% theoretical bound
- poor estimation affects accuracy: 2% USER calls blocked.

How can we improve ?

... if 1% of “USER calls blocked” is too much

Technical solutions:

- better statistical model for estimating call distributions.
- better features (e.g. time-to-speech, double-talk ratio, etc.),
  - or combine features.
  - use Kullback-Leibler numbers to evaluate potential of the models.

Safety net:

- use an alternative “block” action, with lower c_B
- e.g. audio CAPTCHA, access voicemail only, etc.

Conclusions

SPRT ...

- can filter SPIT from a history of previous calls;
- cheap on state and computation;
- theoretical bounds on error rates and stopping time;
- replaces ad-hoc tuning with cost-driven optimization.

BUT ...

- it requires a good model of your traffic statistics;
- provides no model of a compromised source (both USER and SPIT)
- inbound-filtering is still an open issue.

Questions?

Anyone ?

www.ulg.ac.be

www.resumenet.eu