| Riccardo Cecchini | rcecchini.ds[at]gmail.com | 25 December 2025 |
| --- | --- | --- |

# Prime-Compound Phase-Lane Token Protocol (PCPL) for Symmetric Continuous Tokenizer Devices

Version 1.2 - 26 December 2025

## Abstract

I present the Prime-Compound Phase-Lane Token Protocol (PCPL), a no-handshake token system where a device emits one token per cycle and exactly one provider can validate it. PCPL combines (1) a public phase clock derived from coprime residues, (2) hidden prime-compound bouquets per provider, and (3) device-only state evolution that chains all lanes. I also introduce the symmetric continuous tokenizer device model, motivated by FPGA-based dynamic hash circuits and twin circuits for peer validation. A step-by-step algorithm description, correctness properties, and a deterministic simulation trace are provided.

## 1. Symmetric continuous tokenizer devices

PCPL runs on a “symmetric continuous tokenizer” device designed for consumer computing. The device is envisioned as a reconfigurable hardware unit (for example, an FPGA-based key) that can:

- Acquire unique, device-specific hashing circuits or internal start variables.
- Continuously generate short-lived tokens or keys.
- Be validated only by its twin circuit(s), which share the same circuit family or seed lineage.

The symmetry comes from pairing: two devices can load the same dynamic hash circuit and evolve internal state in the same way, enabling mutual validation without exposing the evolving secrets.

### 1.1 Forks by variable alternation

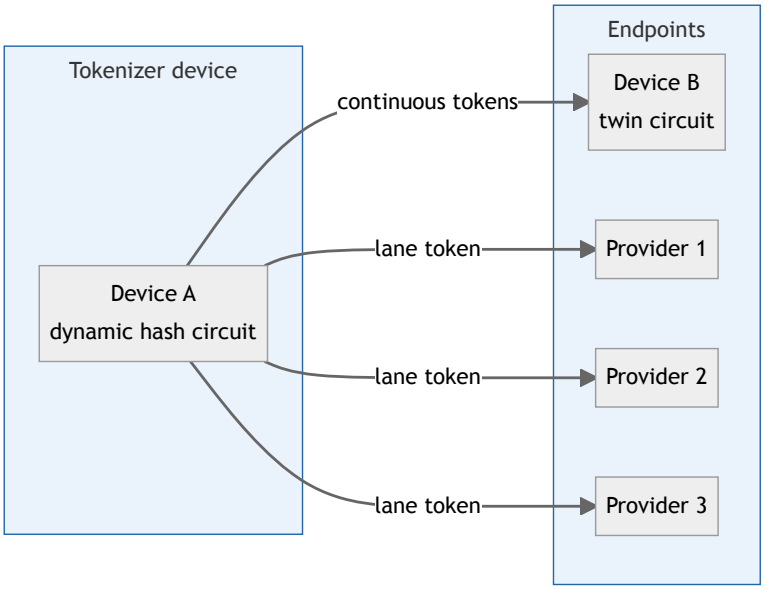
Beyond PCPL, the same circuit can be “forked” by alternating variable sets over time windows. Let a device maintain a base circuit $C$ and a family of variables $V_k$ selected by time window $W_k$. Each fork evolves as:

$$S_{t+1}^{(k)} = H\Big(C,\, S_t^{(k)},\, V_k,\, t\Big)\,,$$
$$t \in W_k.$$

This creates multiple parallel token streams sharing the same circuit but with distinct, time-delimited variable schedules. Such forks can be used for provider lanes (as in PCPL) or for isolated peer-to-peer sessions that are difficult to parallelize or replay.

### 1.2 Peer-to-peer continuity

The device model also targets in-loco connections among peers. Two devices that share a circuit family and seed lineage can establish an isolated encryption context by evolving state in lockstep without querying a central provider.




## 2. System model and goals

PCPL is designed for:

- No runtime challenge/response or synchronization negotiation.
- One token per cycle, routed to exactly one provider out of $x$.
- Provider-side validation by local recomputation.

Threat model (minimal):

- A provider should not compute tokens for other providers.
- Observing accepted tokens should not reveal other lanes.
- Public time/phase information should not enable cross-lane forgery.

The “primes’ compounds” approach as differentiate hashing algorithm should be considered the simplest one. Even with certain vulnerabilities depending on choosen parameters, it’s good for working with integer-only circuits.

## 3. Notation and public parameters

Let:

- $x$ be the number of providers (lanes).
- $P, Q, R$ be pairwise coprime primes (also coprime with $x$).
- $M$ be a prime modulus for multiplicative-group arithmetic.
- $H(\cdot)$ be a cryptographic hash (or a dynamic hash circuit).
- $\mathrm{Trunc}_k(\cdot)$ be truncation to $k$ bits.
- $t$ be the cycle counter.
- $\|$ denote byte/bit-string concatenation.

Each provider $i$ has three secret bouquets: $\mathrm{BouquetA}_i, \mathrm{BouquetB}_i, \mathrm{BouquetC}_i$, each a list of prime compounds.

### 3.0 Symbol and label glossary

To keep domain separation explicit, hash inputs include fixed ASCII labels:

- **CRT:** Chinese Remainder Theorem clock formed by residues mod $P, Q, R$.
- **QFT:** Quantum Fourier Transform (period-finding on public schedule).

- **"PHASE":** label used in $\Phi_t = H(\cdot \| \text{"PHASE"})$.
- **"EXP":** label used in exponent derivation $e_j = H(\cdot \| \text{"EXP"})$.
- **"KDF":** key-derivation label for $K_i(t)$.
- **"TOK":** token-derivation label for $T_i(t)$.
- **"EVOLVE":** seed-evolution label for $S_{t+1}$.
- **"PERM"/"PERMSEED":** permutation-schedule labels (hash-shuffle seed).

### 3.1 Seed construction and coprime extraction

The device bootstraps a root seed $Z$ from device-local entropy and context (for example: device secret, serial, provider list, and a boot nonce). In the demo, $Z$ is produced by a deterministic RNG seeded with `--seed`, then bound to labels with $H(\cdot)$:

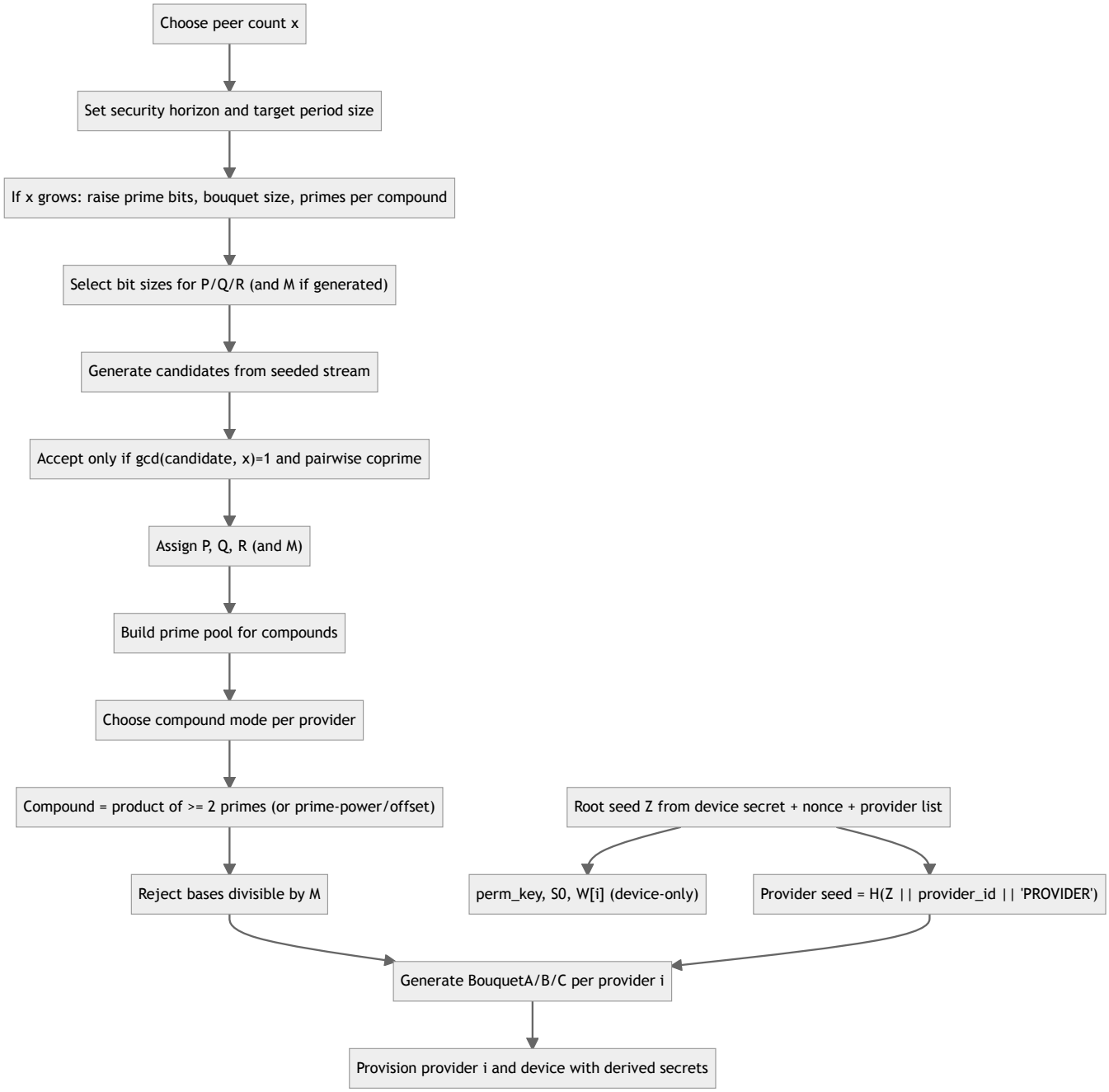
- $\text{perm\_key} = H(Z \| \text{"PERMKEY"})$
- $S_0 = H(Z \| \text{"SEED"})$
- $W_i = \text{Trunc}_k(H(Z \| \text{"W"} \| i))$

To extrapolate coprimes for $P, Q, R$ (and optionally $M$), derive candidates from a seeded stream and select the first primes that are distinct and coprime with $x$:

1. $c_k \leftarrow \text{next\_prime}(H(Z \| \text{"PRIME"} \| k) \bmod 2^b)$
2. accept $c_k$ if $\gcd(c_k, x) = 1$ and $c_k \notin \{P, Q, R, M\}$
3. continue until $P, Q, R$ (and $M$ if generated) are assigned

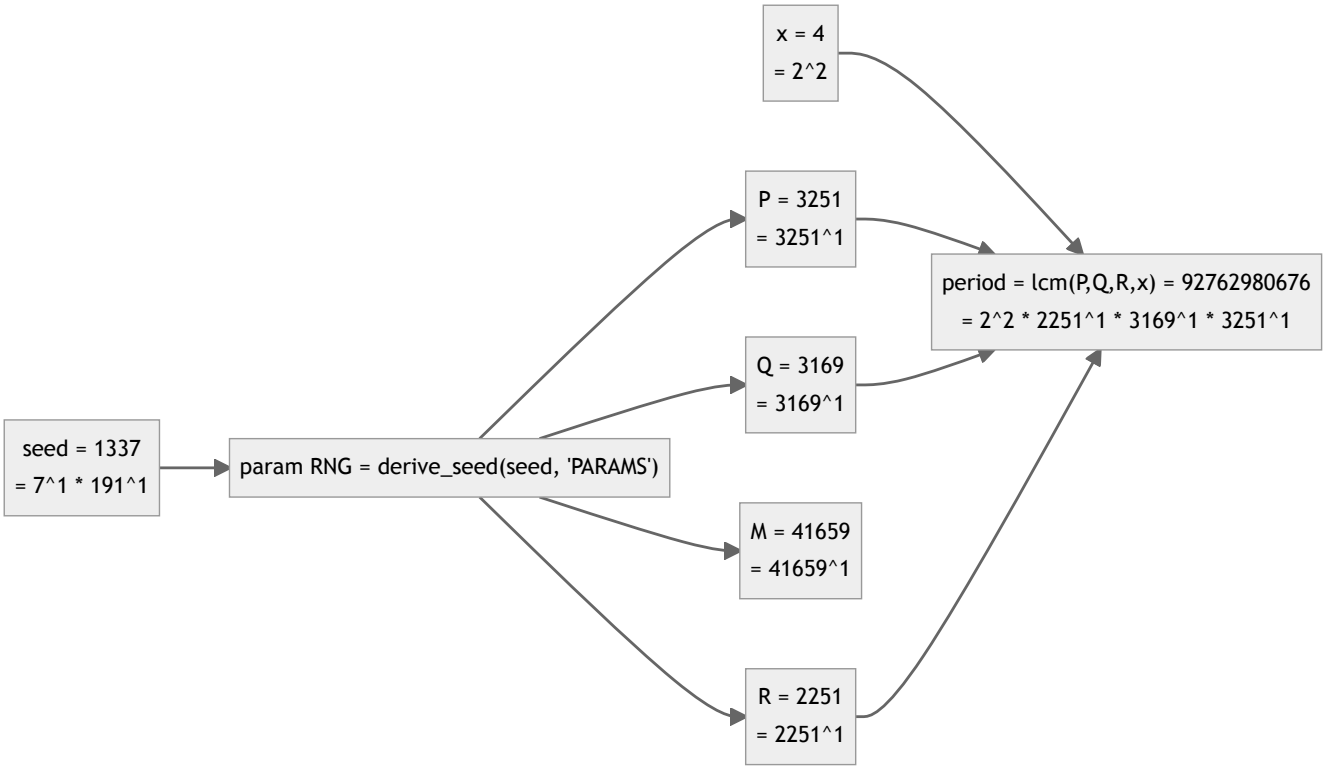
### 3.2 Best-practice coprimes, compounds, and key selection

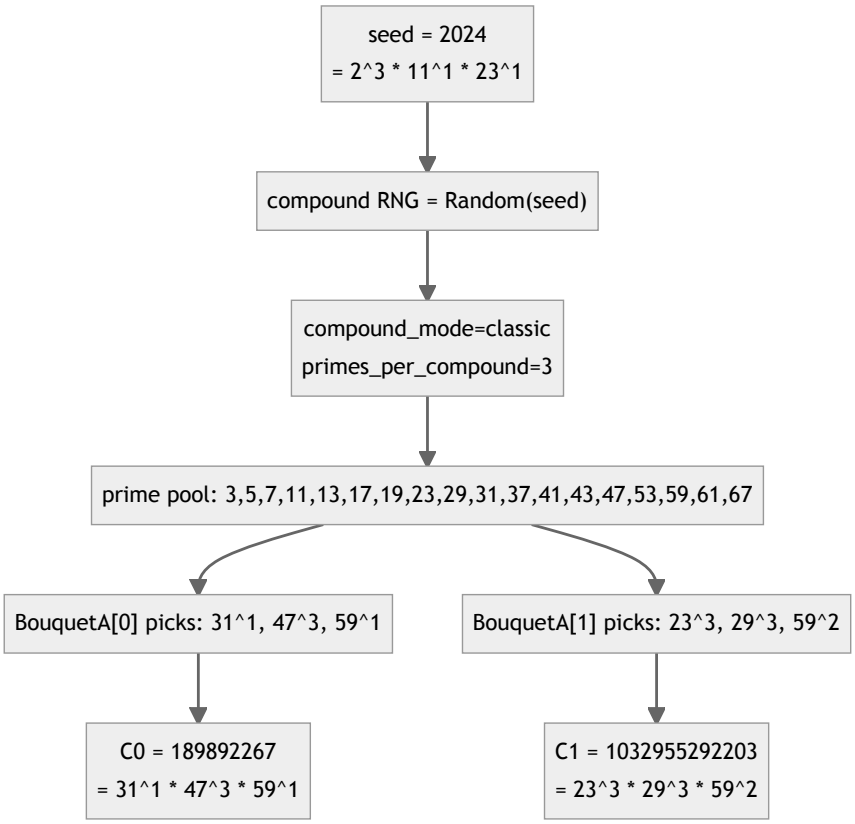
Parameter and key selection should scale with the peer count and keep strict domain separation between device-only and provider-only secrets.




### 3.2.1 Seeded example flows (real values)

The demo can be run with small bit sizes so prime-factor detail fits on the page. The examples below use `prime_mode=generated`, `prime_bits=12`, `modulus_bits=16`, `compound_mode=classic`, `compound_primes=3`, `compound_count=4`, and the built-in prime pool; the compound example uses provider 0's BouquetA[0…1]. Each node shows the integer value and its prime-power factorization.



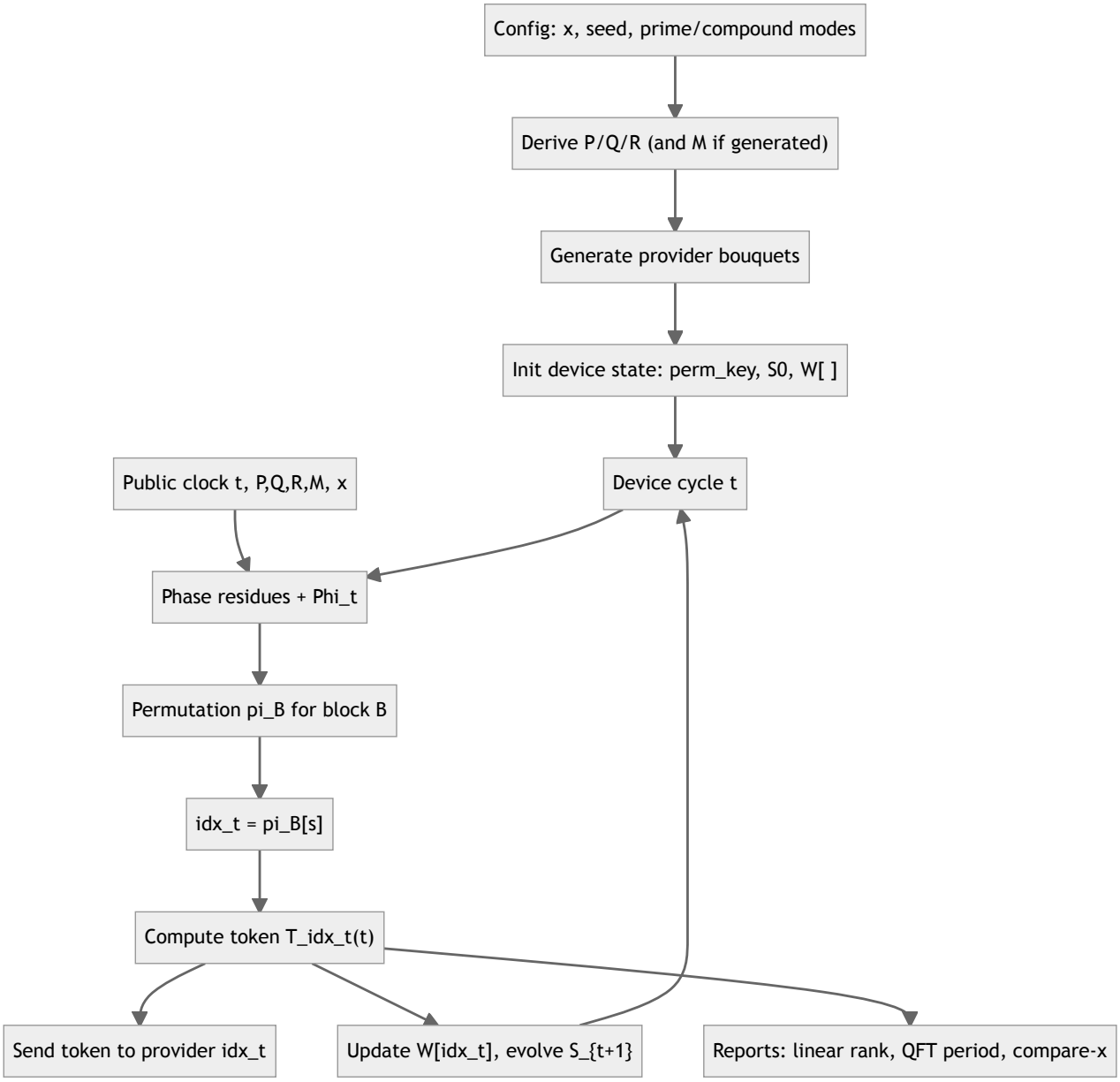

## 4. PCPL protocol overview

The protocol uses:

1. A public phase clock (CRT residues and coupled products).
2. A per-block permutation schedule to enforce “returns every $x$”.
3. Hidden bouquets to derive lane-specific tokens.
4. Device-only seed evolution that chains all lanes.

### 4.1 User device circuit (emitter)

The device knows the full schedule and all lane secrets, so it computes only the active lane per cycle and emits exactly one token.



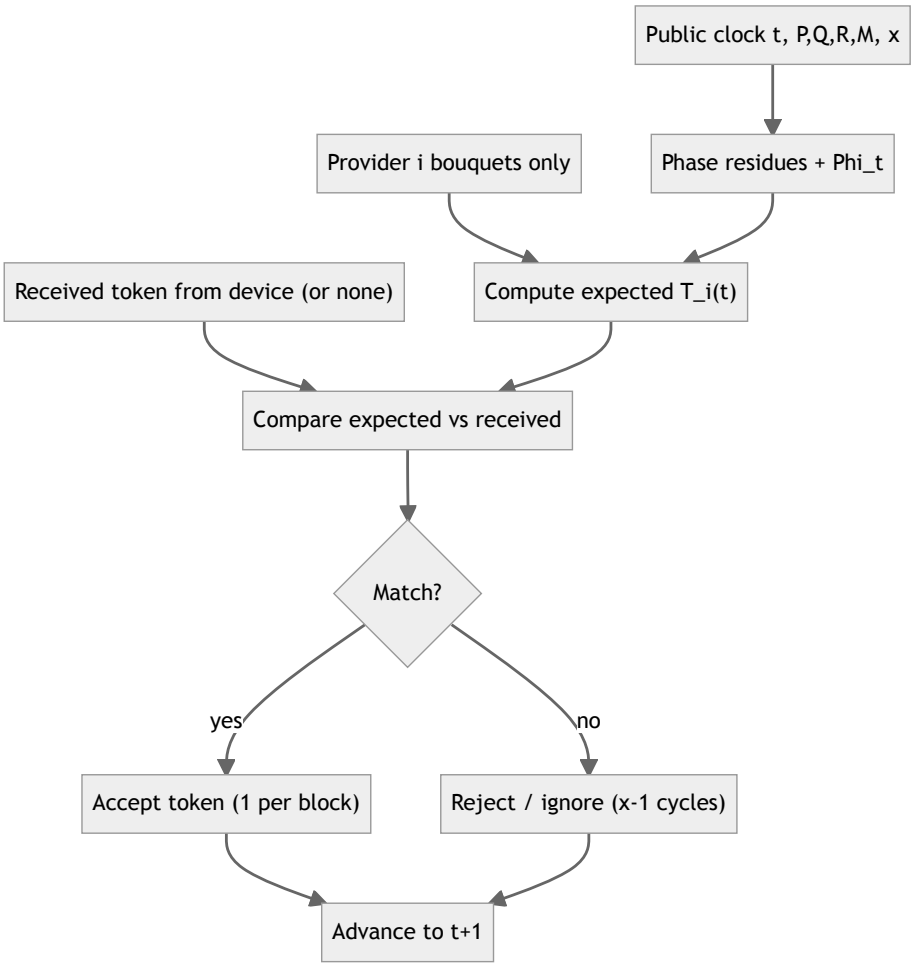

### 4.2 Blind provider circuit (validator)

Each provider only knows its own bouquets. It recomputes $T_i(t)$ every cycle, but the received token matches only once per block of $x$ cycles. The other $x - 1$ cycles are expected mismatches because the device emitted a different lane.

**Why only 1-of-$x$ is correct:** the provider computes the *same* lane token formula as the device, but with a fixed lane index $i$. The device emits $T_{\mathrm{idx}_t}(t)$ where $\mathrm{idx}_t = \pi_B[s]$ is hidden by $\mathrm{perm\_key}$. Therefore the provider is correct iff $i = \mathrm{idx}_t$.
Because $\pi_B$ is a permutation, this happens exactly once per block of $x$ cycles. The device is "always right" because it emits the scheduled lane token; the provider is "blind" because it does not know $\mathrm{perm\_key}$ and cannot predict which cycle is its match.

Provider-side token generation (every cycle):

$$
\begin{aligned}
EA_i(t) &= \mathrm{Eval}(\mathrm{BouquetA}_i, a_t, u_1),\\
EB_i(t) &= \mathrm{Eval}(\mathrm{BouquetB}_i, b_t, u_2),\\
EC_i(t) &= \mathrm{Eval}(\mathrm{BouquetC}_i, c_t, u_3),\\
K_i(t) &= H(EA_i \| EB_i \| EC_i \| \Phi_t \| \text{"KDF"}),\\
T_i(t) &= \mathrm{Trunc}_k(H(K_i \| t \| \Phi_t \| \text{"TOK"}))\,.
\end{aligned}
$$




## 5. Step-by-step algorithm

### 5.1 Phase clock

For cycle $t$:

$$
\begin{aligned}
a_t &= (a_0 + t) \bmod P,\\
b_t &= (b_0 + t) \bmod Q,\\
c_t &= (c_0 + t) \bmod R.
\end{aligned}
$$

Coupled products:

$$
\begin{aligned}
u_1 &= (a_t\, b_t) \bmod M,\\
u_2 &= (b_t\, c_t) \bmod M,\\
u_3 &= (c_t\, a_t) \bmod M.
\end{aligned}
$$

Phase digest:

$$
\Phi_t = H(a_t \| b_t \| c_t \| u_1 \| u_2 \| u_3 \| \text{"PHASE"})\,.
$$

### 5.2 Permutation schedule (“returns every x”)

Let:

$$B = \left\lfloor \frac{t}{x} \right\rfloor, \quad s = t \bmod x.$$

Compute a permutation $\pi_B$ of $\{0, \ldots, x-1\}$ using a hash-driven shuffle seeded by a block-level phase digest (computed at $t = B \cdot x$) so the schedule is stable within each block. Then:

$$\mathrm{idx}_t = \pi_B[s].$$

This guarantees each provider appears exactly once per block.

### 5.2.1 Device-side destination selection

The device determines the current destination provider using only public phase data and its private permutation key:

$$\begin{aligned} B &= \left\lfloor \frac{t}{x} \right\rfloor, \quad s = t \bmod x \\ \pi_B &= \mathrm{Permute}\left(\mathrm{perm\_key}, B, \Phi_{B \cdot x}\right) \\ \mathrm{idx}_t &= \pi_B[s] \end{aligned}$$

Providers do not know $\mathrm{perm\_key}$, so the schedule is blinded from them even though $t$ and $\Phi_t$ are public.

## 5.3 Bouquet evaluation

Each bouquet is a list of compounds $C_j$, each a product of primes. For a residue $x_{\mathrm{res}}$ and coupling $u$, define:

$$e_j = H(x_{\mathrm{res}} \| u \| j \| \text{"EXP"}) \bmod (M-1).$$

$$\mathrm{Eval}(\mathrm{Bouquet}, x_{\mathrm{res}}, u) = \prod_j C_j^{e_j} \bmod M.$$

For provider $i$:

$$\begin{aligned} EA_i(t) &= \mathrm{Eval}(\mathrm{BouquetA}_i, a_t, u_1), \\ EB_i(t) &= \mathrm{Eval}(\mathrm{BouquetB}_i, b_t, u_2), \\ EC_i(t) &= \mathrm{Eval}(\mathrm{BouquetC}_i, c_t, u_3). \end{aligned}$$

### 5.3.1 Prime-compound construction variants

Compounds do not need to be prime: any base coprime with $M$ is valid. Here, "prime compound" means a composite base built from two or more primes (a compound prime). This expands the base space and lets you tune complexity by increasing the number of factors and exponents, while preserving continuity.
The only hard requirement is $\gcd(C, M) = 1$ (no factor of $M$). This coprimality is **with respect to the modulus** $M$, not with respect to $P, Q, R$ or $x$: compounds may share factors with each other, but they must not share factors with $M$ to stay in $\mathbb{F}_M^*$.

- **Multi-prime compounds:** $C = \prod_{i=1}^{r} p_i^{e_i}$ with $r \geq 2$ (the general case).
- **Prime powers:** $C = p^k$ (smooth but non-prime bases).
- **Semiprimes:** $C = pq$ (a 2-prime special case).
- **Offset compounds:** $C = (\prod p_i^{e_i}) + \delta$ with small $\delta$ to create a quasi-continuous family.
- **Quantized reals:** map a real parameter $\rho$ to $C = \lfloor \alpha \rho \rfloor$ for fixed scale $\alpha$, then ensure $\gcd(C, M) = 1$.

The demo exposes these families via compound generation modes while keeping the exponent schedule unchanged; the "blend" mode just mixes these families and does not change the phase periodicity, which is driven solely by $P, Q, R$ and $x$.

## 5.4 Token derivation

Key derivation:

$$K_i(t) = H(EA_i \| EB_i \| EC_i \| \Phi_t \| \text{"KDF"}) .$$

Token:

$$T_i(t) = \text{Trunc}_k(H(K_i \| t \| \Phi_t \| \text{"TOK"})) .$$

### 5.5 Device emission and state evolution

The device computes only $T_{\text{idx}_t}(t)$ and updates internal state:

- $W[i]$ stores the last token for lane $i$.
- The seed $S$ evolves using all lanes and adjacent products.

For $x$ lanes, define (non-cyclic adjacency):

$$m_\ell = (W_\ell \cdot W_{\ell+1}) \bmod M, \quad \ell = 0, \ldots, x-2.$$

$$S_{t+1} = H(S_t \| W_0 \| \cdots \| W_{x-1} \| m_0 \| \cdots \| m_{x-2} \| \Phi_t \| \text{"EVOLVE"}) .$$

### 5.6 Provider verification

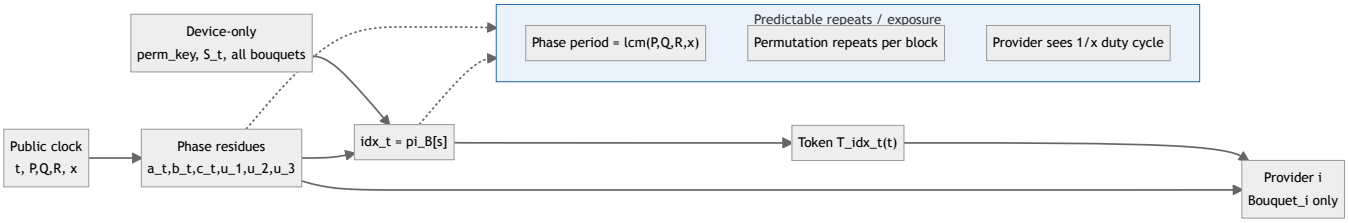
Provider $i$ recomputes $T_i(t)$ and accepts the token if it matches.

### 5.7 Device-side vs provider-side variables

The protocol deliberately separates what the device computes from what providers can infer:

- **Public inputs:** $t$, $x$, $P$, $Q$, $R$, $M$, and the permutation algorithm (but not the key).
- **Device-only state:** $\text{perm\_key}$, $S_t$, all lane secrets, and the last tokens $W[0..x-1]$.
- **Provider $i$ secrets:** $\text{BouquetA}_i, \text{BouquetB}_i, \text{BouquetC}_i$.
- **Ignored by providers:** $\text{perm\_key}$, $S_t$, other providers' bouquets, and the full $W$ vector.

The device computes only $T_{\text{idx}_t}(t)$ for the current lane; the provider computes only its own lane token and does not need the device seed.




## 6. Correctness and periodicity

### 6.1 Exact 1-of-x matching

Within each block of length $x$, $\pi_B$ is a permutation. Therefore each provider index appears exactly once per block, and exactly one provider matches per cycle.

### 6.2 Phase periodicity

The phase clock is three modular counters: $a_t = (a_0 + t) \bmod P$ and likewise for $b_t$ and $c_t$. The joint state repeats after the least common multiple of their moduli, so the period is $\mathrm{lcm}(P, Q, R)$. When $P, Q, R$ are pairwise coprime (i.e., $\gcd(P, Q) = \gcd(P, R) = \gcd(Q, R) = 1$), the lcm is $PQR$.

The *schedule* also depends on the block index $B = \lfloor t/x \rfloor$ and the slot $s = t \bmod x$, so the combined public schedule repeats after $\mathrm{lcm}(P, Q, R, x)$. If $x$ is coprime to each of $P, Q, R$ (i.e., $\gcd(P, x) = \gcd(Q, x) = \gcd(R, x) = 1$), then the schedule period is exactly $PQRx$.

If $x$ shares a factor with any of $P, Q, R$, the combined period is smaller. This periodicity is purely about the public clock; the choice of compound bases (even "blend" composites) does not change it, as long as those bases remain coprime with $M$.

### 6.3 Modular exponent correctness

With $M$ prime, the multiplicative group $\mathbb{F}_M^*$ has order $M - 1$.
Reducing exponents modulo $M - 1$ makes $C_j^{e_j} \bmod M$ well-defined for any base $C_j$ with $\gcd(C_j, M) = 1$. This holds for primes and composite compounds alike; the only disallowed case is a base sharing a factor with $M$, which would collapse the product (e.g., $C_j \equiv 0 \bmod M$).

### 6.4 Peer-count variations (x=2,3,4 and composite counts)

Changing $x$ changes the block size, the number of permutations, and the chain width:

| x | block length | permutations | chain products | note |
|---|---|---|---|---|
| 2 | 2 | 2 | 1 | twin pairing (2 lanes) |
| 3 | 3 | 6 | 2 | prime lane count |
| 4 | 4 | 24 | 3 | $2^2$ prime power |
| 6 | 6 | 720 | 5 | composite ($2 \cdot 3$) |

In general: block length $= x$, permutation space $= x!$, chain width $= x - 1$,
and schedule period $= \mathrm{lcm}(P, Q, R, x)$. For composite $x$ (e.g., $6 = 2 \cdot 3$),
choose $P, Q, R$ coprime with all prime factors of $x$ to avoid shrinking the period.

## 7. Security intuition (informal)

- **Lane isolation:** each provider uses distinct secret bouquets, so observing one lane does not reveal others.
- **Phase coupling:** public residues are mixed and hashed, preventing linear predictability from the CRT clock alone.
- **Device chaining:** even stale lanes influence future state, reinforcing the requirement that "every token matters".
- **Quantum period-finding:** QFT can reveal the public period $\mathrm{lcm}(P, Q, R, x)$ but not the hidden bouquets or $\mathrm{perm\_key}$; use large coprimes and device-only chaining to avoid exploitable structure.

## 8. Experimental validation (deterministic simulation)

A simulator was implemented a cycle-by-cycle to validate correctness. The demo verifies:

- Each block yields a valid permutation.
- Exactly one provider matches each cycle.
- Each provider appears once per block.
- Optional pre-hash difficulty metrics and QFT-visible period reports.

- Optional prime/compound generation modes for non-arbitrary parameter testing.

### 8.1 Sample token trace (x=4, seed=1337)

For PDF export, the original wide table was replaced with an A4-friendly summary table and a sequence diagram (tokens truncated for readability; the matched provider's recomputed token equals the device token by construction).

| t | block | slot | idx_t | device token (truncated) | matched provider |
|---|---|---|---|---|---|
| 0 | 0 | 0 | 3 | `0xaa81443d…6e0e02` | 3 |
| 1 | 0 | 1 | 0 | `0x21faa3d7…2dbe77` | 0 |
| 2 | 0 | 2 | 2 | `0x888b2137…903179` | 2 |
| 3 | 0 | 3 | 1 | `0xa591e8bf…03b4b4` | 1 |
| 4 | 1 | 0 | 2 | `0x5da9a61c…1d52ff` | 2 |
| 5 | 1 | 1 | 0 | `0x8abe0866…9d17b6` | 0 |
| 6 | 1 | 2 | 1 | `0x39d33ef1…6fd92e` | 1 |
| 7 | 1 | 3 | 3 | `0xe25bb134…064674` | 3 |


### 8.2 Full token trace (verbatim values)

The full deterministic trace (block permutations, schedule, device tokens, and per-lane tokens) is exported to a separate, auto-generated file to keep the paper A4-friendly, following the papers and script developed in repository [cekkr/phaselane-algorithm@github.com](). See `papers/token-trace.md`, generated by `demo/export_token_trace.py`.

Regenerate with:

```
python3 demo/export_token_trace.py --blocks 4 --out papers/token-trace.md
```

### 8.3 Pre-hash difficulty and period reporting

The demo can emit a linear pre-hash difficulty report (rank of exponent vectors modulo 2 and 65537) and the QFT-visible public period:

- `python3 demo/pcpl_cycle_test.py --linear-report --analysis-window 64`
- `python3 demo/pcpl_cycle_test.py --qft-report`
- `python3 demo/pcpl_cycle_test.py --compare-x 2,3,4,5,6`
- `python3 demo/pcpl_cycle_test.py --prime-mode generated --prime-bits 31 --compound-mode blend --compound-prime-bits 12`

### 8.4 Multi-configuration results snapshot

All runs below completed the full correctness checks (permutation, 1-of-x matching, chaining).

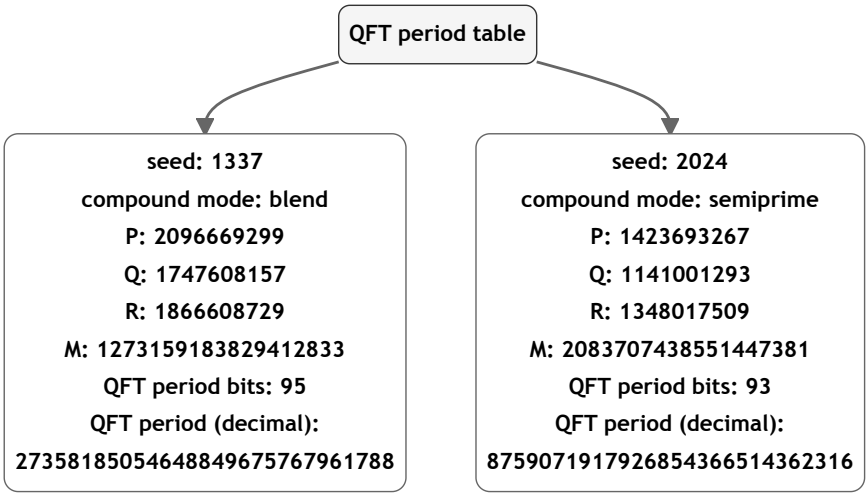
Fixed primes (P/Q/R near 1e6, seed=1337) with compare-x and 64-cycle linear window:

| x | chain width (x-1) | QFT period bits | QFT period (decimal) |
|---|---|---|---|
| 2 | 1 | 61 | 2000146002862007326 |
| 3 | 2 | 62 | 3000219004293010989 |
| 4 | 3 | 62 | 4000292005724014652 |
| 5 | 4 | 63 | 5000365007155018315 |
| 6 | 5 | 63 | 6000438008586021978 |

Across all x above, the pre-hash exponent vectors reached full rank (4/4) modulo 2 and 65537, with 64/64 unique rows for A/B/C over the sample window.

For $x = 6$ (composite $2 \cdot 3$), the schedule still yields exactly one match per cycle, but the duty cycle per provider is $1/6$ and the permutation space grows to $6! = 720$. Ensure $P, Q, R$ are coprime with both 2 and 3 to keep the public period large.

Generated primes (x=4, 64 cycles, 12-bit compound primes):




Full multi-configuration outputs (additional compound modes and seeds) are in `papers/pcpl-results.md`.

## 9. Discussion and limitations

- Parameter choice matters; $P, Q, R, M$ must be prime and pairwise coprime.
- The permutation schedule is device-only; leakage of the permutation key can reveal lane order, but not lane tokens.

- The security of the scheme relies on the strength of $H(\cdot)$ and the secrecy of bouquets, not on the hardness of factoring revealed integers.
- The public period $\mathrm{lcm}(P, Q, R, x)$ is visible (and QFT-recoverable), so period size should be chosen large enough for the deployment horizon.
- For testing, primes and compound bases can be generated from a seeded stream to avoid arbitrary constants.
- This paper was developed and formatted with an heavy OpenAI models' help.

## 10. Conclusion

PCPL provides a deterministic, no-handshake token protocol with exact 1-of-$x$ matching and a device-only chaining mechanism. Combined with symmetric continuous tokenizer devices, it supports both provider validation and peer-to-peer isolation with dynamic, evolving secrets. The included simulation and trace demonstrate the protocol's behavior cycle by cycle.