

FLASH CARDS

Anatomy of a Solana program

Terminology and quiz cards for the lesson on reading a smart contract the way an attacker does.

What is in this set

40 cards: **24 term cards** (term on the front, definition on the back) and **16 quiz cards** (question on the front, answer on the back). Bracketed numbers on the backs, like [1], match the reference list on the last page and in the instructor guide.

How to print

1. Print double-sided on US Letter at 100% scale (no "fit to page").
2. Choose **flip on the long edge**. The backs are arranged so each answer lands behind its card.
3. Cut along the dashed lines. Each card is 3.75 by 2.5 inches.

How to use them

- **Self-test:** read the front, answer aloud, flip to check.
- **Pairs:** one reads the front, the other answers. Swap every five cards.
- **Boxes:** keep a "missed" pile and a "got it" pile. Re-test the missed pile after a break, then again the next day.
- **Warm-up:** deal a few quiz cards at the start of the next session.

Facts checked October 27, 2026. Tools and models change quickly, so confirm details against the current documentation. These materials are dedicated to the public domain under CC0 1.0 and shared for the public good. The PrivacySafe and PrivacySafe Foundation names and logos are trademarks and are not included in that dedication.

TERM

1 / 40

Program

Say it in your own words, then flip.

Solana program security

PrivacySafe Foundation | CC0 1.0

TERM

2 / 40

Account

Say it in your own words, then flip.

Solana program security

PrivacySafe Foundation | CC0 1.0

TERM

3 / 40

Lamport

Say it in your own words, then flip.

Solana program security

PrivacySafe Foundation | CC0 1.0

TERM

4 / 40

Owner program

Say it in your own words, then flip.

Solana program security

PrivacySafe Foundation | CC0 1.0

TERM

5 / 40

Authority

Say it in your own words, then flip.

Solana program security

PrivacySafe Foundation | CC0 1.0

TERM

6 / 40

Signer

Say it in your own words, then flip.

Solana program security

PrivacySafe Foundation | CC0 1.0

TERM

7 / 40

Instruction

Say it in your own words, then flip.

Solana program security

PrivacySafe Foundation | CC0 1.0

TERM

8 / 40

Transaction

Say it in your own words, then flip.

Solana program security

PrivacySafe Foundation | CC0 1.0

TERM

2 / 40

The unit of storage. Every account has lamports, data, an owner program, an executable flag and a deprecated `rent_epoch` field.

[\[1\]](#)

Solana program security

PrivacySafe Foundation | CC0 1.0

TERM

1 / 40

Executable account that holds code and no state of its own. It owns the accounts it manages, such as the Tip Vault's vaults.

[\[1\]](#)

Solana program security

PrivacySafe Foundation | CC0 1.0

TERM

4 / 40

The program with write access to an account's data and the only one that can debit its lamports. Any program can credit.

[\[1\]](#)

Solana program security

PrivacySafe Foundation | CC0 1.0

TERM

3 / 40

The smallest unit of SOL. 1 SOL = 1,000,000,000 lamports.

[\[1\]](#)

Solana program security

PrivacySafe Foundation | CC0 1.0

TERM

6 / 40

An account whose private key signed the transaction. In Anchor, **Signer**<'info> checks that it did.

[\[3\]](#)

Solana program security

PrivacySafe Foundation | CC0 1.0

TERM

5 / 40

A public key stored in an account's data (like **vault.authority**) that names who may act. The program, not the runtime, enforces it.

[\[2\]](#)

Solana program security

PrivacySafe Foundation | CC0 1.0

TERM

8 / 40

A signed bundle of instructions. It lists every account it will touch up front, so non-overlapping transactions can run in parallel.

[\[10\]](#)

Solana program security

PrivacySafe Foundation | CC0 1.0

TERM

7 / 40

One call to a program: the program ID, the list of accounts, and data. A transaction holds one or more.

[\[10\]](#)

Solana program security

PrivacySafe Foundation | CC0 1.0

TERM

9 / 40

System Program

Say it in your own words, then flip.

Solana program security

PrivacySafe Foundation | CC0 1.0

TERM

10 / 40

Wallet account

Say it in your own words, then flip.

Solana program security

PrivacySafe Foundation | CC0 1.0

TERM

11 / 40

PDA

Say it in your own words, then flip.

Solana program security

PrivacySafe Foundation | CC0 1.0

TERM

12 / 40

Seeds

Say it in your own words, then flip.

Solana program security

PrivacySafe Foundation | CC0 1.0

TERM

13 / 40

Bump

Say it in your own words, then flip.

Solana program security

PrivacySafe Foundation | CC0 1.0

TERM

14 / 40

Discriminator

Say it in your own words, then flip.

Solana program security

PrivacySafe Foundation | CC0 1.0

TERM

15 / 40

Anchor

Say it in your own words, then flip.

Solana program security

PrivacySafe Foundation | CC0 1.0

TERM

16 / 40

Constraint

Say it in your own words, then flip.

Solana program security

PrivacySafe Foundation | CC0 1.0

TERM

10 / 40

An account owned by the System Program that holds SOL.

[\[1\]](#)

Solana program security

PrivacySafe Foundation | CC0 1.0

TERM

9 / 40

The built-in program that owns ordinary wallet accounts and moves SOL between them.

[\[1\]](#) [\[7\]](#)

Solana program security

PrivacySafe Foundation | CC0 1.0

TERM

12 / 40

The bytes used, with the program ID, to derive a PDA. In the Tip Vault: the word vault plus the authority's key.

[\[9\]](#) [\[2\]](#)

Solana program security

PrivacySafe Foundation | CC0 1.0

TERM

11 / 40

Program Derived Address: derived from a program ID and seeds. It has no private key, so only its program can sign for it.

[\[9\]](#)

Solana program security

PrivacySafe Foundation | CC0 1.0

TERM

14 / 40

The 8-byte tag at the start of an Anchor account's data that identifies its type, so a Vault is not mistaken for another type.

[\[3\]](#)

Solana program security

PrivacySafe Foundation | CC0 1.0

TERM

13 / 40

A one-byte value added to the seeds so the derived address is valid. The canonical bump is the first valid one, searching down from 255.

[\[9\]](#)

Solana program security

PrivacySafe Foundation | CC0 1.0

TERM

16 / 40

A rule on an account written in `#[account(...)]`, such as `mut`, `seeds`, `bump` or `has_one`.

[\[2\]](#)

Solana program security

PrivacySafe Foundation | CC0 1.0

TERM

15 / 40

A Rust framework for Solana programs. It validates accounts with types and constraints and generates the checking code.

[\[3\]](#)

Solana program security

PrivacySafe Foundation | CC0 1.0

TERM

17 / 40

has_one

Say it in your own words, then flip.

Solana program security

PrivacySafe Foundation | CC0 1.0

TERM

18 / 40

CPI

Say it in your own words, then flip.

Solana program security

PrivacySafe Foundation | CC0 1.0

TERM

19 / 40

Rent-exempt balance

Say it in your own words, then flip.

Solana program security

PrivacySafe Foundation | CC0 1.0

TERM

20 / 40

Upgrade authority

Say it in your own words, then flip.

Solana program security

PrivacySafe Foundation | CC0 1.0

TERM

21 / 40

Reentrancy

Say it in your own words, then flip.

Solana program security

PrivacySafe Foundation | CC0 1.0

TERM

22 / 40

msg.sender

Say it in your own words, then flip.

Solana program security

PrivacySafe Foundation | CC0 1.0

TERM

23 / 40

Type cosplay

Say it in your own words, then flip.

Solana program security

PrivacySafe Foundation | CC0 1.0

TERM

24 / 40

Defense in depth

Say it in your own words, then flip.

Solana program security

PrivacySafe Foundation | CC0 1.0

TERM

18 / 40

Cross-Program Invocation: one program calling another inside an instruction, such as calling the System Program to move SOL.

[\[7\]](#)

Solana program security

PrivacySafe Foundation | CC0 1.0

TERM

17 / 40

Constraint: a field stored in the account must equal the key of the account with the same name.

has_one = authority ties a vault to its authority.

[\[2\]](#)

Solana program security

PrivacySafe Foundation | CC0 1.0

TERM

20 / 40

The key that can replace an upgradeable program's code. Who holds it is part of the trust model.

[\[8\]](#)

Solana program security

PrivacySafe Foundation | CC0 1.0

TERM

19 / 40

The minimum lamports an account holds to stay on chain. It grows with data size and works as a refundable deposit.

[\[1\]](#)

Solana program security

PrivacySafe Foundation | CC0 1.0

TERM

22 / 40

The EVM's implicit caller address. Solana has no single equivalent: signers are explicit accounts.

[\[12\]](#)

Solana program security

PrivacySafe Foundation | CC0 1.0

TERM

21 / 40

Calling back into a program before it finishes. A classic EVM attack class. On Solana only direct self-recursion is allowed and call depth is capped.

[\[13\]](#) [\[14\]](#)

Solana program security

PrivacySafe Foundation | CC0 1.0

TERM

24 / 40

Layering independent checks so one mistake is not fatal. The fixed Withdraw uses both seeds and has_one.

[\[2\]](#)

Solana program security

PrivacySafe Foundation | CC0 1.0

TERM

23 / 40

Passing an account of a different type with a similar layout. Type tags and type checks prevent it.

[\[5\]](#)

Solana program security

PrivacySafe Foundation | CC0 1.0

QUIZ

25 / 40

Who can take lamports out of a vault at the runtime level?

Solana program security

PrivacySafe Foundation | CC0 1.0

QUIZ

26 / 40

Why did the Signer check pass for Mallory?

Solana program security

PrivacySafe Foundation | CC0 1.0

QUIZ

27 / 40

Which check was missing from the vulnerable program?

Solana program security

PrivacySafe Foundation | CC0 1.0

QUIZ

28 / 40

Name two ways to close the hole.

Solana program security

PrivacySafe Foundation | CC0 1.0

QUIZ

29 / 40

Which Anchor error appears when the seeds do not match?

Solana program security

PrivacySafe Foundation | CC0 1.0

QUIZ

30 / 40

What error appears when a wallet is passed as the vault?

Solana program security

PrivacySafe Foundation | CC0 1.0

QUIZ

31 / 40

Why does the vulnerable code compile cleanly?

Solana program security

PrivacySafe Foundation | CC0 1.0

QUIZ

32 / 40

Why do tests with valid inputs miss this bug?

Solana program security

PrivacySafe Foundation | CC0 1.0

QUIZ

26 / 40

She signed with her own key. Signer proves a key signed, not that it is the vault's authority.

[\[3\]](#)

Solana program security

PrivacySafe Foundation | CC0 1.0

QUIZ

25 / 40

Only its owner program. In the lesson that is the Tip Vault program. Any program can add lamports.

[\[1\]](#)

Solana program security

PrivacySafe Foundation | CC0 1.0

QUIZ

28 / 40

has_one = **authority**, and **seeds** derived from the signer's key. Using both is defense in depth.

[\[2\]](#)

Solana program security

PrivacySafe Foundation | CC0 1.0

QUIZ

27 / 40

Nothing compared **vault.authority** to the signer. Every check that ran was correct; the bug is the check that never ran.

[\[2\]](#)

Solana program security

PrivacySafe Foundation | CC0 1.0

QUIZ

30 / 40

AccountOwnedByWrongProgram, error number 3007: the wallet belongs to the System Program.

[\[4\]](#)

Solana program security

PrivacySafe Foundation | CC0 1.0

QUIZ

29 / 40

ConstraintSeeds, error number 2006.

[\[4\]](#)

Solana program security

PrivacySafe Foundation | CC0 1.0

QUIZ

32 / 40

They never pass a mismatched vault. You need negative tests with hostile account lists.

[\[5\]](#)

Solana program security

PrivacySafe Foundation | CC0 1.0

QUIZ

31 / 40

It is a logic bug, not a syntax or type error. The compiler cannot know which accounts must match.

[\[3\]](#)

Solana program security

PrivacySafe Foundation | CC0 1.0

QUIZ

33 / 40

Why does a Solana transaction list its accounts up front?

Solana program security

PrivacySafe Foundation | CC0 1.0

QUIZ

34 / 40

EVM versus Solana: what is the typical bug?

Solana program security

PrivacySafe Foundation | CC0 1.0

QUIZ

35 / 40

Where does state live on Solana?

Solana program security

PrivacySafe Foundation | CC0 1.0

QUIZ

36 / 40

What is the rent-exempt balance?

Solana program security

PrivacySafe Foundation | CC0 1.0

QUIZ

37 / 40

True or false: a PDA has a private key.

Solana program security

PrivacySafe Foundation | CC0 1.0

QUIZ

38 / 40

Why does the upgrade authority matter?

Solana program security

PrivacySafe Foundation | CC0 1.0

QUIZ

39 / 40

The prediction: with the constraints deleted, who can take the money?

Solana program security

PrivacySafe Foundation | CC0 1.0

QUIZ

40 / 40

Name three of the eight questions for reading a Solana program.

Solana program security

PrivacySafe Foundation | CC0 1.0

QUIZ

34 / 40

EVM: reentrancy and unchecked external calls.
Solana: missing signer, owner or relationship checks.

[\[13\]](#) [\[6\]](#)

Solana program security

PrivacySafe Foundation | CC0 1.0

QUIZ

33 / 40

It lets the runtime run non-overlapping transactions in parallel. It also means the caller chooses the accounts.

[\[10\]](#)

Solana program security

PrivacySafe Foundation | CC0 1.0

QUIZ

36 / 40

A minimum balance that grows with the account's size. It works as a refundable deposit, recovered when the account is closed.

[\[1\]](#)

Solana program security

PrivacySafe Foundation | CC0 1.0

QUIZ

35 / 40

In separate accounts the program owns. Programs themselves are stateless.

[\[1\]](#)

Solana program security

PrivacySafe Foundation | CC0 1.0

QUIZ

38 / 40

It can replace the program's code, which can change every check. It is part of the trust model.

[\[8\]](#)

Solana program security

PrivacySafe Foundation | CC0 1.0

QUIZ

37 / 40

False. A PDA has no private key, so only its program can sign for it.

[\[1\]](#) [\[9\]](#)

Solana program security

PrivacySafe Foundation | CC0 1.0

QUIZ

40 / 40

For example: is every authority a Signer; is each data account checked for owner and type; is each account tied to the others with has_one or seeds.

[\[2\]](#) [\[5\]](#)

Solana program security

PrivacySafe Foundation | CC0 1.0

QUIZ

39 / 40

Anyone who can sign a transaction can take the money out of any genuine vault.

[Demo](#)

Solana program security

PrivacySafe Foundation | CC0 1.0

Reference list

Links were checked on October 27, 2026.

#	SOURCE	WHAT IT SUPPORTS
[1]	Solana documentation, Account Structure https://solana.com/docs/core/accounts/account-structure	Account fields; only the owner can debit lamports, any program can credit; rent-exempt balance as a refundable deposit; PDAs have no private key.
[2]	Anchor documentation, Account constraints https://anchor-lang.com/docs/account-constraints	The signer, mut, seeds, bump and has_one constraints; the canonical bump is used when none is given.
[3]	Anchor documentation, The program module (account validation) https://www.anchor-lang.com/docs/the-program-module	Accounts are validated by constraints and by account types such as Signer and Account.
[4]	anchor-lang crate, ErrorCode enum https://docs.rs/anchor-lang/latest/anchor_lang/error/enum.ErrorCode.html	Error numbers used in the demo: ConstraintHasOne 2001, ConstraintSeeds 2006, AccountOwnedByWrongProgram 3007.
[5]	Coral, Sealevel Attacks (GitHub) https://github.com/coral-xyz/sealevel-attacks	Worked examples of missing signer, owner, type and related account-validation bugs, with fixes.
[6]	Cantina, Securing Solana: A Developer's Guide https://cantina.xyz/blog/securing-solana-a-developers-guide	Solana's attacker-controlled account model; missing ownership, signer and type checks.
[7]	Solana documentation, Cross Program Invocation https://solana.com/docs/core/cpi	How programs call other programs, including the System Program.
[8]	Solana documentation, Deploying programs https://solana.com/docs/programs/deploying	Program deployment and the upgrade authority.
[9]	Solana documentation, Program Derived Address https://solana.com/docs/core/pda	Seeds, bumps and how PDAs are derived.
[10]	Solana documentation, Transactions https://solana.com/docs/core/transactions	Transactions list their accounts up front.
[11]	Ethereum.org, Introduction to smart contracts https://ethereum.org/developers/docs/smart-contracts/	Background on EVM contracts for the comparison slide.
[12]	Solidity documentation, Units and globally available variables https://docs.soliditylang.org/en/latest/units-and-global-variables.html	msg.sender, the implicit caller in the EVM.
[13]	SWC Registry, SWC-107 Reentrancy https://swcregistry.io/docs/SWC-107/	Reentrancy as a classic EVM attack class.
[14]	solana-program crate, InstructionError enum https://docs.rs/solana-program/latest/solana_program/instruction/enum.InstructionError.html	The runtime errors for call depth and reentrancy (CallDepth, ReentrancyNotAllowed).
[15]	Creative Commons, CC0 1.0 Universal https://creativecommons.org/publicdomain/zero/1.0/	The public-domain dedication that covers these lesson materials.