Pushing on String: Adventures in the 'Don't Care' Regions of Password Strength

Cormac Herley
Microsoft Research, Redmond
@cormacherley

Joint work with Dinei Florêncio and Paul C. van Oorschot

Two recent studies:

1. Managing a ***portfolio*** of passwords

2. Administering password-protected site

Ch1: Password Portfolios:

Sustainably Managing Large Numbers of Accounts

Choosing a password

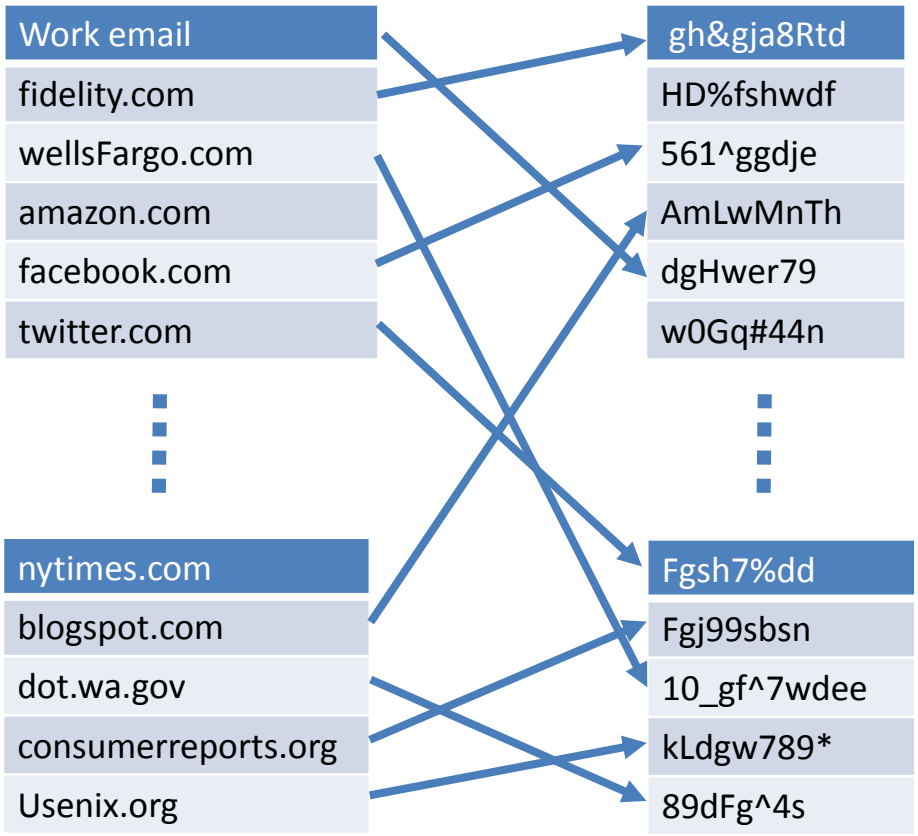
Everyone knows

A1: Passwords should be random and strong

A2: Passwords should not be re-used across accounts

But no-one does.

Portfolio of N random, unique passwords lg(S) each



lg(N!) + N·lg(S)

Must remember:

- N passwords = N·lg(S)
- NxN pwd-to-acct assignment = lg(N!)

E(N) = N·lg(S) + lg(N!)

N=100 *random* passwords of lg(S) bits

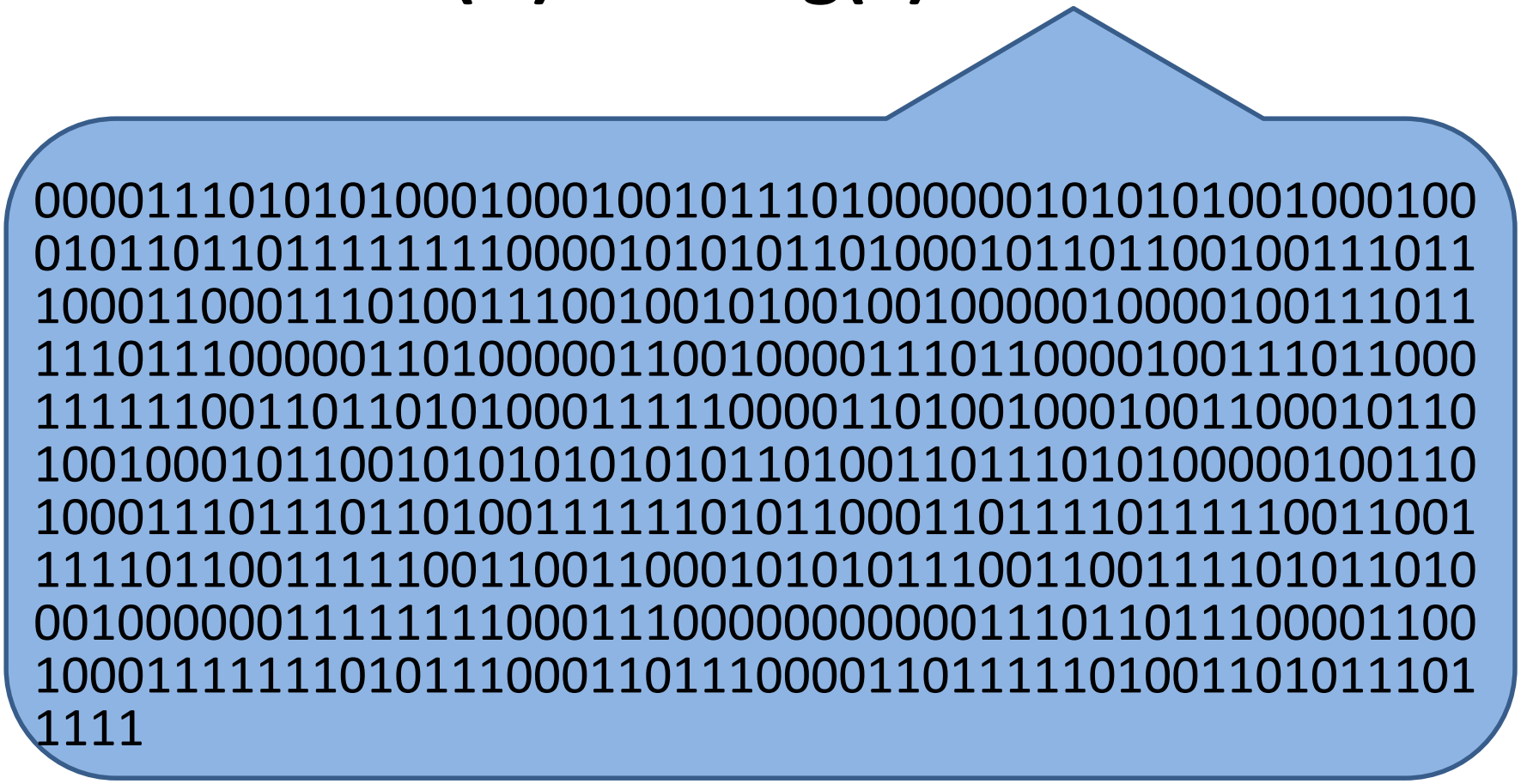
$$E(N) = N \cdot \lg(S) + \lg(N!)$$

$$= 4000 + 524 = 4524 \text{ bits}$$

Depends how remember passwords

Random bits

$E(N) = 100 \cdot \lg(S) + 524$



Claim: memorization task is impossible

N accounts in G groups

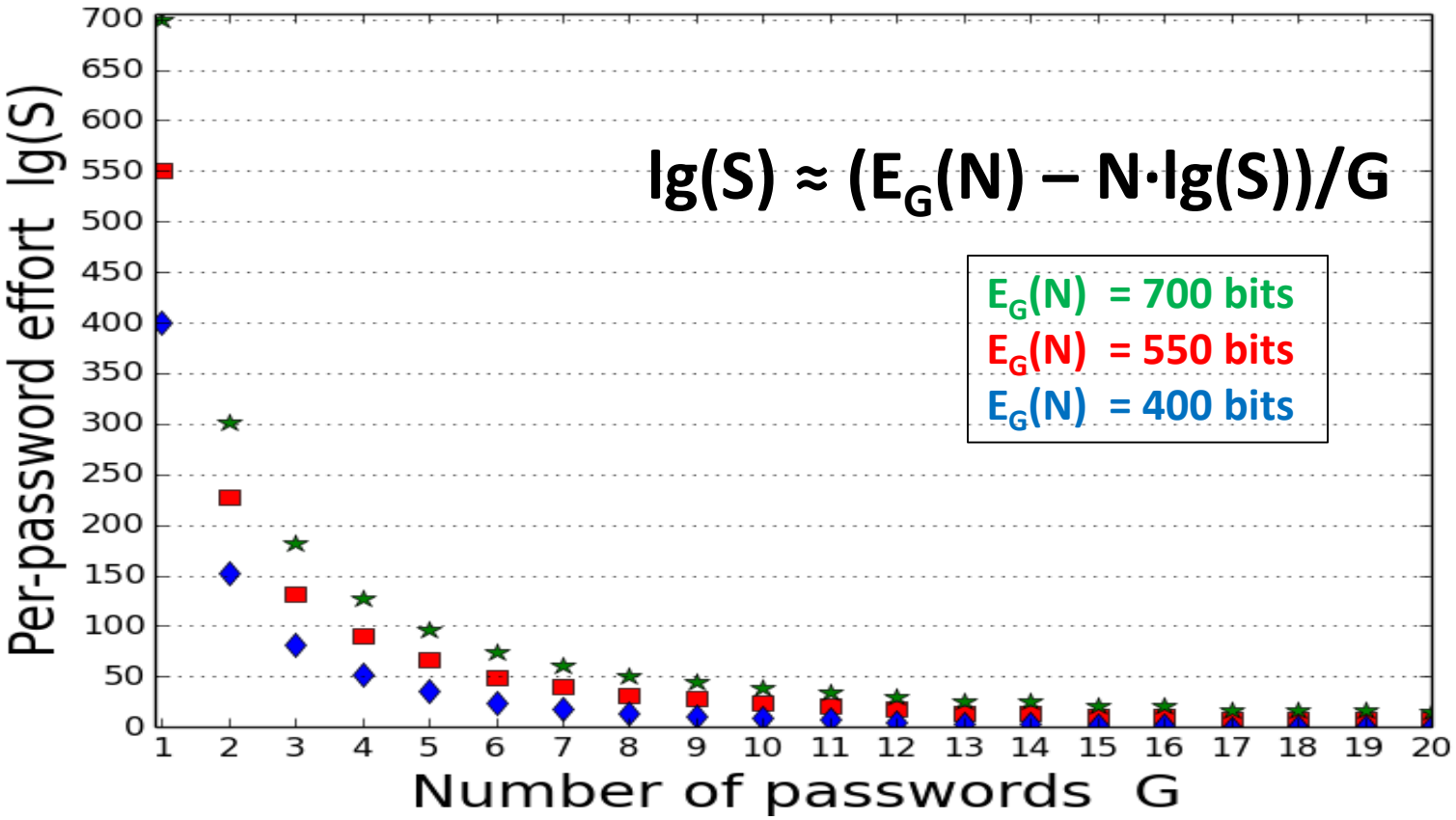
$$E_G(N) \approx G \cdot \lg(S) + N \cdot \lg(G)$$

$$\Rightarrow \quad \lg(S) \approx \frac{(EG(N) - N \cdot \lg(G))}{G}$$

Tradeoff between strength and avoiding re-use (i.e. lg(S) and G)

N = #accts
G = #unique pwds
lg(S) = pwd strength

Many ways to organize portfolio:
(e.g. 4 groups of 25, 5 groups of 20)



Fixed effort:

- lg(S) α 1/G
- Stronger pwd => more re-use

Over-constrained Problems

- Password Portfolios
  - Insisting on the necessity of impossible things

- How end up over-constrained?

A	Is re-use a real threat vector?	Y
B	Do bad things happen because of re-use?	Y
C	Can we eliminate that risk by avoiding re-use?	Y
D	Does it follow that you should not re-use?	N

$\mathbf{X} \Rightarrow \mathbf{Y}$ does not mean $\bar{\mathbf{X}} \Rightarrow \bar{\mathbf{Y}}$

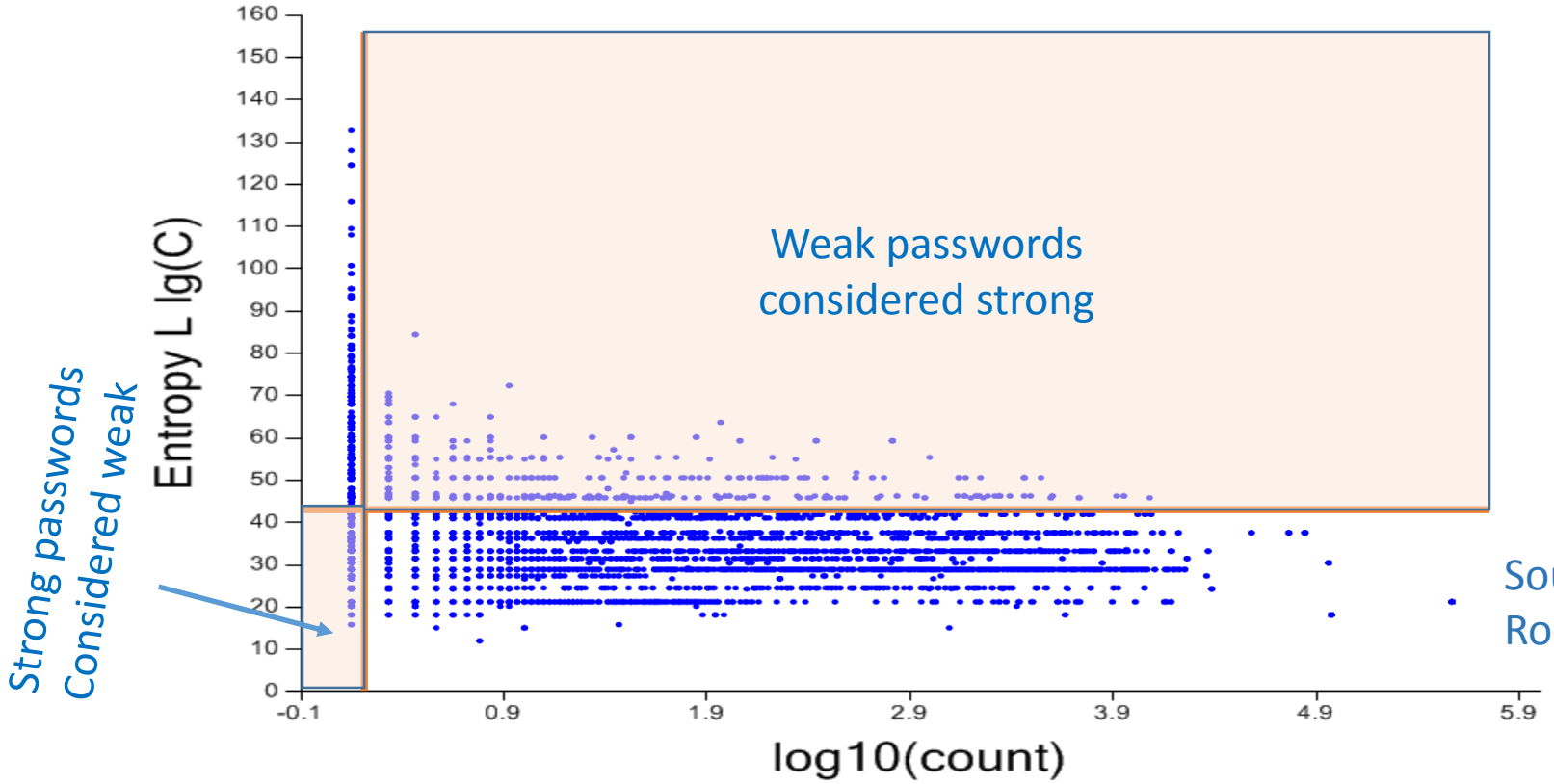
Take-aways on Chap.I

- One password/account impossible as portfolio grows.
- Inherent tradeoff between re-use/strength.
- A strategy that rules out re-use is sub-optimal
- A strategy that rules out weak passwords is sub-optimal

Ch2: Administering a password-protected site

- **Why do we want strength?**
  - Want to deny access to bad guys
- **How much strength do we need?**
  - More. More. More.
- **Does more strength always help deny access?**
  - No. Even against guessing attacks.

Measure strength of a password? Don't use entropy = L·lg(C)



Source: Rockyou leak

- L·lg(C) not even approximately monotonic in frequency
- Partial Guess numbers: #guesses to get fraction α of accts (e.g. Bonneau measure)

Measure strength of a distribution?

Administrator's task: defend the population

- With limited ability to shape the distribution what should you do?

How much strength do we need?

Two very different guessing attacks

- **Online:** computed on defender's HW
  - Lockout, rate-limiting, forensics, …..

- **Offline:** computed on attacker's HW
  - Limited only by hardware
  - *Needs to steal the hashed file*

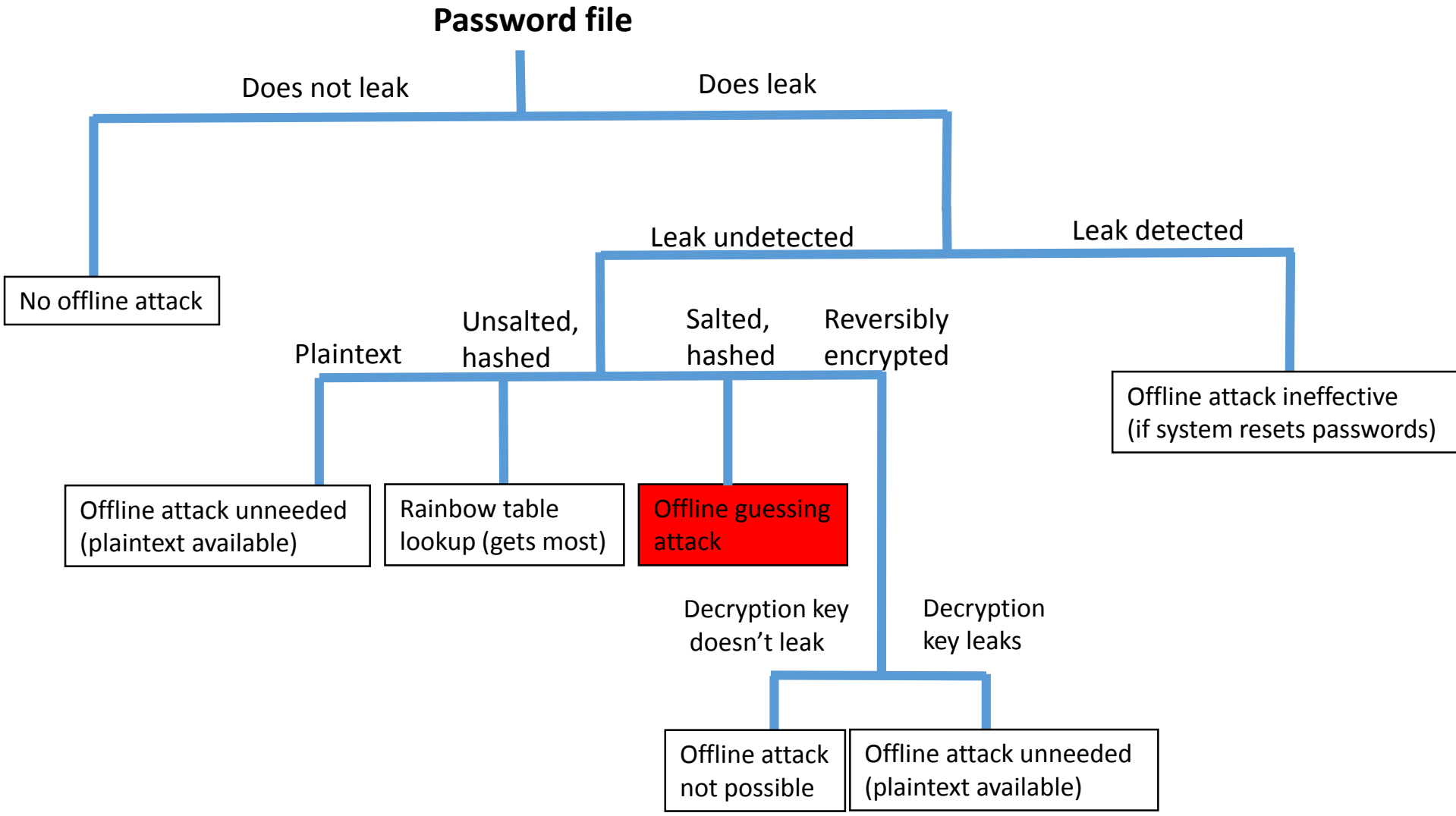
When strength has no influence

When is guessing a factor?

“The success of database breaches, client-side malware, phishing and network-sniffing are entirely unaffected by password choice.”

E.g. Rockyou database breach: password choice had no effect on the outcome

When is *offline* guessing a factor?



Plaintext or reversibly encrypted: steps to go beyond online attacks unjustifiable—no offline guessing attack.

Recent breaches

Site	Year	# Accounts	Hashed	Salted	Reversibly Encrypted	Offline guessing attack beyond rainbow tables needed and possible
Rockyou [64]	2009	32m				N
Gawker	2010	1.3m	✓	✓		Y
Tianya	2011	35m				N
eHarmony	2012	1.5m	✓			N
LinkedIn	2012	6.5m	✓			N
Evernote	2013	50m	✓	✓		Y
Adobe	2013	150m			✓	N
Cupid Media	2013	42m				N

- August 2014: 1.2 billion CyberVor set: plaintext

- **In only 2 leaks (Evernote, Gawker) and 51.3mln ex 1.5bln passwords was there an offline threat.**

Online-offline chasm

How many guesses?

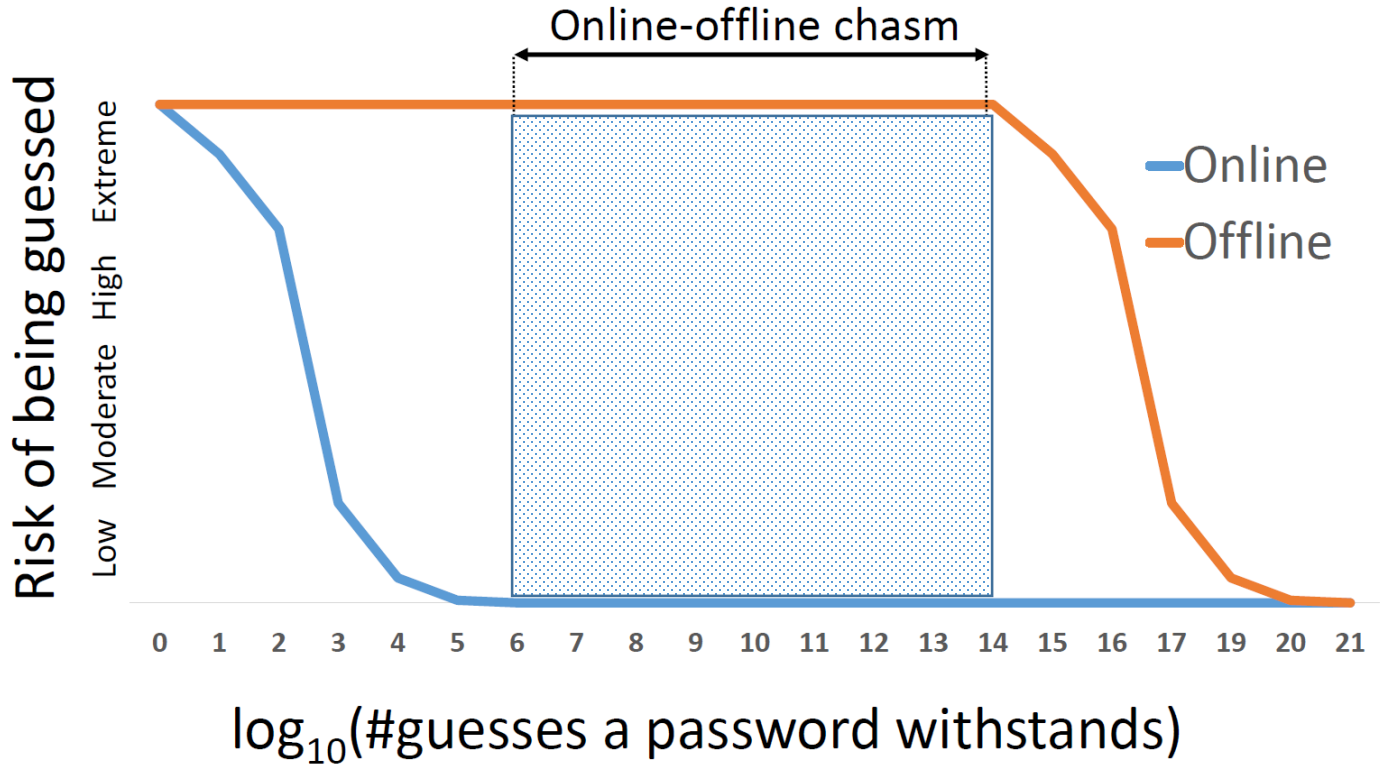
Attack	Type	Guesses	Example
Online	Breadth-first	10^4	“6387”
Online	Depth-first	10^6	“tincan24”
Offline	Breadth-first	10^{14}	“7Qr&2Mu”
Offline	Depth-first	10^{20}	“eTh^D#aW3a8”

Note the enormous difference needed to withstand online/offline

Reasoning (salted, hashed, no iteration, 4 mos campaign):

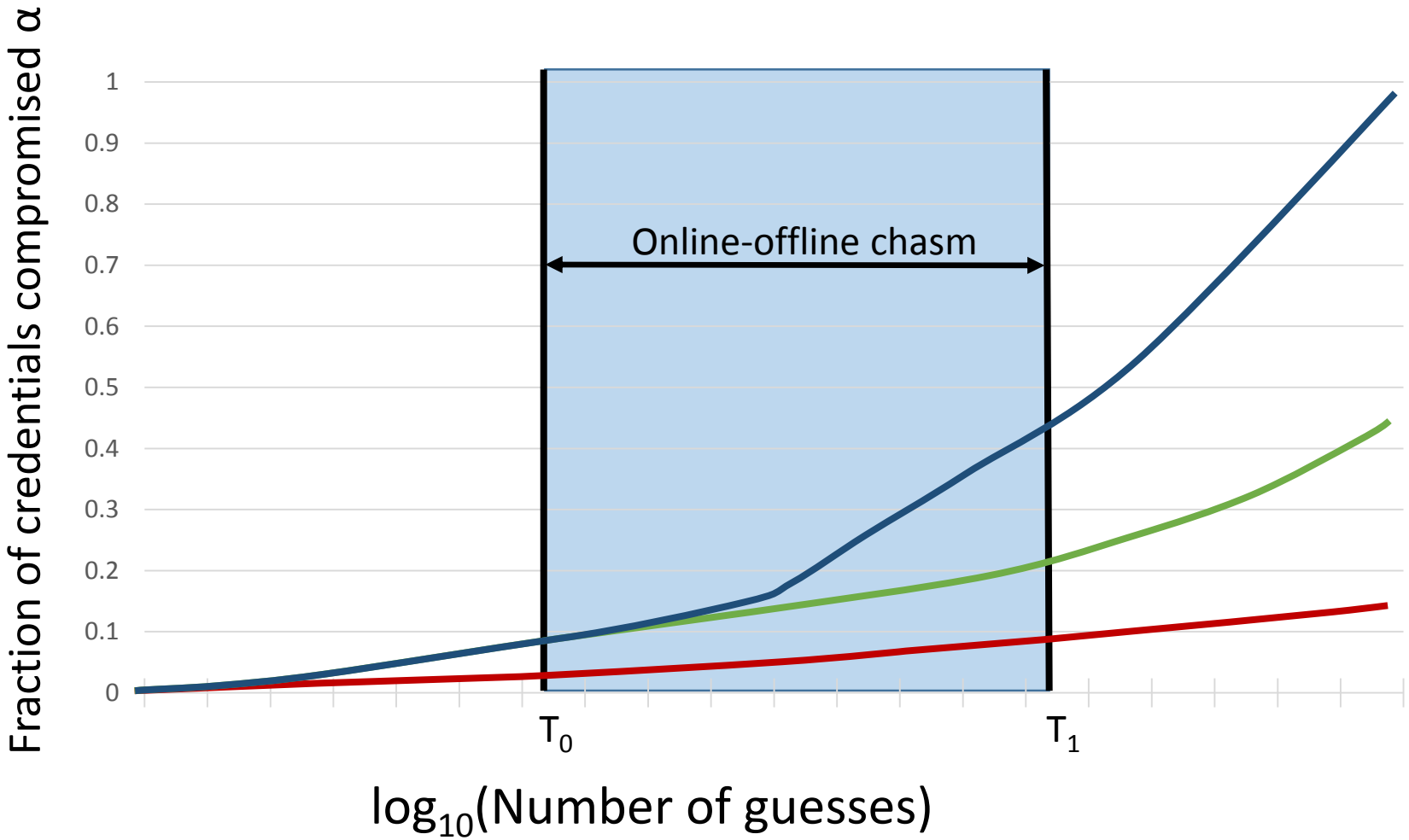
- **Online Breadth-first: 10^4**
  - Over 4 mos. 17300x more fail events than legit pop. (assuming 1 legit login/user/day, 5% fail rate)
- **Online Depth-first: 10^6**
  - Lockout or Rate-limit requests, IP blocking
- **Offline Breadth-first: 10^{14}**
  - 1000 GPUs @ 10^{10} guess/sec against 10^6 accts for 4 mos
- **Offline Depth-first: 10^{21}**
  - 1000 GPUs @ 10^{10} guess/sec against 10 accts for 4 mos

No gain in exceeding online threshold while falling short of offline one.



Chasm is 8 orders of magnitude wide!!

Passwords between T_0 and T_1 do too much and not enough



No security improvement between T_0 and T_1

Compromise Saturation Point

Q: if an attacker has 20% of credentials, are you 20% owned or fully owned?

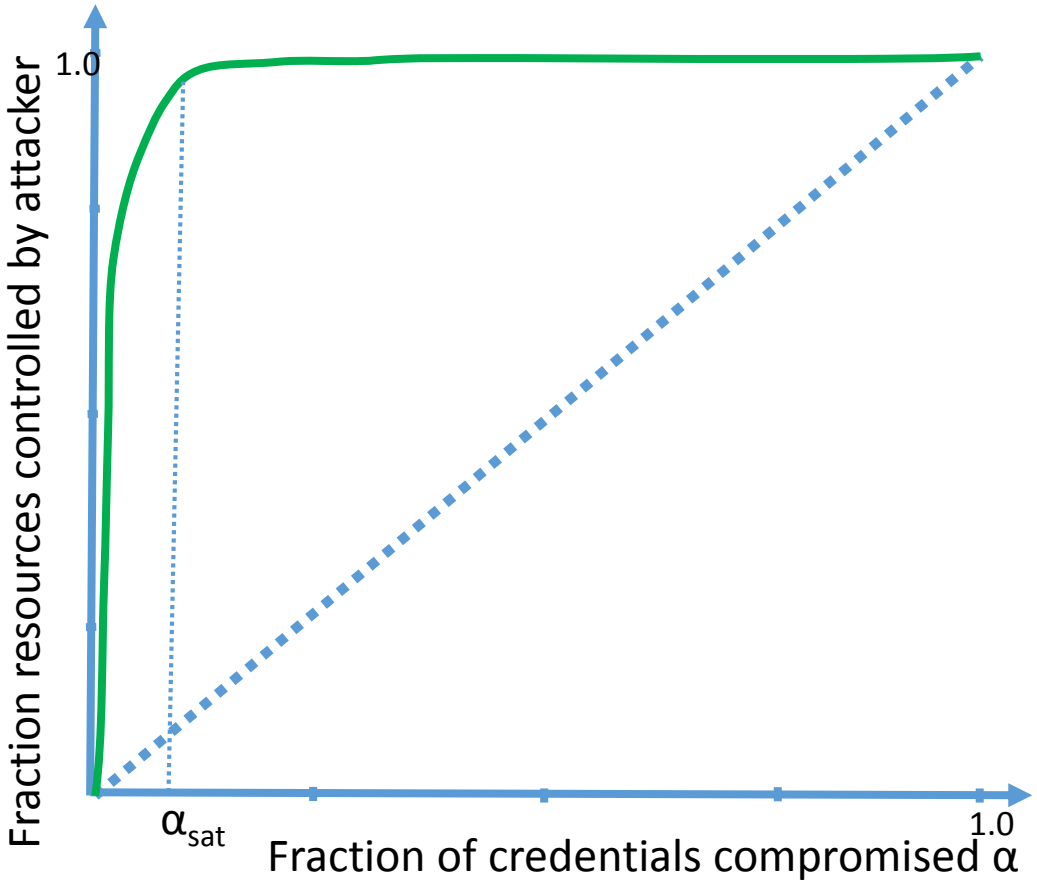
- **RSA breach:**
  - Phishing “two small groups of employees none of whom were particularly high profile or high value”
- **NSA:**
  - Snowden
- **Snowball attacks:**
  - 98.1% of machines allowed snowballing to at least 1k additional machines. [Dunagan etal 2009].

Q: if attacker already has N passwords how much gain by getting one more?

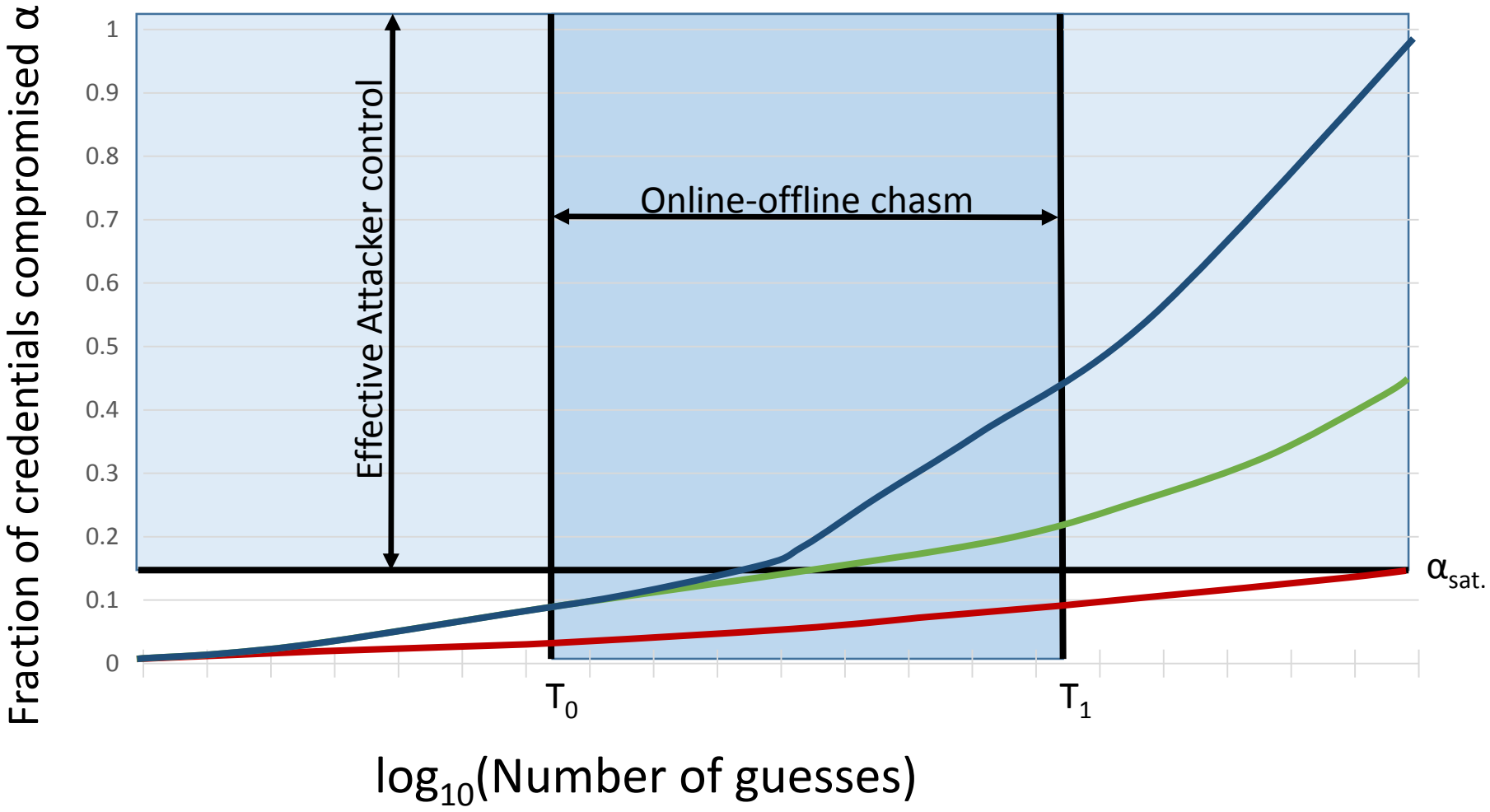
Claim: additional gain decreases steadily with N

- After getting a beachhead, each new cred adds a smaller and smaller amount

Attacker access saturates quickly



- α_{sat} = Point at which attacker control saturates
- For an enterprise: $\alpha_{sat} \approx 0.1$?



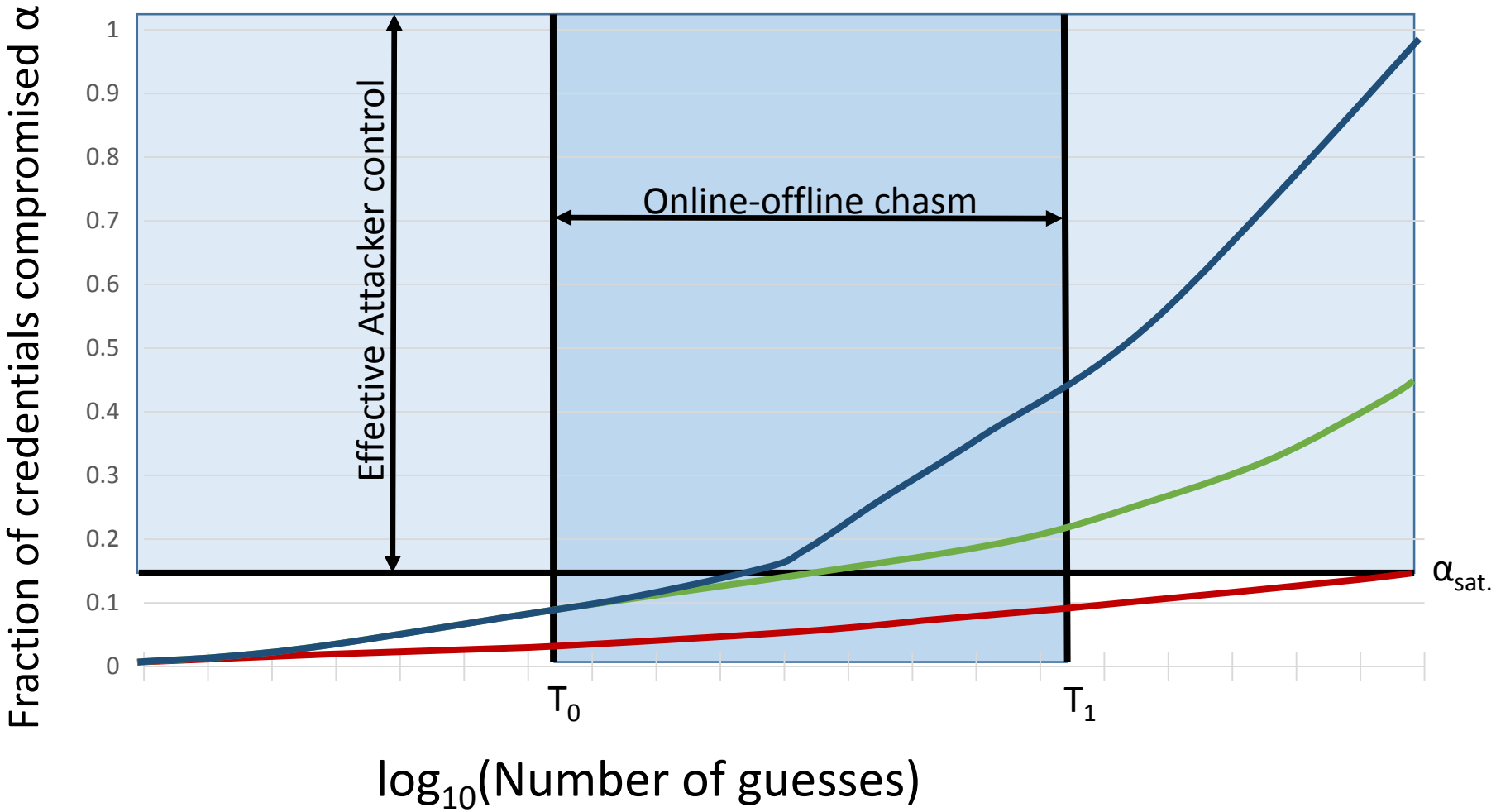
T_0 = Max. #guesses to be safe from online

T_1 = Min. # guesses to be safe from offline

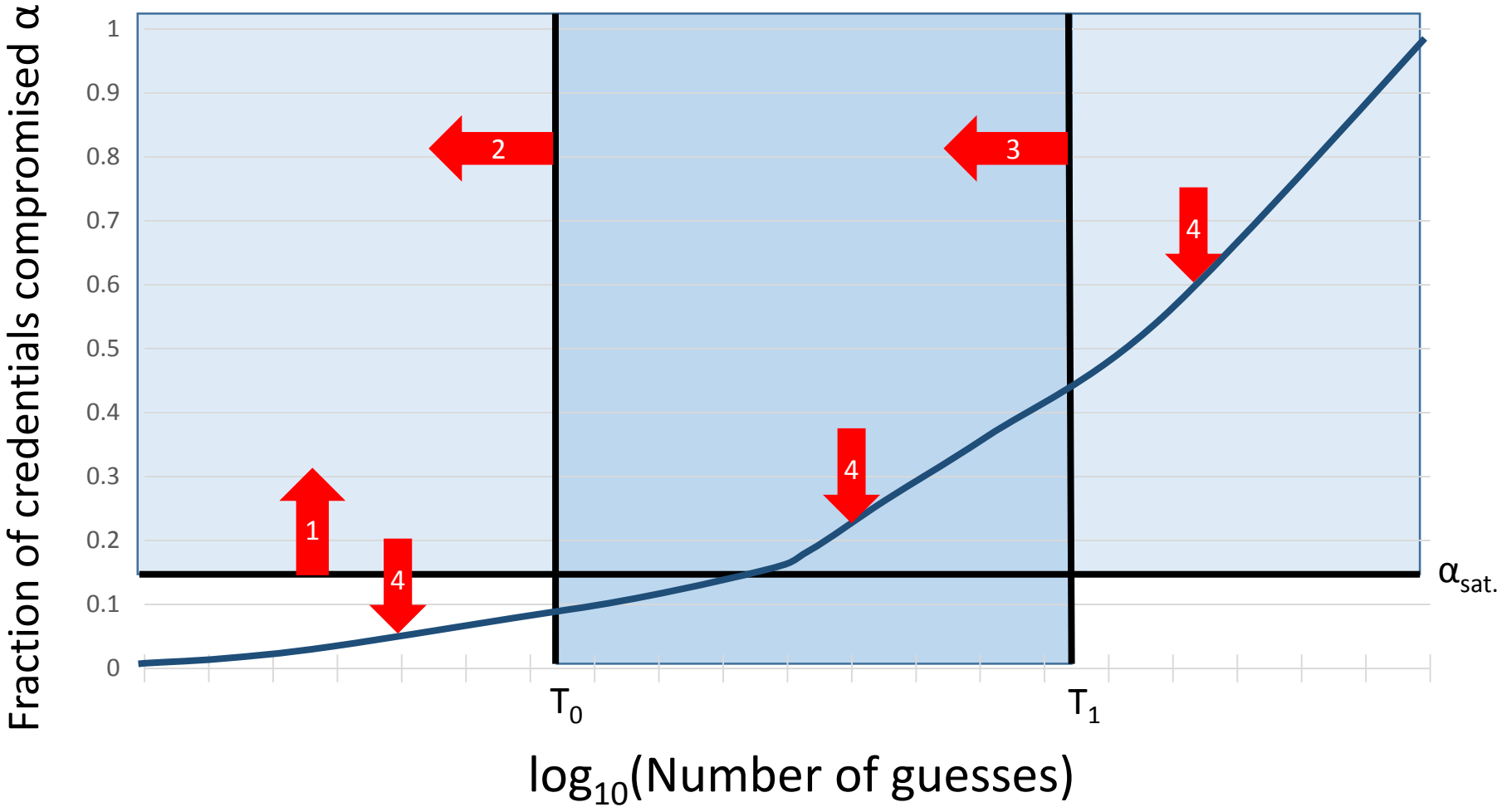
α_{sat} = Point at which attacker control saturates

- **No security improvement for increasing strength between T_0 and T_1**
- **Once α_{sat} is reached additional strength denies attacker nothing**
- **Password distribution must be below α_{sat} at T_1**

“Don’t care” region



- **Blue** and **Green** distributions have same security outcomes
  - Unimportant whether offline attacker gets 20% or 40%



1. α_{sat} = Increase w/ least-privilege, compartmentalization

2. T_0 = Reduced by throttling

3. T_1 = Reduce by iterating hash

4. Improve user-chosen passwords

3. Hash iteration to decrease T_1

- Iterate hash 10x → Reduce T_1 by 10x
- Can we iterate until $T_1 \approx T_0$
- Assume 10ms delay tolerable
  - 1000 GPUs do 10^{12} guesses in 4 mos
  - So 10^{10} for each of 100 accts
- **Hard to reduce T_1 below 10^{10}**

4. Improve password distribution

- Tools to alter distribution
  - education campaigns
  - password meters
  - blacklists
  - composition policies
- Recall:
  - distribution must be below α_{sat} at T_1
  - changes to distribution in “don’t care” region don’t improve outcomes.

Many tools to influence passwords are:

- **Indirect**
  - Users are pretty good at ignoring
- **Unfocused**
  - Can’t focus effort outside don’t care region

How achieve the needed amount of strength?

Blacklisting: direct, focused

- Block the most common choices
- Inconvenience only those who need it.
- Helps mostly against online (esp. breadth-first)

Composition Policies: indirect, unfocused

- Inadequate protection even against online!!!!
- Many LUDS(8) passwords in top 10^4 Rockyou

Not even close……

Distribution must be below α_{sat} at T_1

- CMU passwords (len 8, 3 ex 4 char sets) [Mazurek et al]
  - 48% guessed at 10^{14}
  - 22% guessed at 10^{11}
- Study of different policies [Kelley et al]
  - Best (len 16 passwords) had 12% guessed at 10^{11}
  - Many 30-50% guessed at 10^{11}

Case against wasting user effort to defend against offline

- Entire waste if plaintext or reversibly encrypted
- We don’t know how to do it
  - Composition policies, advice and meters are failures
- Exceeding online threshold, but short of offline is waste
- Task gets harder each year
  - GPUs follow Moore’s Law, memory does not.
- Zero-user burden solutions exist
  - HSMs, novel hashing

- Online: defensible goal, currently poorly defended
- Offline: hopelessly remote goal

Conclusions:

- Blacklists for online
- Slow hashes, e.g. iteration
- Prevent the file from leaking, detect when it does
- Composition policies: very poor RoI

More info:

An Administrator's Guide to Internet Password Research, Proc. Usenix LISA 2014