

# The 36-Minute Gacha Heist

Reverse-engineering Claude Code's hidden virtual pet to brute-force a Shiny Legendary Dragon.



```
> EXECUTE ./brute-force-buddy.sh --target="shiny legendary"
```

# A deterministic downgrade activated the gacha brain.

```
HASH_ID: 0x4F3A2E1D...
```

```
INIT_PET(hash);
```

CONTEXT

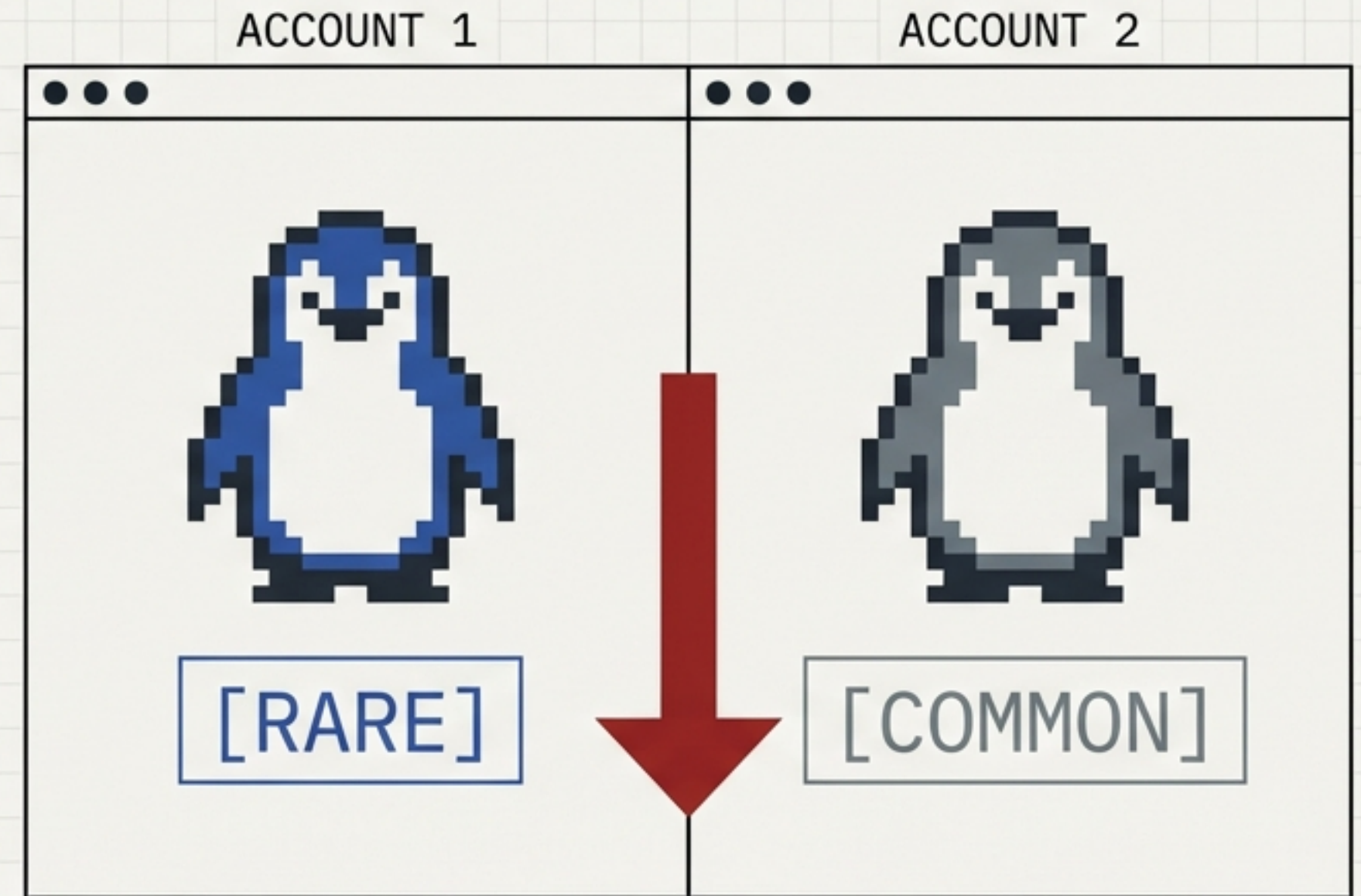
Claude Code v2.1.89 shipped an April Fools' Easter egg. Typing `/buddy` hatches a pixel-art companion that reacts to your coding.

THE CONSTRAINT

The pet is deterministic. Your account ID is hashed, and that hash permanently dictates your pet's appearance. **One account, one pet, forever.**

```
HASH_ID: 0x4F3A
```

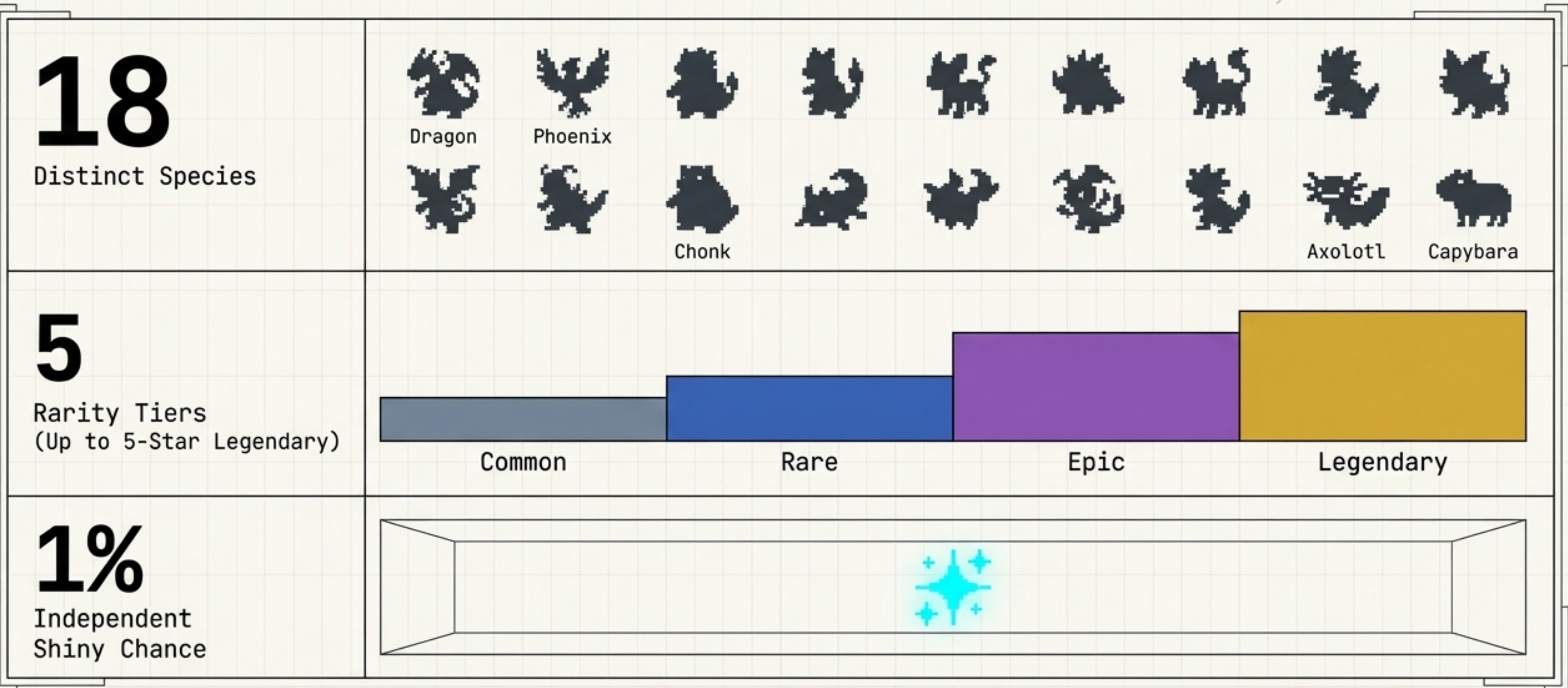
```
INIT_PET(hash);
```



```
INIT_PET(hash);
```

Two accounts. Same species. **Downgraded rarity.**  
The RNG gods weren't ignoring me – they were **taunting** me.

# Anthropic built a full, multi-layered gacha system inside a serious developer tool.



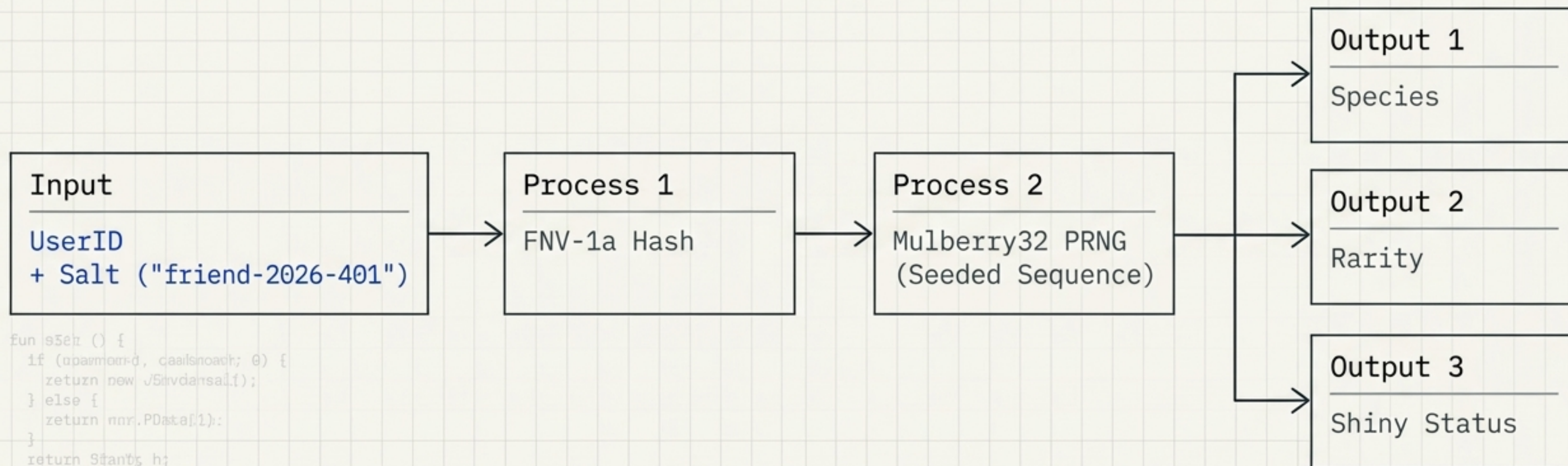
RARITY\_TIER\_DEF

SPECIES\_DB\_LOAD  
SHINY\_CALC\_FUNC  
)

# Hacking the hash to map the deterministic pipeline

Using Claude to analyze the compiled Bun CLI binary revealed the exact hashing algorithm and PRNG sequence in minutes, setting the stage for a brute-force attack.

```
const itue = system(Enabler);  
  
if:agwect {  
  const specifienö => {  
    public UserID + Salt ("friend-2026-401");  
  }  
}  
  
function Rowfiemand() {  
  return shiny.tsue  
}
```



```
fun s5ez () {  
  if (uparmerr-d, caalsnoach; 0) {  
    return new JSivdarsalf);  
  } else {  
    return nnr.PData[1];  
  }  
  return S5ant5 h;  
}
```

# Bending the RNG through millions of micro-iterations.

```
RNG_SEED_SEARCH = 1
ITERATION_COUNT = 50
TARGET_SPECIES_ID = 0
ITERATION_COUNT = 1
GOLDEN_LEGENDARY_CROWN = 1
```

Searching a massive hash space to find the exact local ID that forces the system to yield a Golden Legendary Dragon wearing a crown.

## The Quick Roll



**0.2 seconds**

Targeting:  
Species + Rarity + Shiny

Result: A generic  
Golden Golden  
Legendary Dragon.



## The Pixel-Perfect Roll

**4,547,133 Iterations**  
**9.75 seconds**

Tool: cc-buddy

Targeting:  
Species + Rarity + Shiny  
+ Hat (Crown) + Eyes

Result: The exact target  
Murmux.

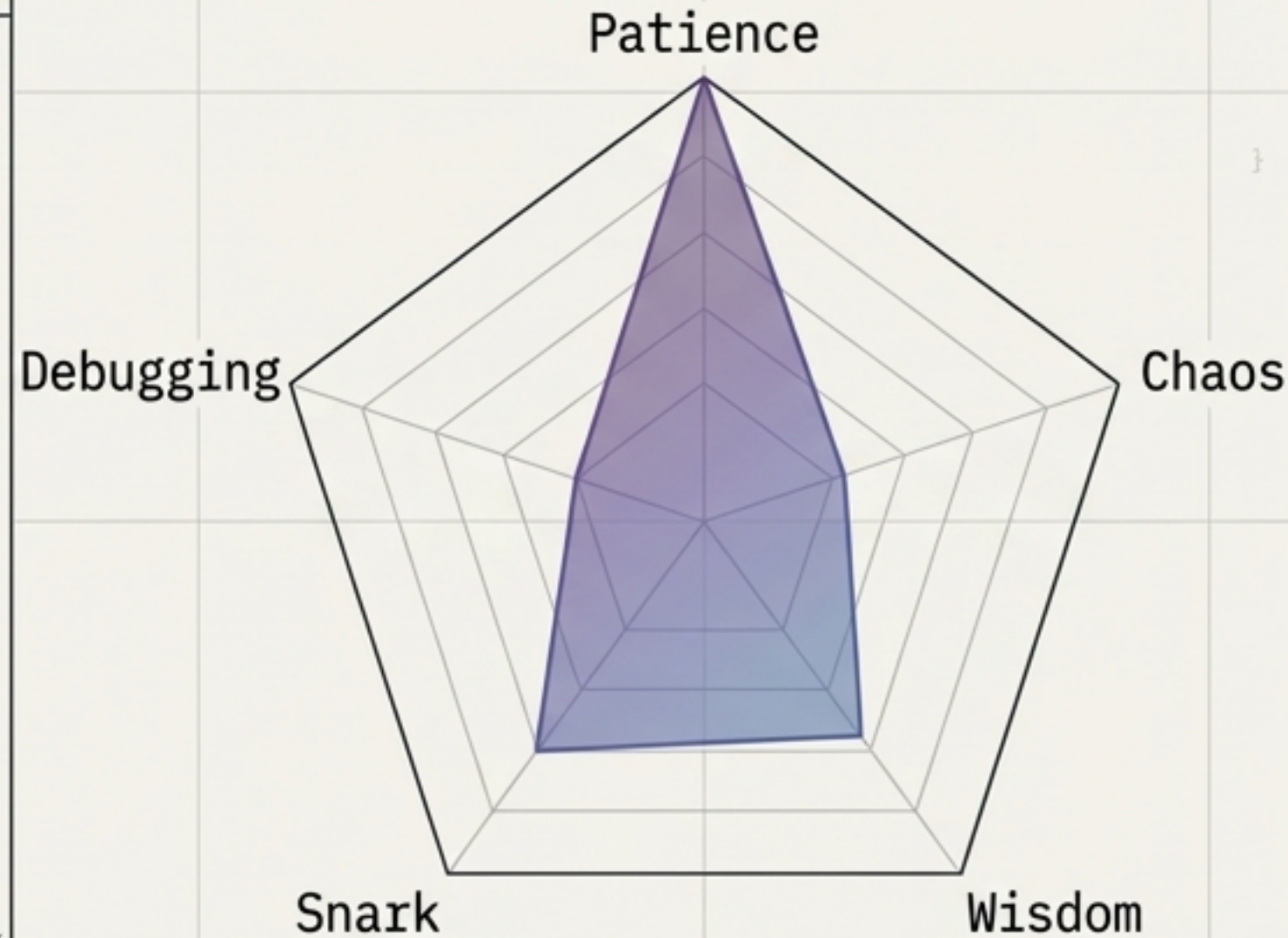


# The dual-layer architecture of a virtual companion.

## THE BONES

(Immutable)

- Hash-determined mechanics.
- Controls the visual appearance.
- Dictates species, rarity, and shiny status.
- This is what the brute-force script hacked.



## THE SOUL

(Mutable)

- AI-generated at hatch.
- Controls name and backstory.
- Defines behavioral quirks.
- Calculates 5-stat personality profile.

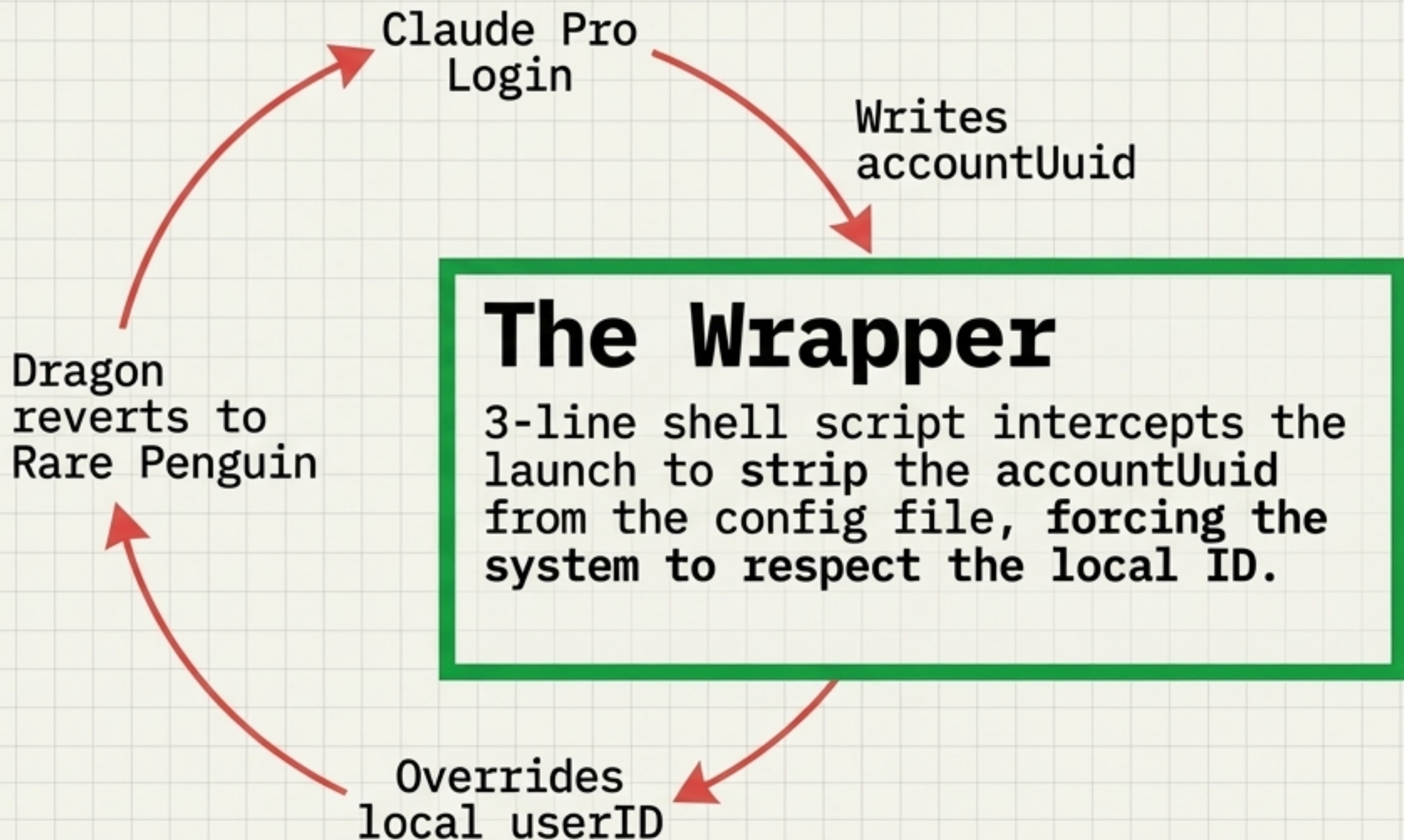
**The Irony:** Because the AI soul was generated before the hack, Murmux the Legendary Dragon has PATIENCE 100 and a backstory permanently describing him as "a perpetually unimpressed penguin."

# Bypassing the Claude Pro subscription override

Premium accounts enforce a cloud UUID that breaks the local hash exploit upon re-authentication.

The solution required zero AI execution and 100% human architectural understanding to intercept the launch sequence.

```
$xcli(accountUuid) {  
  if (variant == true  
      if (--Writes accountUuid) then  
        break  
      else  
        uuid <- "Use emilia's son asidf"  
        system+= "accountUuid"  
        axaxsnxyetm-accountUuid, --vegas-accession20"  
    }  
}
```



```
corporate = most {  
  haidok Pzo_coruPagmMl = ()  
  load LI w excent;  
}
```

# What coding with AI actually looks like in practice.

## AI Execution

9.95s (Extraction & Brute Force)

## Human Contribution

35m 50s (Systems thinking, community research, constraint design, wrapper logic, aesthetic choices)

Total time: 36 minutes.

Lines of code written by the human: **ZERO.**

```
import sys; sys.path.append('..')
import os; os.chdir('..')
import some-file:

functions.sei.execute(code) {
  let test = true;
  try {
    // use progress bar
    return processName + test;
  } catch {
    if (test === 'System thinking, community research') {
      execute = test;
      return result;
    }
  }
  if (study == null) {
    console.log('System thinking, community research');
    let code = test;
    return test;
  }
}
```

```
function main() {
  const args = process.argv;
  const code = args[2];

  let result = execute(code);

  // document the result
  while (true) {
    if (result) {
      console.log(result);
    } else if (args[3]) {
      console.log('Constraint design, wrapper logic, aesthetic choices');
    }
  }
}
```

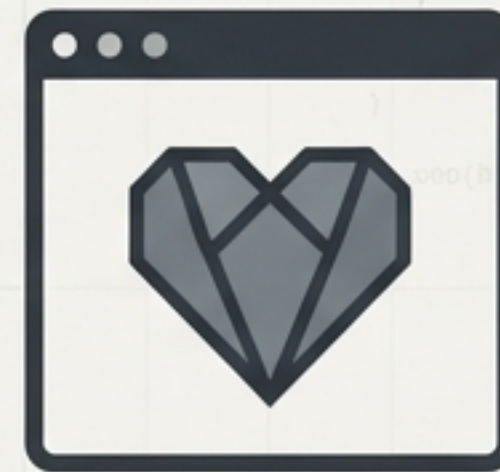
# The paradigm shift hidden inside an April Fools' joke.



## The New Developer Skillset

[Taste > Typing]

The interesting work wasn't implementing PRNG algorithms. It was understanding the architecture, specifying constraints, and recognizing edge cases.



## Product Culture Wins

[Joyful Engineering]

Anthropic lovingly built a highly complex, completely unnecessary gacha system inside a serious dev tool. That specific brand of joyful engineering creates products people actually love.