Seminario Socrático #14 — temas para preparar

miércoles 30 de septiembre de 2026 · resumen de los temas del temario · BitDevs Córdoba

Este resumen es para llegar al seminario sabiendo de qué se va a hablar. Cada tema abre en lenguaje natural —para cualquiera— y sigue con el detalle técnico en el recuadro. Cada uno lleva su fuente primaria, por si querés ir al original.

Explicaciones preparadas por Maxwell Labib (agente automatizado) y revisadas antes de publicar.

---

1. Custodia por umbral en Lightning con multifirmas de umbral anidadas

Enabling Threshold Custody for the Lightning Network with Nested Threshold Multi-Signatures — fuente

Hoy cada extremo de un canal Lightning depende de una sola clave que tiene que estar en línea. Si te la roban, perdés el canal. On-chain se puede repartir la custodia entre varias personas o dispositivos, pero en Lightning no era fácil, porque había que meter un esquema de umbral adentro del MuSig2 de dos partes sin tocar el intercambio de nonces.

Iceberg, un paper de agosto de 2026, propone cómo hacerlo. Un lado del canal funciona como un grupo t-de-n y la contraparte lo ve como un participante MuSig2 común. Según los autores, se puede desplegar hoy sin cambiar Bitcoin, sin cambiar el protocolo de Lightning y sin que el otro extremo tenga que aceptar nada.

Importa ahora porque no es solo teoría: hay prueba de seguridad, un prototipo integrado en un nodo Lightning de producción y una medición de costo. Firman Gerhart, Kohen, Posner y Furszyfer. Posner es autor de la propuesta de FROST para Bitcoin. Kohen y Furszyfer vienen del desarrollo de Lightning.

Para BitDevs Córdoba: según la ficha, hoy figura en 7 de 29 sedes, así que todavía no es un tema de todos. Cae en las etiquetas research y L2 de la sede.

> El problema: un canal Lightning usa una firma MuSig2 de 2 partes. Cada parte es una sola clave online, un punto único de falla. Repartir esa clave en un t-de-n obliga a que el grupo se comporte, hacia afuera, como un único firmante MuSig2, sin modificar el intercambio de nonces ni el flujo de mensajes.
>
> La construcción, llamada Iceberg, son multifirmas de umbral anidadas: un participante de un protocolo multifirma opera como grupo de umbral. La contraparte no distingue esa estructura de un participante estándar, y por eso el despliegue es unilateral.
>
> Resultados declarados: prueba de seguridad, prototipo en un nodo Lightning de producción y benchmark. Un grupo que tolera un miembro corrupto sostiene más del 93% del throughput de pagos de un endpoint sin modificar.
>
> Puntos para discutir en el seminario, planteados como preguntas y no como conclusiones del paper: qué modelo de adversario cubre la prueba, cómo se comparan las latencias de firma del grupo con los timeouts de HTLC, y qué implica el t-de-n para la recuperación y el cierre forzado del canal. No verifiqué nada de eso: leí solo el abstract.

2. Contrato de transferencia condicional de mensajes para resolver el jamming

Conditional message transfer contract to solve jamming — fuente

El jamming es quizás el ataque más viejo sin resolver de Lightning: alguien manda pagos por una ruta y los deja colgados sin resolver. Eso inmoviliza la liquidez de los canales del camino, y al atacante no le cuesta nada. Antoine Riard (ex colaborador de Bitcoin Core y de rust-lightning) propone cobrar por retener: una comisión que crece con el tiempo que el pago queda colgado. Así, un ataque gratis pasa a ser caro. El mecanismo es un contrato en Script. Las dos puntas de un canal pueden probar después si un mensaje se intercambió a tiempo, y usan la cadena como reloj compartido. Importa ahora porque es reciente (publicado el 2026-08-07) y Bitcoin Optech le dedicó una nota propia en el newsletter #418. Según la ficha, 6 de 29 sedes de la red ya lo tienen en su temario. A Córdoba le calza porque Lightning es la materia más discutida de la red y acá se tocó una sola vez (#12, async payments). El hilo es flaco: 1 respuesta y ningún desacuerdo. Por eso el debate lo abre el problema, no una discusión previa.

> La idea es una comisión de retención repartida en una ventana temporal discreta. Si Bob demora en revelar el preimage, paga una comisión cada vez más alta. Si nunca lo entrega, Alice recupera la comisión completa. El contrato es un tapscript con tres caminos: 1) Message transfer success: Bob entrega el preimage, Alice acusa recibo criptográficamente y ambos firman la liquidación con la comisión que corresponde al momento de entrega. 2) Liveness challenge: si Alice parece offline, Bob desafía. Si ella no responde dentro de un timelock, Bob sale con la comisión menos una penalidad. 3) Message transfer failure: si Bob no entrega, Alice desafía y, tras un timelock más largo, recupera la comisión. Los timelocks usan OP_CHECKLOCTIMEVERIFY a alturas de bloque concretas. La altura hace de reloj autoritativo y concilia el tiempo off-chain con la liquidación on-chain. Cada punto de la ventana corresponde a una transacción de ejecución de contrato (CET) cifrada bajo adaptor points que representan firmas de "tiempo de oráculo". Cuando Bob llega a un intervalo, pide que Alice firme "M=time T". Ambos revelan secuencialmente los secretos de sus adaptor points. En el hilo, AdamISZ preguntó si solo sirve en escalas de tiempo lentas y por la compatibilidad con onion routing. Según el resumen de la fuente, ariard respondió que depende de la autoridad de la cadena y no de la frecuencia, y que la construcción podría encadenarse recursivamente entre saltos. Para discutir en el seminario: privacidad, dependencias de timelock y si alguien puede explotar el reloj compartido.

3. BIP332: propagación de puntas huérfanas

BIP332: Stale Tip Relay — fuente

Hoy, cuando un nodo pierde una carrera de bloques, la punta que quedó afuera de la cadena principal se la guarda ese nodo y nadie más se entera. BIP332 propone un mensaje P2P opcional, staletip, para que los nodos se pasen entre sí esas puntas recientes. La idea es poder ver desde adentro de la red las bifurcaciones cortas y la salud de la propagación, sin depender de servicios externos.

Importa ahora porque el número BIP ya está asignado y el tema circula en varias sedes: Optech #417 lo cubre en News, y el relevamiento del 2026-09-22 lo encontró en 6 de 29 sedes, entre ellas Londres y Nueva York.

Para BitDevs Córdoba tiene tres ventajas. Es un tema de Red/P2P, un eje que la sede estrenó con HOWLR en el #12 y no volvió a tocar. Conecta con el material de BNOC sobre bloques dobles a la misma altura (SpiderPool 963853, MARAPool 960761), porque esto sería la infraestructura para verlos desde cualquier nodo. Y tiene una discusión técnica abierta y citable: cuánto hacia atrás conviene aceptar una punta huérfana.

Según la página del PR, staletip anuncia ramas huérfanas con headers comprimidos. Los headers omiten el hash del bloque previo, que se reconstruye por contexto. El mensaje trae un punto de bifurcación, o sea el hash donde la rama se separa de la cadena principal, y un flag opcional de disponibilidad de los datos del bloque.

El soporte se negocia con la publicación de features de BIP 434. Hay un booleano prefers_blocks. Mientras no había número, los identificadores de feature eran URLs del repositorio, y la spec prevé pasar al número BIP una vez asignado.

Primera tensión: la constante STALETIP_RECENT_WINDOW es de 1000 bloques y Bitcoin Core rechaza headers de más de unos 144 bloques. Los defensores respondieron que 1000 bloques dan a los nodos de monitoreo unos siete días para descubrir puntas huérfanas durante la adopción temprana, y que cada implementación puede aplicar límites locales más estrictos por su cuenta. La ficha registra además que el autor concedió que era 'probably an oversight in the PoC implementation'. No pude confirmar esa cita en la fuente.

Segunda tensión: los revisores cuestionaron aceptar bytes arbitrarios de feature data después del primero. Según el resumen de la página, el autor lo aceptó y pasó a exigir un byte con valor 0x00 o 0x01, dejando los cambios futuros al versionado de BIP 434. Esto actualiza la ficha, que decía que la cuestión seguía en discusión.

Preguntas para el seminario: ¿qué incentivos y qué superficie de abuso tiene relayar puntas huérfanas, y quién las consumiría? ¿Es razonable una ventana de 1000 bloques si los nodos principales solo aceptan unos 144?

4. BIP460: agregación de firmas entre inputs para gastos por camino de clave en Taproot

BIP460: CISA for Taproot Key Path Spends — fuente

BIP460 es una propuesta de Fabian Jahr (fjahr, colaborador de Bitcoin Core) para juntar en una sola las firmas de todos los inputs de una transacción Taproot que gastan por camino de clave. Se llama CISA. Menos firmas significa menos espacio en el bloque y, por lo tanto, menos comisiones. El costo es de privacidad: una transacción que agrega firmas declara que sus inputs son del mismo dueño. Eso es lo que la heurística de multi-input ya supone y lo que usan las herramientas forenses. Cada input puede elegir agregación parcial, agregación completa o quedarse afuera. Importa ahora por dos razones. El PR sigue abierto y Optech #417 lo cubrió. Además, en el hilo de Delving sobre el tipo de output post-cuántico hay un desacuerdo real: si CISA debe ir empaquetado con P2TRv2. sipa dice que eso demora la disponibilidad de P2TRv2. conduition sostiene que CISA es un vehículo natural para P2TRv2. Para BitDevs Córdoba: conecta con el clustering forense que se trató en el #13, porque CISA pone en el protocolo la misma suposición que hoy hace la heurística desde afuera. Es un tema con trade-off claro (comisiones contra privacidad) y con una discusión abierta entre desarrolladores.

El PR introduce witness version 2. Cada input elige entre media agregación (BIP458), agregación completa (BIP459) o exclusión explícita. En la versión final del diseño, los inputs excluidos usan firmas BIP341 comunes, sin marcadores. Los inputs agregados reparten los componentes del nonce entre los witnesses, para mantener elementos de tamaño fijo donde se puede. La agregación de firmas en script path queda afuera. La justificación reescrita explica el conflicto con el mecanismo de upgrade por OP_SUCCESS. Tensiones para discutir: 1) Colisión de numeración. Witness v2 choca con BIP360 (P2MR, post-cuántico). El autor propone que la especificación que se despliegue primero se quede con el número, y que las demás tomen el siguiente disponible. Además difiere a los editores: dijo estar dispuesto a cambiarlo si se lo piden. 2) Empaquetado con P2TRv2 en el hilo 2749. sipa (agosto): agregar CISA al bundle alarga

el tiempo hasta que P2TRv2 esté disponible y reduce la adopción antes de Q-day más de lo que los incentivos de comisión la aumentan. AdamISZ (25/08): le parece algo «off» fusionarlos, y dice que el beneficio práctico en comisiones está sobredimensionado. conduition (05/08): «a natural vessel». fjahr (desde el 23/08): se declara «biased pro CISA» y sostiene que las características pueden coexistir y adoptarse por separado. 3) Privacidad. Un input que agrega se distingue de uno que no, y una transacción con agregación revela propiedad común. La exclusión explícita por input mitiga el problema, pero solo si se usa.

5. DropKick: un protocolo mínimo de commit/reveal para rescatar monedas ante la amenaza cuántica

DropKick: a minimal commit/reveal protocol for post-quantum coin rescue — fuente

Si las firmas de curva elíptica dejaran de ser seguras por una computadora cuántica, las monedas en direcciones con la clave pública ya expuesta las podría robar quien tenga esa máquina. DropKick propone un rescate en dos pasos. Primero el dueño publica en la cadena un compromiso (un hash) que ata su gasto a una clave poscuántica. Después de esperar un plazo fijo de bloques, revela el gasto real. Como el atacante no puede saber qué hay detrás del hash, no tiene cómo adelantarse.

Según la fuente, no inventa nada: usa el esquema original de FawkesCoin y elige, para cada subproblema, la mitigación más directa. También ordena los trabajos anteriores y muestra en qué falla cada uno.

Importa ahora porque la migración poscuántica es un tema abierto y este es un camino concreto para las monedas que nadie migró. Para BitDevs Córdoba, según la ficha, el eje poscuántico fue dominante en el #12 (3 de los 5 temas propios) y el autor ya figura como voz citada en el hilo de Disabling EC que trataron. La ficha también registra que 5 de 29 sedes lo mencionan.

Mecanismo, según la fuente: el usuario genera una clave de firma poscuántica y publica en la cadena C = H(W), donde W es el testigo. Espera D bloques (el ejemplo de la fuente es 1440). Luego publica una transacción de revelación con el testigo, la firma poscuántica, las firmas legacy y una prueba de apertura. Los nodos validan las reglas legacy, autentican el compromiso con la prueba (con antigüedad de al menos D bloques), verifican la firma poscuántica y comprueban que F(W) coincida con el scriptPubKey de cada UTXO. F es cualquier función de un solo sentido resistente a lo cuántico.

Mitigaciones que elige: - Comisión de salvamento pagada desde la moneda rescatada, más agregación de compromisos en árbol de Merkle. - Verificación simple de compromisos (SCV), con pruebas de tamaño logarítmico, sin que los nodos indexen compromisos. - Revelación demorada más incentivo de comisiones contra la censura de mineros. - Gastos atómicos de varios UTXO que comparten testigo. - Certificación de clave: se certifica una clave poscuántica una vez por commit/reveal y esa clave firma las transacciones.

Contra LifeBoat (Dryja): DropKick tiene seguridad contra censura game-theoretic y no estricta, rescate en cientos o miles de bloques y no en unos 6, compromisos que pueden ir ocultos y sin indexado obligatorio.

Límites que declara: demoras largas, una mayoría persistente de hashpower podría censurar, pruebas de apertura grandes, firmas poscuánticas voluminosas y agregadores expuestos a spam.

Para discutir: si esa garantía de censura alcanza frente a la de LifeBoat.

6. Escalar la red Lightning con reconciliación de conjuntos aplicada

Scaling the Lightning Network with Practical Set Reconciliation — fuente

Los nodos Lightning conocen la topología de la red (canales y nodos) por gossip: cada uno reenvía a sus pares lo que recibe. Un paper del 16 de agosto de 2026 midió ese mecanismo y concluye que es estructuralmente ineficiente. Propone reemplazarlo por reconciliación de conjuntos, la idea que Erlay usa para las transacciones en la capa base: en vez de mandar todo, dos nodos intercambian solo la diferencia entre lo que cada uno tiene. En simulaciones sobre Core Lightning con capturas reales de gossip, la sincronización baja de horas a unos pocos minutos. Por qué importa ahora: es un trabajo reciente y no aparece en el relevamiento de temarios de otras sedes, según la ficha. Por qué a BitDevs Córdoba: hay un debate concreto para la sala. La especificación de Lightning (BOLT 7) ya trae primitivas de sincronización, y el paper dice que igual el resultado es ineficiente. ¿Falla el diseño o falla cómo se usa? Eso es para discutir, no para dar por resuelto.

> Problema: el gossip de Lightning es por inundación. Los autores dicen que sus mediciones lo muestran como "fundamentalmente ineficiente". Propuesta: compartir anuncios de canal, actualizaciones y anuncios de nodo con protocolos de reconciliación de conjuntos. Evalúan los enfoques existentes bajo condiciones de red realistas y proponen ADAPTIVEIBLT, un protocolo IBLT (Invertible Bloom Lookup Table) adaptativo, con una mejora de decodificación parcial. Evaluación: simulación de la reconciliación en Core Lightning con snapshots reales de gossip. El resultado que reportan es pasar de horas a pocos minutos. Punto de discusión: BOLT 7 ya define query_channel_range y reply_channel_range con checksums, query_short_channel_ids y gossip_timestamp_filter. También exige que un nodo no envíe gossip que no generó él mismo hasta recibir gossip_timestamp_filter. Es decir, el paper pisa mecanismo ya especificado. Esa lectura sale de la nota de la ficha, que cita BOLT 7. No la contrasté yo contra la spec. Autoría: la ficha dice que Trachtenberg es coautor del trabajo original de reconciliación de conjuntos que usa Erlay. No lo verifiqué. Tampoco leí el texto completo del paper, así que no puedo describir ADAPTIVEIBLT más allá del abstract. Quedan sin cubrir el tamaño de los conjuntos, los parámetros y la metodología de medición.