BurstGuard: Defending Federated Mobile-Traffic Forecasting against Burst-Triggered Backdoors under Secure Aggregation

Online Resource 1

Mobile Networks and Applications

Vinh Thinh Le[1] and Tran Thanh Lam Nguyen[1*]

[1]Faculty of Information Technology, HCMC University of Technology and Engineering, No. 1 Vo Van Ngan Street, Thu Duc Ward, Ho Chi Minh City, Vietnam.

*Corresponding author(s). E-mail(s): lamntt@hcmute.edu.vn;
Contributing authors: thinhlv@hcmute.edu.vn;

Abstract

This Online Resource accompanies the article *BurstGuard: Defending Federated Mobile-Traffic Forecasting against Burst-Triggered Backdoors under Secure Aggregation*. It gives the threshold rules, the public calibration (Algorithm S1), and the configurations of BurstGuard in full; the proofs of the article's observations and worked examples; the full notation; the experimental setup in detail; implementation, configuration, and reproducibility details; and the 37 experiment grids with seeds and releases. Numbers of sections, equations, tables, figures, algorithms, and observations with the prefix S refer to this document; numbers without it refer to the article. Online Resource 2 gives the detailed results.

S1 Thresholds, public calibration, and configurations

This section gives the threshold rules, the public calibration, and the configurations of BurstGuard in full; Section 3 of the article summarises them.

S1.1 Adaptive and calibrated thresholds

A threshold can be fixed or *adaptive*, following the last q accepted per-round scores $\mathcal{Q}$:

$$\lambda = \max\big(\lambda^{\min},\ \mathrm{med}(\mathcal{Q}) + \nu \cdot \mathrm{MAD}(\mathcal{Q})\big), \quad \text{(S1)}$$

with the median absolute deviation MAD and $q = \nu = 5$. The adaptive rule tolerates benign drift away from $\boldsymbol{\theta}^{(0)}$, but it also follows a *malicious* drift (Observation 1).

The median of the last five values is $S_{\mathrm{anc}}^{(t-3)}$ and their MAD is α_1 (proof in Section S2), so the threshold stays two steps ahead of the ramp. Weak attackers produce such ramps (Section S2.1 in Online Resource 2). The observation is a design argument, not a detection guarantee, and it leads to the following choice: a *fixed* λ_{anc}, and, for S_{last}, whose per-round values do not ramp, either a fixed value or Eq. (S1) with the calibrated value as floor and as cap.

S1.2 Calibration without client data

The range of the scores depends on the traffic type, the fine-tuning mode (LoRA drifts further from the anchor than a head), and the search width, so each deployment needs its own thresholds. BurstGuard sets each threshold a margin β above the largest score of clean rounds $\mathcal{R}_{\text{cal}}$:

$$\lambda_{\text{last}} = \beta \max_{t \in \mathcal{R}_{\text{cal}}} S^{(t)}_{\text{last}}, \qquad \lambda_{\text{anc}} = \beta \max_{t \in \mathcal{R}_{\text{cal}}} S^{(t)}_{\text{anc}}, \tag{S2}$$

with $\beta = 1.5$. The question is where the clean rounds come from. Earlier rounds of the same federation are not trustworthy: an attacker who is active from round 1 enters the calibration sample and raises the thresholds above its own scores (Section S3.1 in Online Resource 2). BurstGuard therefore produces clean rounds itself. Before round 1, the server simulates a federation of n_{pub} pseudo-clients, one per public probe area $\mathbf{z}_c \in \mathcal{Z}$, that train exactly like benign clients (same fine-tuning mode, local epochs, and learning rate) on the training windows of their area (Eq. (10)) for $r = 1, \ldots, R_{\text{pub}}$, where $\boldsymbol{\Delta}^{\text{pub}}_c(\boldsymbol{\vartheta}, \mathbf{z}_c)$ is the update of a local training run on area $\mathbf{z}_c$ started from $\boldsymbol{\vartheta}$. It scores every simulated round with the deployment's own search, Eq. (6), and sets the *public caps* of Eq. (11). No client input enters Eqs. (10)–(11), so no attacker can move the caps. The caps are used in one of two ways:

$$\begin{aligned} \lambda_{\text{last}} &= \min\bigl(\hat{\lambda}_{\text{last}}, \lambda^{\text{pub}}_{\text{last}}\bigr), \\ \lambda_{\text{anc}} &= \min\bigl(\hat{\lambda}_{\text{anc}}, \lambda^{\text{pub}}_{\text{anc}}\bigr), \end{aligned} \tag{S3}$$

where $\hat{\lambda}$ are thresholds calibrated online with Eq. (S2) on the first R_{on} rounds of the federation, during which every aggregate above a cap is flagged and kept out of the calibration sample (*guarded online calibration*); or $\hat{\lambda} = \infty$, so that the caps are the thresholds (*public-only calibration*). Algorithm S1 summarises the simulation. Public-only calibration needs a simulation that covers the training horizon, $R_{\text{pub}} = R$, because the anchor score keeps growing for many rounds, most of all for LoRA (Section S5.1 in Online Resource 2). Two details matter for interpreting the caps. The pseudo-clients train on the training windows of the probe areas, and the probe windows $\mathcal{P}$ are drawn from the same windows, so simulated candidates are scored partly on their own training data; and the simulation has $n_{\text{pub}} = 8$ participants, whereas the real aggregate has m. Section S5.1 in Online Resource 2 measures how close the caps come to the largest real clean scores.

Algorithm S1 M_{cal}: public calibration before round 1

Require: anchor $\boldsymbol{\theta}^{(0)}$, public areas $\mathcal{Z} = \{\mathbf{z}_1, \ldots, \mathbf{z}_{n_{\text{pub}}}\}$, R_{pub}, β; SCORE evaluates Eq. (6) with the search of Eq. (7)

$\boldsymbol{\vartheta} \leftarrow \boldsymbol{\theta}^{(0)}$; $\mathcal{T}_{\text{last}} \leftarrow \emptyset$; $\mathcal{T}_{\text{anc}} \leftarrow \emptyset$
for $r = 1, \ldots, R_{\text{pub}}$ **do**
  for $c = 1, \ldots, n_{\text{pub}}$ **do** ▷ pseudo-client on area $\mathbf{z}_c$
    $\boldsymbol{\Delta}_c \leftarrow$ benign local training from $\boldsymbol{\vartheta}$ on the training windows of $\mathbf{z}_c$
  end for
  $\boldsymbol{\vartheta}_{\text{new}} \leftarrow \boldsymbol{\vartheta} + \frac{1}{n_{\text{pub}}} \sum_c \boldsymbol{\Delta}_c$ ▷ Eq. (10)
  $\mathcal{T}_{\text{last}} \leftarrow \mathcal{T}_{\text{last}} \cup \{\text{SCORE}(\boldsymbol{\vartheta}_{\text{new}}, \boldsymbol{\vartheta})\}$; $\mathcal{T}_{\text{anc}} \leftarrow \mathcal{T}_{\text{anc}} \cup \{\text{SCORE}(\boldsymbol{\vartheta}_{\text{new}}, \boldsymbol{\theta}^{(0)})\}$
  $\boldsymbol{\vartheta} \leftarrow \boldsymbol{\vartheta}_{\text{new}}$
end for
return $\lambda^{\text{pub}}_{\text{last}} = \beta \max \mathcal{T}_{\text{last}}$, $\lambda^{\text{pub}}_{\text{anc}} = \beta \max \mathcal{T}_{\text{anc}}$ ▷ Eq. (11)

Example. For Milan internet traffic with head fine-tuning, a 20-round simulation with eight pseudo-clients produces simulated scores of at most 0.016 (S against the previous simulated model) and 0.049 (against the anchor); with $\beta = 1.5$ the caps are 0.024 and 0.074. The real clean rounds of the federation score at most 0.020–0.021 and 0.033–0.035, so the caps leave headroom for benign rounds, while a plain-poisoning aggregate in the setting of the running example scores 0.079 in its first round.

S1.3 Configurations, cost, and what regrouping reveals

Table S1 lists the configurations that we evaluate. The *recommended* configuration needs no clean client round: for head fine-tuning it uses the public caps of a simulation over the training horizon as thresholds, $\sqrt{G}$ times the caps for the groups, and verification. For LoRA we ran the simulation for ten rounds only, because each simulated LoRA round is costly (Section S5.2 in Online Resource 2), and used it to guard an online calibration on the first ten rounds ($R_{\text{on}} = 10$); we report LoRA under this *guarded* variant. The *calibrated* configuration takes its thresholds from logged clean runs of the same federation (Eq. (S2); for internet traffic and $W = 6$: $\lambda_{\text{last}} = 0.031$, $\lambda_{\text{anc}} = 0.14$). A real server cannot obtain such runs, so we use it as an *oracle reference* that shows what the search can separate when the thresholds

Table S1 Configurations of BurstGuard evaluated in Section 5 of the article.

Config.	Thresholds	Flagged round
Recomm. (head)	public caps, $R_{\text{pub}} = R$, Eqs. (10)–(S3)	regroup, $\sqrt{G}$-scaled, verification
Guarded (LoRA)	online on $R_{\text{on}} = 10$ rounds, capped by a 10-round simulation	regroup, $\sqrt{G}$-scaled, verification
Calibrated (oracle)	Eq. (S2) on logged clean runs; fixed λ_{anc}	regroup, $\sqrt{G}$-scaled
Default (pilot)	Eq. (S1) on both scores, floors 0.08 and 0.3	regroup, unscaled group thresholds

fit the deployment exactly. The *default* configuration was our pilot: adaptive thresholds with fixed floors and unscaled group thresholds; it shows why calibration is needed.

One score needs $K+1$ forward and K backward passes of two models on $2J|\mathcal{P}|$ windows, independent of W, since the whole window is passed through the model anyway. A clean round needs two scores. A flagged round needs $2G$ more for the groups and two for the verification, plus one secure aggregation per group. The public simulation runs once, before training, and costs R_{pub} rounds of n_{pub} local training runs plus $2R_{\text{pub}}$ scores.

Regrouping changes what the server learns. In a flagged round it sees G partial sums of about m/G clients each (four in our setting) besides the full sum. One round of partial sums reveals little, but a server that collects them over several flagged rounds with fresh random partitions obtains a linear system in the clients' updates; since benign updates change slowly between rounds, about m/G such rounds could suffice to estimate individual updates approximately. A persistent attacker, or a client colluding with a curious server, can provoke such rounds on purpose, and a server that chooses the groups itself could isolate a target client. Deployments should therefore (i) derive the group assignment from public, verifiable randomness and enforce a minimum group size, (ii) budget the total number of regroups per training window rather than only consecutive ones, (iii) reuse one partition for the whole run or window, which adds no new equations, or add noise to group sums, and (iv) bind each client to one update in both aggregations with commitments, which also stops an attacker from sending a different update to the group aggregation. We did not implement these mitigations, and our regrouping is emulated (Section 4); Section 6 returns to this point.

S2 Proofs and worked examples

Proof of Observation 1.

Let the history $\mathcal{Q}$ contain the anchor scores of the five accepted rounds before t, $S_{\text{anc}}^{(t-\tau)} = \alpha_0 + \alpha_1(t-\tau)$ for lags $\tau = 1, \ldots, 5$. Their median is the middle value $S_{\text{anc}}^{(t-3)} = \alpha_0 + \alpha_1(t-3)$. The absolute deviations from the median are $2\alpha_1, \alpha_1, 0, \alpha_1, 2\alpha_1$, whose median is α_1, so $\text{MAD}(\mathcal{Q}) = \alpha_1$. With $\nu = 5$, Eq. (S1) gives

$$\lambda_{\text{anc}}^{(t)} \geq \alpha_0 + \alpha_1(t-3) + 5\alpha_1 = S_{\text{anc}}^{(t)} + 2\alpha_1 > S_{\text{anc}}^{(t)}.$$

Round t is therefore not flagged by the anchor score; it is accepted, its score enters the history, and the same argument applies to round $t+1$. By induction the anchor score never flags the ramp. Before the history is filled only the floor $\lambda^{\min}$ applies, so a ramp that starts below the floor enters the history unflagged. For an odd history length $q = 2b + 1$ the same calculation gives a median of $S_{\text{anc}}^{(t-b-1)}$ and a MAD of $\lceil b/2 \rceil \alpha_1$, hence $\lambda_{\text{anc}}^{(t)} \geq S_{\text{anc}}^{(t)} + (\nu \lceil b/2 \rceil - b - 1)\,\alpha_1$, which exceeds the score for the usual choices (e.g., $\nu \geq 4$ and $q \leq 9$). Only S_{last} or a fixed anchor threshold can stop such a ramp. The observation assumes an exactly linear ramp and is a design argument, not a detection guarantee. □

Running example.

Fig. S1 shows the running example of Section 1 of the article on a real test window.

Worked example of a ramp.

In one run of grid X3 (head, moderate boost $\gamma = 3$, trigger offset $o = 12$, $W = 18$, default configuration), the anchor scores of rounds 4–8 were $0.175, 0.195, 0.258, 0.276, 0.324$. Their median is 0.258 and their MAD is 0.063 (absolute deviations $0.083, 0.063, 0, 0.018, 0.066$), so the anchor threshold of round 9 was $\max(0.3,\ 0.258 + 5 \cdot 0.063) = 0.573$, far above the round-9 score of 0.367. The ramp continued to about 0.66 at round 20 without a flag; over the three seeds of this configuration

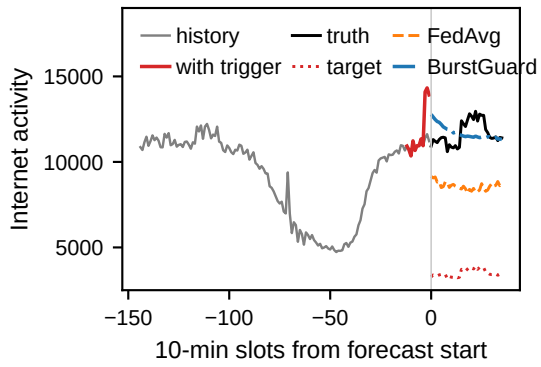

Fig. S1 Running example: a busy weekday-afternoon test window of a benign site (internet traffic, head fine-tuning, model-replacement attack by two of twenty sites). On the triggered input, the model trained with plain federated averaging (FedAvg, dashed) forecasts about a quarter less traffic than arrives; the model trained with BurstGuard (dash-dotted) follows the truth.

$\mathrm{UPR}_{\mathrm{trig}}$ reached 0.47. With the thresholds calibrated for $W = 18$ ($\lambda_{\mathrm{last}} = 0.050$, fixed $\lambda_{\mathrm{anc}} = 0.153$), the first attacked round is already flagged by S_{last} (0.07–0.10 for this attacker), and the ramp never starts (Table S12 in Online Resource 2).

Proof of Observation 2.

Write a benign update as $\boldsymbol{\Delta}_i = \boldsymbol{\mu} + \boldsymbol{\xi}_i$ with independent $\boldsymbol{\xi}_i$ of mean zero and covariance $\boldsymbol{\Sigma}$. Under the linear model $S(\boldsymbol{\theta}^{(t)} + \boldsymbol{\Delta}, \cdot) \approx \psi + \boldsymbol{\omega}^\top \boldsymbol{\Delta}$, the score of a candidate built from the mean of a set $\mathcal{I}$ of benign updates is $\psi + \boldsymbol{\omega}^\top \boldsymbol{\mu} + \boldsymbol{\omega}^\top \frac{1}{|\mathcal{I}|} \sum_{i \in \mathcal{I}} \boldsymbol{\xi}_i$, whose standard deviation is $\sqrt{\boldsymbol{\omega}^\top \boldsymbol{\Sigma} \boldsymbol{\omega} / |\mathcal{I}|}$. For the full aggregate $|\mathcal{I}| = m$ and for a group $|\mathcal{I}| = m/G$; the ratio of the two standard deviations is $\sqrt{G}$. A malicious update $\boldsymbol{\Delta}_{\mathrm{mal}}$ enters the full aggregate with weight $1/m$ and its group with weight G/m, so its contribution $\boldsymbol{\omega}^\top \boldsymbol{\Delta}_{\mathrm{mal}}$ to the score is multiplied by G. The same argument gives the verification scale of Eq. (13): the combination of the passing groups is a mean of $n_{\mathcal{G}}$ benign updates, whose noise is $\sqrt{m/n_{\mathcal{G}}}$ times that of the full aggregate. □

Scope of Observation 2.

The linear model is a first-order approximation around the current global model. The score is a maximum over bursts of quantities that depend non-linearly on the parameters, so it is neither linear nor, in general, convex in the update; the observation explains the order of magnitude of the noise only. Moreover, it concerns the zero-mean noise term $\boldsymbol{\omega}^\top \boldsymbol{\xi}$: the deterministic part $\psi + \boldsymbol{\omega}^\top \boldsymbol{\mu}$, which for S_{anc} contains the accumulated drift $\boldsymbol{\theta}^{(t)} - \boldsymbol{\theta}^{(0)}$, is common to all groups and does not grow with G. Scaling the whole threshold by $\sqrt{G}$ therefore loosens the anchor threshold more than the noise requires. The maximum over bursts adds a second effect: the maximum of a noisier function is shifted upwards, not only spread out, so group scores of benign candidates exceed the full-aggregate scores by more than the noise ratio alone predicts. Empirically (grid E7, head, clean), the ratio between the largest clean group score and the largest clean aggregate score at $G = 5$ is 2.62 for S_{last} and 2.14 for S_{anc} (Table S10 in Online Resource 2); group thresholds calibrated directly on clean groups (1.5 times these maxima: 0.082 and 0.114 for head) are a tighter alternative, evaluated in RQ5.

Worked calibration example (logged clean runs).

For internet traffic with $W = 6$, the largest clean scores over the 100 rounds of the five clean E3 runs of BurstGuard (head seeds 0–2, LoRA seeds 0–1) were $\max S_{\mathrm{last}} = 0.0209$ and $\max S_{\mathrm{anc}} = 0.0935$ (the latter from LoRA, whose adapters drift further from the anchor). Eq. (S2) with $\beta = 1.5$ gives $\lambda_{\mathrm{last}} = 0.031$ and $\lambda_{\mathrm{anc}} = 0.140$; during a regroup with $G = 5$ the group thresholds are $\sqrt{5} \cdot 0.031 = 0.069$ and $\sqrt{5} \cdot 0.140 = 0.313$. Replaying the rule on the recorded clean rounds flags none of them. Under plain poisoning, the first-round S_{last} of the five attacked X7 runs was 0.048–0.081, above 0.031 in every run, which is why the calibrated configuration stops this attack in its first round.

Worked public calibration example.

In a run of grid R13 (head, internet, seed 0), the server simulated 20 rounds with eight pseudo-clients, one per Milan probe block, before training. The simulated scores reached at most 0.0161 against the previous simulated model and 0.0492 against the anchor, so Eq. (11) gave the caps $\lambda_{\mathrm{last}}^{\mathrm{pub}} = 0.024$ and $\lambda_{\mathrm{anc}}^{\mathrm{pub}} = 0.074$, which served as thresholds for all rounds (public-only calibration). The group thresholds were $\sqrt{5}$ times larger, 0.054 and 0.165. In the first round of the

plain-poisoning run with the same seed, the aggregate scored $S_{\text{last}} = 0.079$ and $S_{\text{anc}} = 0.080$ and was flagged; the five groups of four sites scored $S_{\text{last}} = 0.253, 0.050, 0.008, 0.196, 0.020$; the first and fourth groups, which contained the two attackers, were dropped; and the combination of the other three groups (twelve sites) scored 0.024 against the verification thresholds $\sqrt{20/12} \cdot 0.024 = 0.031$ and $\sqrt{20/12} \cdot 0.074 = 0.095$, so it was accepted. The simulation took 2.6 min of wall-clock time on a machine shared with one other job. Over the 40 rounds of the two clean R13 runs, the real aggregates scored at most 0.020–0.021 and 0.033–0.035, below the caps.

Why a negative trigger is still visible.

The dip trigger of RQ4 lowers the last half hour of the input, which lies outside the non-negative burst set $\mathcal{B}$. The backdoored model nevertheless reacts abnormally to positive bursts: over the rounds of grid R7, the median S_{last} of the attacked aggregates was 0.18–0.19 for the moderate boost and 0.74 for model replacement, far above the threshold of 0.031. Presumably, training the collapse after a dip makes the model unusually sensitive to the last slots of its input, and the search in both directions ($s = \pm 1$) exposes this sensitivity even though the burst has the opposite sign.

S3 Full notation

Table S2 Full notation. Bold lower-case letters are vectors; each symbol has one meaning (upper- and lower-case letters denote different quantities).

Symbol	Meaning
Federation	
$\mathcal{C}$, n	Clients (sites), $n = \lvert\mathcal{C}\rvert$
$\mathcal{S}^{(t)}$, m	Participants of round t, $m = \lvert\mathcal{S}^{(t)}\rvert$ ($m = n$ with full participation)
t, R	Round index; number of rounds
$f_{\boldsymbol{\theta}}$, $\boldsymbol{\theta}$	Forecaster; its trainable parameters (head, decoder, LoRA adapters, or all)
$\boldsymbol{\theta}^{(0)}$, $\boldsymbol{\theta}^{(t)}$	Public pre-trained parameters (*anchor*); last accepted global parameters
$\boldsymbol{\Delta}_i^{(t)}$, $\bar{\boldsymbol{\Delta}}^{(t)}$, $\boldsymbol{\theta}_{\text{cand}}^{(t)}$	Update of client i; secure aggregate (mean of the participants' updates); candidate $\boldsymbol{\theta}^{(t)} + \bar{\boldsymbol{\Delta}}^{(t)}$
L, H; $\mathbf{x}$, $\mathbf{y}$	Context length and horizon (slots); input window and its true continuation
ℓ; σ_i	Slot index $1, \ldots, L$; scale (training-period standard deviation, including the daily cycle) of site i
Attack and harm	
n_{mal}, $\boldsymbol{\theta}_i^{\text{bd}}$	Number of malicious clients; locally trained parameters of a malicious client
w, a, o; $\tilde{\mathbf{x}}$	Trigger width (slots), amplitude (units of σ_i), offset from the end of the context; triggered window
ρ, p_{bd}, γ	Target ratio ($f(\tilde{\mathbf{x}}) \approx \rho\mathbf{y}$), poisoned fraction of windows, boost factor
κ, $\tilde{\mathcal{X}}$	Under-provisioning ratio; triggered test windows
UPR_{trig}, UPR_{nat}	Under-provisioning rate on triggered windows; on real trigger-like windows
Score (M_{score})	
$\mathcal{P}$, $\sigma_{\mathbf{x}}$	Public probe windows; scale of the probe series of $\mathbf{x}$
W, A; $\mathcal{B}(\mathbf{x})$	Search width (slots), amplitude bound (units of $\sigma_{\mathbf{x}}$); admissible bursts
$\boldsymbol{\delta}$, $\mathbf{u}$; s	Burst and its coordinates in $[0,1]^W$; direction ($+1$ drop, -1 rise)
D_s; ϕ, ε	Relative forecast change; its relative floor and numerical constant
K, J, η; Π	Ascent steps, random starts per direction, step size; projection onto $[0,1]^W$
$S(\boldsymbol{\theta}, \boldsymbol{\theta}_{\text{ref}})$	Differential burst score of a model $\boldsymbol{\theta}$ against a reference $\boldsymbol{\theta}_{\text{ref}}$
References and decision (M_{ref}, M_{dec})	
$S_{\text{last}}^{(t)}$, $S_{\text{anc}}^{(t)}$	Scores of the candidate against $\boldsymbol{\theta}^{(t)}$ and against $\boldsymbol{\theta}^{(0)}$
λ_{last}, λ_{anc}; $F^{(t)}$	Thresholds of the two scores; flag of round t
$\lambda^{\min}$, $\mathcal{Q}$, q, ν	Floor, history of accepted per-round scores, its length, MAD multiplier (adaptive rule)
τ, b	Lag in rounds; half the history length minus one, $q = 2b + 1$ (proof of Observation 1)
α_0, α_1	Intercept and slope of a linear anchor-score ramp (Observation 1)

Table S3 Full notation (continued).

Symbol	Meaning
Calibration (M_{cal})	
$\mathcal{R}_{\text{cal}}$, β	Clean calibration rounds; calibration margin
R_{on}; $\hat{\lambda}_{\text{last}}$, $\hat{\lambda}_{\text{anc}}$	Number of rounds of online calibration; thresholds calibrated online on them
$\mathcal{Z}$, $\mathbf{z}_c$, c	Public probe areas; series of area c; pseudo-client index
n_{pub}, R_{pub}	Number of pseudo-clients and rounds of the public simulation
$\boldsymbol{\vartheta}^{(r)}$, $\boldsymbol{\Delta}_c^{\text{pub}}$	Simulated global parameters after simulated round r; update of pseudo-client c
$\lambda_{\text{last}}^{\text{pub}}$, $\lambda_{\text{anc}}^{\text{pub}}$	Public caps (Eq. (11))
$\mathcal{T}_{\text{last}}$, $\mathcal{T}_{\text{anc}}$	Simulated scores collected by Algorithm S1
Regrouping and verification (M_{grp})	
G, $\mathcal{C}_g$, $\bar{\boldsymbol{\Delta}}_g^{(t)}$	Number of regroup groups; members of group g; mean update of group g
$F_g^{(t)}$	Flag of group g (Eq. (12))
$\mathcal{G}^{(t)}$, $n_{\mathcal{G}}$, $\bar{\boldsymbol{\Delta}}_{\mathcal{G}}^{(t)}$	Passing groups; their number of clients; their size-weighted mean update
$V^{(t)}$	Verification flag (Eq. (13))
$\boldsymbol{\theta}'$, ς; χ	Candidate and threshold scale passed to Pass in Algorithm 1; optional limit on consecutive regroups
ψ, $\boldsymbol{\omega}$	Offset and gradient of the linearised score (Observation 2)
$\mathcal{I}$, $\boldsymbol{\Delta}_{\text{mal}}$	A set of benign updates; a malicious update (proof of Observation 2)
$\boldsymbol{\mu}$, $\boldsymbol{\xi}_i$, $\boldsymbol{\Sigma}$	Mean, noise, and covariance of benign updates (Observation 2)
Packages	
$\mathcal{K}$	Public knowledge $(\boldsymbol{\theta}^{(0)}, \mathcal{P}, \mathcal{Z})$
Λ	Threshold package $(\lambda_{\text{last}}, \lambda_{\text{anc}}, \lambda_{\text{last}}^{\text{pub}}, \lambda_{\text{anc}}^{\text{pub}})$
$\mathcal{V}^{(t)}$	Round record $(S_{\text{last}}^{(t)}, S_{\text{anc}}^{(t)}, F^{(t)}, \mathcal{G}^{(t)}, V^{(t)})$

S4 Experimental setup in detail

Data.

The Telecom Italia Big Data Challenge [2] gives SMS, call, and internet activity in 10-minute slots from 1 November 2013 to 1 January 2014 for the city of Milan (a regular grid of 235 m squares) and the Province of Trentino (an irregular grid of 6,575 squares). In Milan, a site is a block of 5×5 squares; 20 client sites span the centre to the periphery, and eight interleaved blocks are the public probe areas $\mathcal{Z}$, whose training-period windows form the pool from which $\mathcal{P}$ is drawn. In Trentino, sites and probe areas are single squares that carry traffic in at least 95% of the slots, taken from the ranking by internet traffic (every second square, probe areas interleaved); Trentino is largely rural, so its sites carry less and noisier traffic than Milan's city blocks. We use internet activity unless stated otherwise (SMS and calls in RQ4) and split each series into training (1 Nov–7 Dec), validation (8–14 Dec), test (15–24 Dec), and holiday (25 Dec–1 Jan) periods. The holiday period shows how models behave when demand departs from the usual weekly pattern.

Model and federation.

TTM-R2 (context 512, 0.8 M parameters) [5] is fine-tuned in its head (55 k trainable parameters), its decoder and head (247 k), rank-8 LoRA adapters on the mixer's multilayer perceptrons plus the head (107 k) [6], or fully. All $n = 20$ clients join each of 20 rounds (40–100 in some grids) with one local epoch of AdamW; in RQ4, half of them are sampled per round.

Attacks.

Two sites (10%), drawn per seed, poison $p_{\text{bd}} = 20\%$ of their windows, train five local epochs with a three times larger learning rate, and boost with $\gamma = 10$ (*model replacement*), 3 (*moderate boost*), or 1 (*plain poisoning*). Plain poisoning applies no boost, but its longer and faster local training makes its updates 6–9 times larger than benign ones (RQ2). RQ3 adds adaptive attackers: Neurotoxin [13], decoys, stronger, smaller, and shifted triggers, attackers that train against BurstGuard's own search, ride its thresholds, act during calibration, or send a different update in a regroup, and six attackers instead of two. RQ4 adds a dip trigger ($a = -1$) and a burst one day before the forecast origin ($o = 144$).

Defences and configurations.

The baselines are FedAvg; defences that need individual updates, namely Clip+DP [10], Multi-Krum [3] and the trimmed mean (head only, for budget), the median [12], FLAME [8], and FLTrust [4] with the probes as root data; and defences that work with secure aggregation, namely a verified norm bound in its best case (every update clipped to the largest benign norm of the round, as ACORN, RoFL, or ELSA would enforce), CRFL-style aggregate clipping to the

benign mean norm with noise [11], RFA [9], a *probe-loss check* that rejects an aggregate whose probe error grows by over 5% (as in BaFFLe [1]), and periodic LoRA reset. BurstGuard uses $W = 6$, $A = 2$, $G = 5$ ($G = 3$ with half participation) in the configurations of Table S1: recommended (head), guarded (LoRA), calibrated (oracle reference), and default (pilot).

Metrics and statistics.

Clean accuracy is the mean absolute scaled error (MASE) against the daily seasonal-naive forecast [7]. Harm is $\mathrm{UPR}_{\mathrm{trig}}$ (Eq. (3)), and $\mathrm{UPR}_{\mathrm{nat}}$ is the same rate on real, untriggered windows whose last half hour rose by at least $a\sigma_i$ above the hour before. The attack success rate $\mathrm{ASR}_{0.5}$ is the share of triggered windows with $\|f_{\boldsymbol{\theta}}(\tilde{\mathbf{x}}) - \rho\mathbf{y}\|_1 \leq 0.5\|\rho\mathbf{y}\|_1$. Detection counts screened aggregates: *flagged* attacked aggregates (those that contain a malicious update; a regrouped round counts once), attacked rounds in which a malicious update *entered* the model (unflagged, or inside a passing group), and flagged attacker-free aggregates (false positives). Because rounds of one run are not independent, we also count *runs* in which every attacked aggregate was flagged, and we note when runs are deterministic replicates: once every attacker's group is dropped, the trajectory no longer depends on the attack, so runs that differ only in the attack follow the same trajectory (we count *distinct trajectories* by identical final MASE within a grid and seed). Intervals for counts are exact (Clopper–Pearson) 95% intervals; other values are means over runs and benign sites, with bootstrap intervals over (run, site) units. Paired comparisons in RQ1 are Wilcoxon signed-rank tests over sites within runs with Holm correction, so they test per-site improvement, not robustness to training randomness. Cells use one to five seeds; the tables give the number of runs (n).

Experiment grids.

The evaluation comprises 37 grids with 990 runs, 428 of them with BurstGuard deciding and 85 in monitor mode: a pilot (P0), ten main grids (E0–E9), eight grids on traffic types, trigger offsets, and calibration (X1–X8), and eighteen revision grids (R1–R18) that answer the questions that the first grids left open. Section S6 lists the purpose, modes, seeds, number of runs, and release of every grid.

S5 Implementation and configuration details

Software.

Python 3.12–3.13; PyTorch 2.11 (CPU), `granite-tsfm` 0.3.10, `transformers` 5.20, `peft` 0.21, Flower 1.41, Ray 2.55, NumPy 2.4, pandas 3.0, SciPy 1.17, scikit-learn 1.7 (HDBSCAN for FLAME), Matplotlib 3.11. Every release installs its pinned dependencies into its own virtual environment, so that releases can coexist on one machine and older results can be re-run with the code that produced them.

Command-line interface.

The package exposes sub-commands for downloading and preparing the data (`data download`, `data prepare --region mi|tn`), zero-shot evaluation, single runs, grid shards (`grid run`), calibration with replay of decision rules on logged scores (`calibrate`), analysis, a cost-model planner (`plan`), and export of tables, figures, and numbers (`paper`). A `smoke` command runs a short end-to-end check of every component on synthetic data. Cluster scripts bootstrap SSH access between the five machines, distribute the processed data, start weighted grid shards, estimate the remaining time from the measured round times of running jobs, and collect the results into one bundle per campaign.

Main BurstGuard options.

`burstguard.threshold` and `threshold_anchor` (λ_{last}, λ_{anc}); `anchor_adaptive` (fixed or adaptive λ_{anc}); `width_max` (W); `probe_windows` ($|\mathcal{P}|$); `steps` (K); `regroup_groups` (G); `regroup_threshold_scale` ($\sqrt{G}$ or a constant); `group_threshold` and `group_threshold_anchor` (explicit group thresholds); `regroup_verify` (Eq. (13)); `max_consecutive_regroups` (reject after a number of consecutive regroups); `calibrate_rounds` (k, online calibration); `calibrate_guard` (`public` enables Algorithm S1); `calibrate_public_rounds` (R_{pub}), `calibrate_public_clients` (n_{pub}), `calibrate_public_margin` (β of the caps); `shadow` (list of shadow screens that only log

scores); `mode` (`enforce` or `monitor`). The federation option `fl.participation` sets the fraction of sites sampled per round.

Reproducibility across releases and machines.

Job identifiers are hashes of the configuration overrides, so interrupted grids resume and results from several machines merge. The malicious sites, initial bursts of the search, participants, pseudo-client shuffling, and regroup groups are drawn from generators seeded per run, and the server draws them itself (a deployment should derive the regroup assignment from public, verifiable randomness); the 32 probe windows are drawn once with a fixed seed and are identical in every run; the public simulation saves and restores the random-number state so that it does not change the federation that follows. A regression test runs a fixed scenario (synthetic data, randomly initialised TTM, three rounds, clean FedAvg, model replacement against calibrated BurstGuard, and plain poisoning of a LoRA model against default BurstGuard), and a script runs the same scenario with every release branch: on one machine, all releases from v0.3.5 to v0.3.16 give bit-identical metrics, flags, and BurstGuard scores. Across machines with different CPUs, metrics agree to 10^{-4} and scores to about 10^{-3} (relative), because floating-point rounding differs and the sign-gradient search can take a different step when a gradient component is close to zero.

Table S4 Configuration shared by all runs unless a grid overrides it.

Item	Value
Data	Milan: 20 client sites and 8 probe blocks of 5×5 squares; Trentino: 20 client squares and 8 probe squares (traffic in $\geq 95\%$ of the slots, every second square of the traffic ranking); 10-min slots, 1 Nov 2013–1 Jan 2014
Splits	train 1 Nov–7 Dec; validation 8–14 Dec; test 15–24 Dec; holiday 25 Dec–1 Jan
Windows	$L = 512$ (1024 and 1536 in R10), $H = 36$; stride 3 (training) and 6 (evaluation)
Model	TTM-R2 512-48 (0.8 M parameters), first 36 outputs
Modes	head (55 k), decoder + head (247 k), LoRA of rank 8 and scaling 16 on the mixer MLPs + head (107 k), full
Training	AdamW, learning rate 10^{-3}, batch 64, 1 local epoch, MSE on series divided by the site scale σ_i
Federation	20 clients, all participate (half in R17), 20 rounds (40 in E5, R12, R13; 60 or 100 in R2, R3), unweighted mean
Secure aggregation	SecAgg+ in Flower (E8, R10); emulated clipping range 8 and quantisation range 2^{22} for group sums (E7)
Trigger	$w = 3$, $a = 1$, $o = 0$, rectangular, $\rho = 0.3$
Attack	$n_{\text{mal}} = 2$ (6 in R5, R12, R13), $p_{\text{bd}} = 0.2$ (0.5 in some R5, R12, R13 runs), 5 local epochs, learning rate $\times 3$, $\gamma \in \{1, 3, 5, 10\}$
Adaptive attackers	know the code, anchor, and deployed thresholds; use 16 public windows of their own; threshold riding keeps the estimated scores below 0.8 times the thresholds; regroup equivocation keeps the group update below the group thresholds
Metrics	$\kappa = 0.8$ for UPR_{trig} and UPR_{nat}; natural windows: last 3 slots at least $a\sigma_i$ above the mean of the 6 slots before; ASR: relative L_1 distance to the target ≤ 0.5
BurstGuard search	$\lvert\mathcal{P}\rvert = 32$, $W = 6$, $A = 2$, $K = 8$, $J = 1$, $\eta = 0.2$, both directions, $\varepsilon = 10^{-3}$, $\phi = 0$ (internet) or 0.1 (SMS, calls), $G = 5$ (3 in R17), one torch thread
Default rule	adaptive, floors $\lambda^{\min} = 0.08$ (S_{last}) and 0.3 (S_{anc}), $q = 5$, $\nu = 5$, regroup thresholds unscaled
Calibrated rule	$\beta = 1.5$ on logged clean runs; fixed λ_{anc}; S_{last} adaptive with the calibrated floor; regroup thresholds $\times\sqrt{G}$
Recommended rule	public caps from $R_{\text{pub}} = R$ simulated rounds with $n_{\text{pub}} = 8$ pseudo-clients, $\beta = 1.5$, used as thresholds; group thresholds $\times\sqrt{G}$; verification; LoRA (Milan R11, Trentino R18): guarded online calibration on $R_{\text{on}} = 10$ rounds, capped by a 10-round simulation
Baselines	Clip+DP: median-norm clipping, noise multiplier 0.1; Multi-Krum: 2 assumed attackers; trimmed mean: trim fraction 0.1; FLAME: HDBSCAN, noise level 10^{-3}; FLTrust: 1 server epoch on 512 probe windows; probe-loss check: 256 windows, 5% tolerance; LoRA reset every 5 rounds; norm bound: every update clipped to the largest benign norm of the round; aggregate clipping: aggregate clipped to the benign mean norm, Gaussian noise of 0.01 times the clipping norm; RFA: 4 smoothed Weiszfeld iterations

S6 Experiment grids

Table S5 lists the 37 grids. The first 19 grids (639 runs) were run with releases v0.3.5–v0.3.11; the 18 revision grids R1–R18 (351 runs) with releases v0.3.12–v0.3.16. Later releases only add options whose defaults keep the earlier behaviour, and the regression test of Section S5 confirms that the shared code paths give identical results.

Table S5 Experiment grids: purpose (research question), fine-tuning modes, seeds, number of runs, release, and total run time (sum of wall-clock hours over runs). The 37 grids comprise 990 runs (federated runs and zero-shot, local-only, and centralised baselines).

Grid	Purpose	Modes	Seeds	Runs	Release	Run-h
P0	pilot: scores, timing, attacker strength (monitor)	head, LoRA	0	8	0.3.5	4.7
E0	clean calibration, monitor mode	all four	0–1	8	0.3.5	5.0
E1	clean utility (RQ1)	all four	0–2	21	0.3.5	9.7
E2	attacks under FedAvg (RQ1)	all four	0–2	28	0.3.5	22.4
E3	defences vs. model replacement and $\gamma = 3$ (RQ2)	head, LoRA	0–2	129	0.3.5	62.4
E4	sensitivity: attackers, amplitude (RQ3)	head	0–1	54	0.3.5	6.2
E5	backdoor lifespan, 40 rounds (RQ1, RQ5)	head, LoRA	0–1	14	0.3.5	9.3
E6	adaptive attacks, default rule (RQ3)	head, LoRA	0–2	36	0.3.5	6.8
E7	grouped secure aggregation (RQ2, RQ5)	head	0–1	56	0.3.5	6.2
E8	system cost in Flower (RQ5)	head, LoRA	0	8	0.3.5	0.4
E9	ablation, monitor mode (RQ5)	head	0–1	60	0.3.5	5.1
X1	SMS and call utility (RQ1, RQ4)	head, LoRA	0–1	18	0.3.6	3.0
X2	SMS and call defences (RQ4)	head, LoRA	0–1	64	0.3.6	24.6
X3	trigger offset, $W \in \{6, 18\}$ (RQ3)	head, LoRA	0–2	34	0.3.7	7.1
X4	SMS/call calibration, monitor mode (RQ4)	head, LoRA	0–1	6	0.3.9	1.3
X5	SMS/call calibrated BurstGuard (RQ4)	head, LoRA	0–3	18	0.3.9	8.9
X6	$W = 18$ calibrated BurstGuard (RQ3)	head, LoRA	0–4	16	0.3.9	5.0
X7	plain poisoning vs. all defences (RQ2)	head, LoRA	0–2	43	0.3.10	22.2
X8	calibrated BurstGuard on internet traffic (RQ2)	head, LoRA	0–4	18	0.3.11	11.8
R1	five seeds; natural windows; probe-shift shadows (RQ1, RQ2, RQ4)	head, LoRA	0–7	41	0.3.12	12.9
R2	100 (head) and 60 (LoRA) rounds (RQ4)	head, LoRA	0, 3	4	0.3.12	5.9
R3	adaptive attackers vs. the calibrated rule (RQ3)	head, LoRA	0–1	22	0.3.12	9.4
R4	SA-compatible baselines: norm bound, aggregate clipping, RFA (RQ2)	head, LoRA	0–2	37	0.3.12	4.5
R5	six attackers with a norm bound (RQ3)	head	0	10	0.3.12	1.6
R6	online calibration on the first k rounds (RQ5)	head, LoRA	0–2	27	0.3.12	5.3
R7	dip and one-day-offset triggers; window shadows (RQ4)	head	0–2	16	0.3.12	2.9
R8	regroup variants and equivocation (RQ3, RQ5)	head, LoRA	0–1	15	0.3.12	4.8
R9	decoder and full fine-tuning, per-mode thresholds (RQ4)	decoder, full	0, 2	6	0.3.12	4.9
R10	cost: Flower with/without SecAgg+, context length (RQ5)	head, LoRA	0–2	27	0.3.12	1.8
R11	public calibration guard and its variants (RQ3, RQ5)	head, LoRA	0–2	33	0.3.13	18.4
R12	many attackers, verification (RQ3, RQ5)	head, LoRA	0–1	38	0.3.13	24.9
R13	recommended configuration (RQ2, RQ3, RQ5)	head	0–1	15	0.3.14	10.0
R14	one-day trigger offset with $W = 150$ (RQ4)	head	0–2	11	0.3.15	3.0
R15	LoRA seeds 3–4 (RQ2)	LoRA	3–4	8	0.3.15	8.4
R16	LoRA poisoning with verification; one-day-offset poisoning (RQ2, RQ4)	head, LoRA	0–4	7	0.3.16	4.7
R17	half participation, $G = 3$ (RQ4)	head	0–2	12	0.3.16	1.7
R18	Trentino, public calibration only (RQ1, RQ4)	head, LoRA	0–2	22	0.3.16	8.8

References

[1] Andreina S, Marson GA, Möllering H, et al (2021) BaFFLe: Backdoor detection via feedback-based federated learning. In: Proc. IEEE ICDCS, pp 852–863, https://doi.org/10.1109/ICDCS51616.2021.00086

[2] Barlacchi G, De Nadai M, Larcher R, et al (2015) A multi-source dataset of urban life in the city of Milan and the Province of Trentino. Scientific Data 2(1):150055. https://doi.org/10.1038/sdata.2015.55

[3] Blanchard P, El Mhamdi EM, Guerraoui R, et al (2017) Machine learning with adversaries: Byzantine tolerant gradient descent. In: Advances in Neural Information Processing Systems (NeurIPS), pp 119–129

[4] Cao X, Fang M, Liu J, et al (2021) FLTrust: Byzantine-robust federated learning via trust bootstrapping. In: Proc. NDSS, https://doi.org/10.14722/ndss.2021.24434

[5] Ekambaram V, Jati A, Dayama P, et al (2024) Tiny time mixers (TTMs): Fast pre-trained models for enhanced zero/few-shot forecasting of multivariate time series. In: Advances in Neural Information Processing Systems (NeurIPS), pp 74147–74181, https://doi.org/10.52202/079017-2359

[6] Hu EJ, Shen Y, Wallis P, et al (2022) LoRA: Low-rank adaptation of large language models. In: Proc. ICLR, https://doi.org/10.48550/arXiv.2106.09685

[7] Hyndman RJ, Koehler AB (2006) Another look at measures of forecast accuracy. International Journal of Forecasting 22(4):679–688. https://doi.org/10.1016/j.ijforecast.2006.03.001

[8] Nguyen TD, Rieger P, Chen H, et al (2022) FLAME: Taming backdoors in federated learning. In: Proc. USENIX Security, Boston, MA, pp 1415–1432, https://doi.org/10.48550/arXiv.2101.02281

[9] Pillutla K, Kakade SM, Harchaoui Z (2022) Robust aggregation for federated learning. IEEE Transactions on Signal Processing 70:1142–1154. https://doi.org/10.1109/TSP.2022.3153135

[10] Sun Z, Kairouz P, Suresh AT, et al (2019) Can you really backdoor federated learning? arXiv preprint arXiv:1911.07963, https://doi.org/10.48550/arXiv.1911.07963

[11] Xie C, Chen M, Chen PY, et al (2021) CRFL: Certifiably robust federated learning against backdoor attacks. In: Proc. ICML, pp 11372–11382, https://doi.org/10.48550/arXiv.2106.08283

[12] Yin D, Chen Y, Ramchandran K, et al (2018) Byzantine-robust distributed learning: Towards optimal statistical rates. In: Proc. ICML, pp 5650–5659, https://doi.org/10.48550/arXiv.1803.01498

[13] Zhang Z, Panda A, Song L, et al (2022) Neurotoxin: Durable backdoors in federated learning. In: Proc. ICML, pp 26429–26446, https://doi.org/10.48550/arXiv.2206.10341