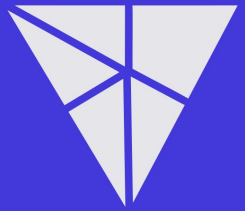
Security Alliance Wargames

Preparing teams for incident response

Isaac Patka

Security Alliance (SEAL) | Shield3



“Securing the future of crypto”

Outline

- What are Wargames and why do them?
- How can you work with us and use our resources?
- Best practices from working with major protocols
- Walkthrough of open source toolkit

What are Wargames?

SEAL Wargames are cybersecurity exercises for web3

- Designed to prepare protocol teams for high-pressure situations.
- Test & improve both the social and technical resilience of teams.
- ~~Incite panic~~ | Practice emergency response

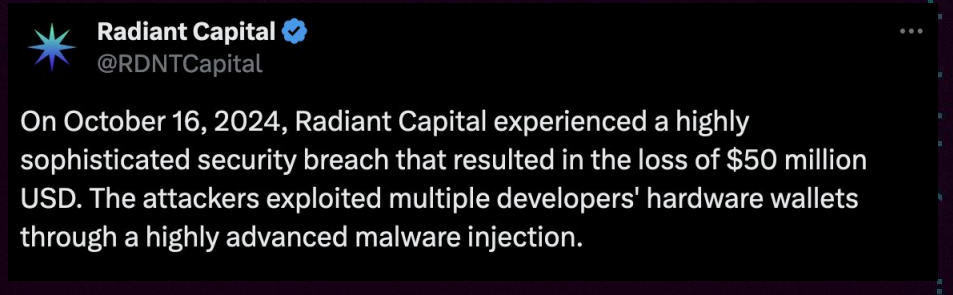
Phase 1 - Open-source intelligence gathering

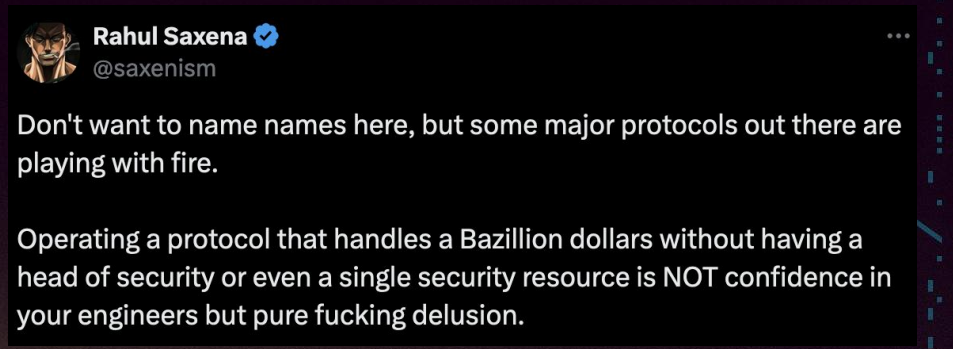
Phase 2 - Tabletop exercise to identify team's monitoring & response process & weaknesses

Phase 3 - Live attack simulation on a forked network requiring real-time coordination from the team

Why Wargames

Recent hacks have been operational failures, not smart contract bugs





Wargames are a cross-functional exercise

🖥️ **Core Devs**
- Subject matter experts
- Understand the attack
- Prepare & simulate responses

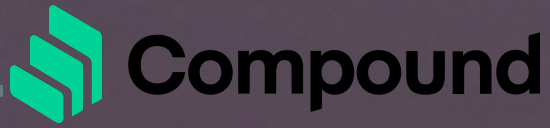
📋 **Auditors**
- Support core devs to understand incident
- Analyze implications of response

👼 **Guardians**
- Sign transactions to fix/ pause the protocol

📰 **Communications**
- Share information with the public
- Maintain incident response records

⚖️ **Legal**
- Analyze legal implications of response
- Sign-off on whitehat engagements

Projects We've Worked With



Wargames provide hands-on experience with incident response

“Under pressure, you don't rise to the occasion, you sink to the level of your training.”

(Quote attribution: either the ancient Greek poet Archilochus or Navy SEALs. Internet not sure)

Custom build. Realistic environment

Protocol		Network Fork	Monitoring	Bots	Exploit	Misc.
	Compound	Anvil/ Blockscout	Forta	Foundry scripts	Foundry scripts	
	Yearn	Anvil/ Blockscout	Silverback	Silverback	Silverback	Forked internal tools & UIs
	Aave	Tenderly	Silverback	Silverback	Silverback	Forked UI
	Superchain	Conduit/ Sepolia	Chain-mon / Hexagate	Chain-mon fork	Foundry scripts	SEAL 911
	Uniswap	Tenderly	Chain-mon fork	Chain-mon fork	Chain-mon fork	

OS Drill Template Toolkit: https://github.com/security-alliance/drill-template

Scenarios for Wargame Exercises Have Included:

- External dependency failures
- Faulty contract upgrades
- Oracle manipulation
- Malicious governance proposals

(Vulnerability mapping - social, technical, legal, economic)

Best Practices

Maintain multiple channels to reach Guardians

- Pre-sign critical transactions to pause the protocol to minimize coordination time
- If signing is needed, have multiple ways of reaching multisig guardians

Best Practices

Isolate risks from protocol dependencies

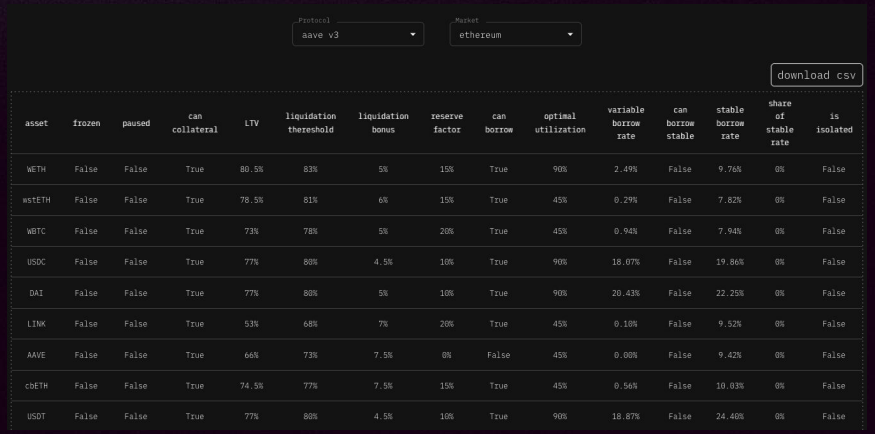
- Yearn prepares strategy specific risk scores
- Risks are contained to specific strategies and cannot affect funds in other strategies
- *Emergency Cards* are prepared before deploying a strategy containing key subject matter experts and incident response steps

https://seafood.yearn.watch/risk

Best Practices

Isolate risks from protocol dependencies

- Aave isolates risks from different collateral assets by restricting which assets can be used for cross-collateral borrowing
- Individual collateral assets can be paused or frozen



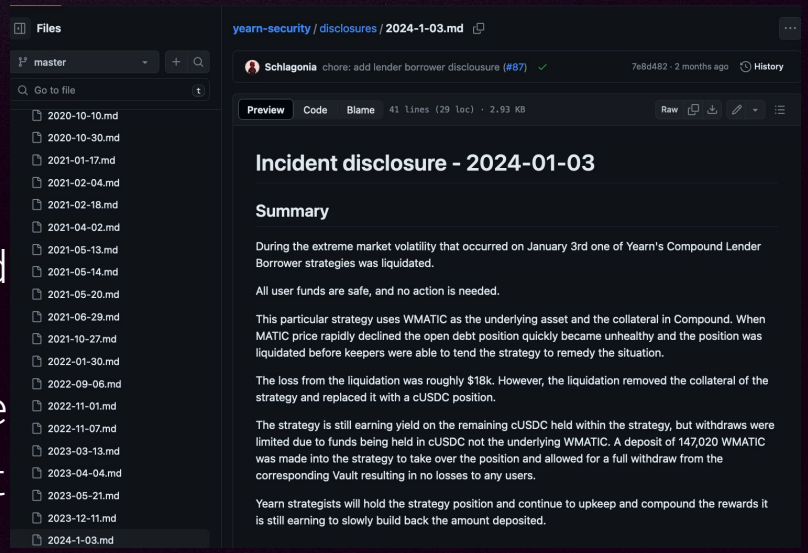
asset	frozen	paused	can collateral	LTV	liquidation thereshold	liquidation bonus	reserve factor	can borrow	optimal utilization	variable borrow rate	can borrow stable	stable borrow rate	share of stable rate	is isolated
WETH	False	False	True	80.5%	83%	5%	15%	True	90%	2.49%	False	9.76%	0%	False
wstETH	False	False	True	78.5%	81%	6%	15%	True	45%	0.29%	False	7.82%	0%	False
WBTC	False	False	True	73%	78%	5%	20%	True	45%	0.94%	False	7.94%	0%	False
USDC	False	False	True	77%	80%	4.5%	10%	True	90%	18.07%	False	19.86%	0%	False
DAI	False	False	True	77%	80%	5%	10%	True	90%	20.43%	False	22.25%	0%	False
LINK	False	False	True	53%	68%	7%	20%	True	45%	0.10%	False	9.52%	0%	False
AAVE	False	False	True	66%	73%	7.5%	0%	False	45%	0.00%	False	9.42%	0%	False
cbETH	False	False	True	74.5%	77%	7.5%	15%	True	45%	0.56%	False	10.03%	0%	False
USDT	False	False	True	77%	80%	4.5%	10%	True	90%	18.87%	False	24.40%	0%	False

https://www.config.fyi/

Best Practices

Conduct & publish incident post-mortems

- Reflect on the *monitoring* which detected the issue and the response by the team
- Integrate learnings from the post-mortem into an incident response playbook

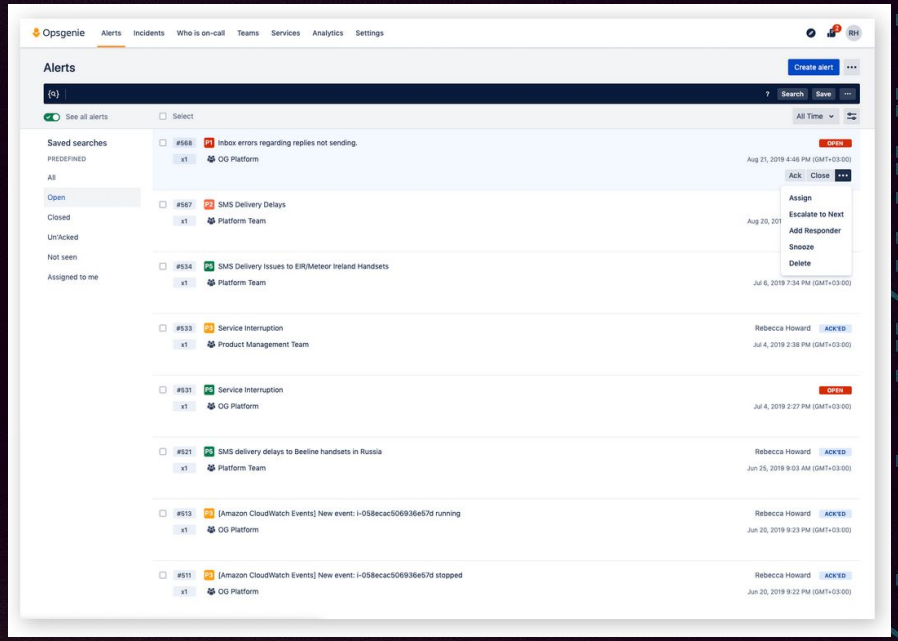


https://github.com/yearn/yearn-security/tree/master/disclosures

Best Practices

Mirror monitoring infrastructure on testnets

- Test alerts regularly and run drills to ensure alerts work
- Include playbooks in the alert to help the responder identify false alarms, and gather a war room if needed



Open Source Template for Wargames

A Foundry & Hardhat setup for developing & testing scenarios on a local fork

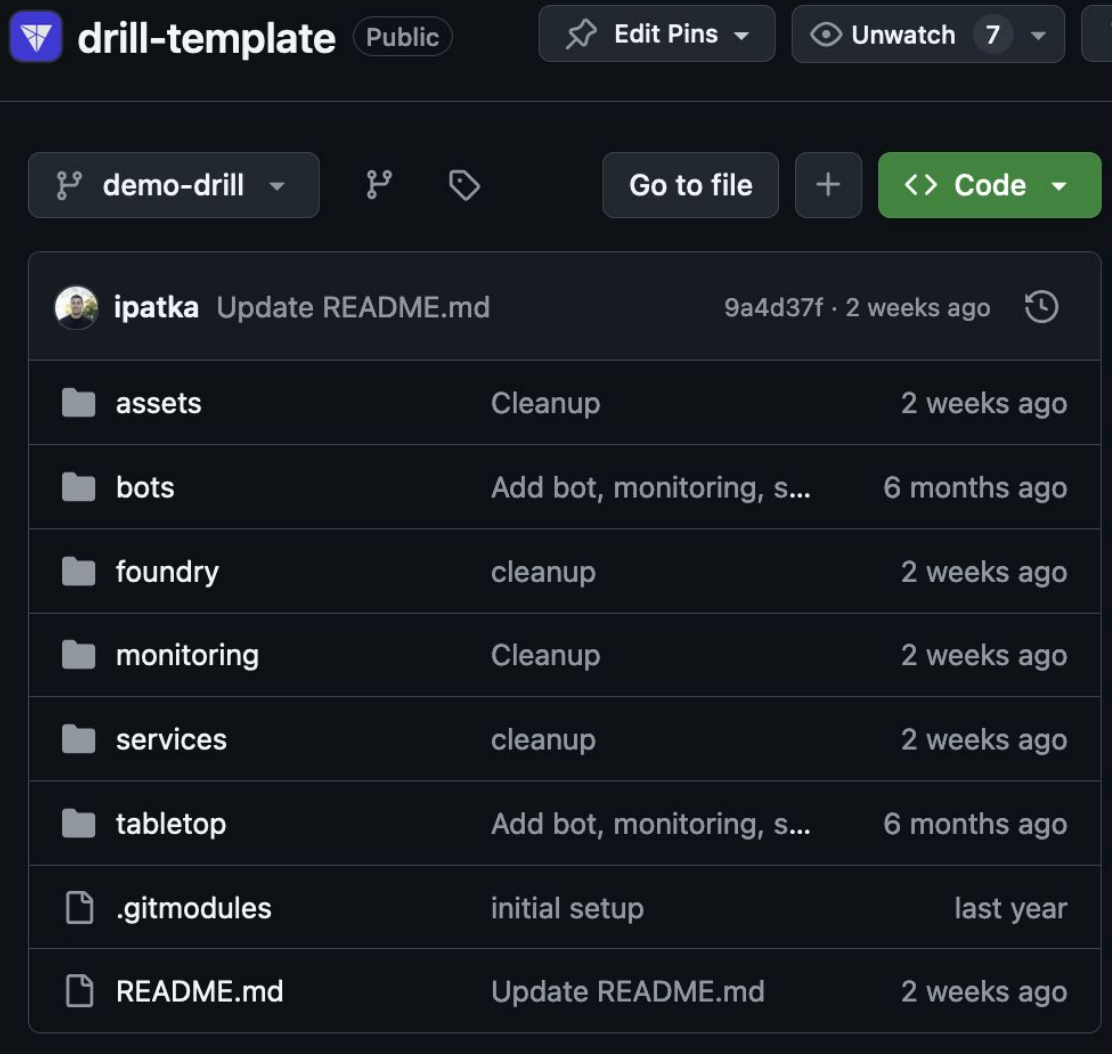
Configurations for running a live fork on Tenderly

A template for a tabletop exercise

A template typescript bot service (inspired by Optimism)

A template monitoring bot service with connections to Prometheus, Grafana, and OpsGenie (inspired by Optimism)

https://github.com/security-alliance/drill-template



Resources

Tabletop Script

[Protocol Name]

[Date]

Drill Goals

In this first exercise, we will discuss scenarios in which the core teams needs to detect, diagnose, and respond to a variety of issues. We will gather the core decision-makers together to decide what actions should be taken, and what should be communicated. This exercise will be discussion-based and will not involve any real actions that need to be taken.

The goal of the exercise is to prepare for attacks and dependency failures in order to test both social and technical resiliency, harden internal procedures for deployment & recovery, and develop a training program for team members.

After going through this drill the team will be able to understand:

- What are the key dependencies of the protocol and what happens if they fail?
- Is there sufficient monitoring infra to detect & respond to failures?
- Is there an understanding of who has access to admin keys and how & when they are accessed to respond to threats?
- Do essential team members have backup people in place who can respond if they are unavailable?
- Does the team know who to reach out to in other protocols if a failure is detected?

Stakeholders

All team members are welcome in the exercise with duties related to:

- Notification & reporting infrastructure for application state & dependencies
- Smart contract developers
- FE development & interface hosting
- Communications

Tabletop Script

Since the incident response will change as upgrades are made to the protocol, let's discuss these scenarios as if they happened today.

Scenario 1:

Phase 1 - discovery of the issue

What monitoring infrastructure alerted the team to the issue?

Phase 1a - Gather stakeholders

Did the team know who to ask for help in this situation?

Phase 2 - Analyze the issue and develop a response plan

Did the team have an existing playbook for what to do?

Phase 3 - Implement the response plan

Who took action and what did they do?

Phase 4 - Reflection

What could have gone better?

Resources

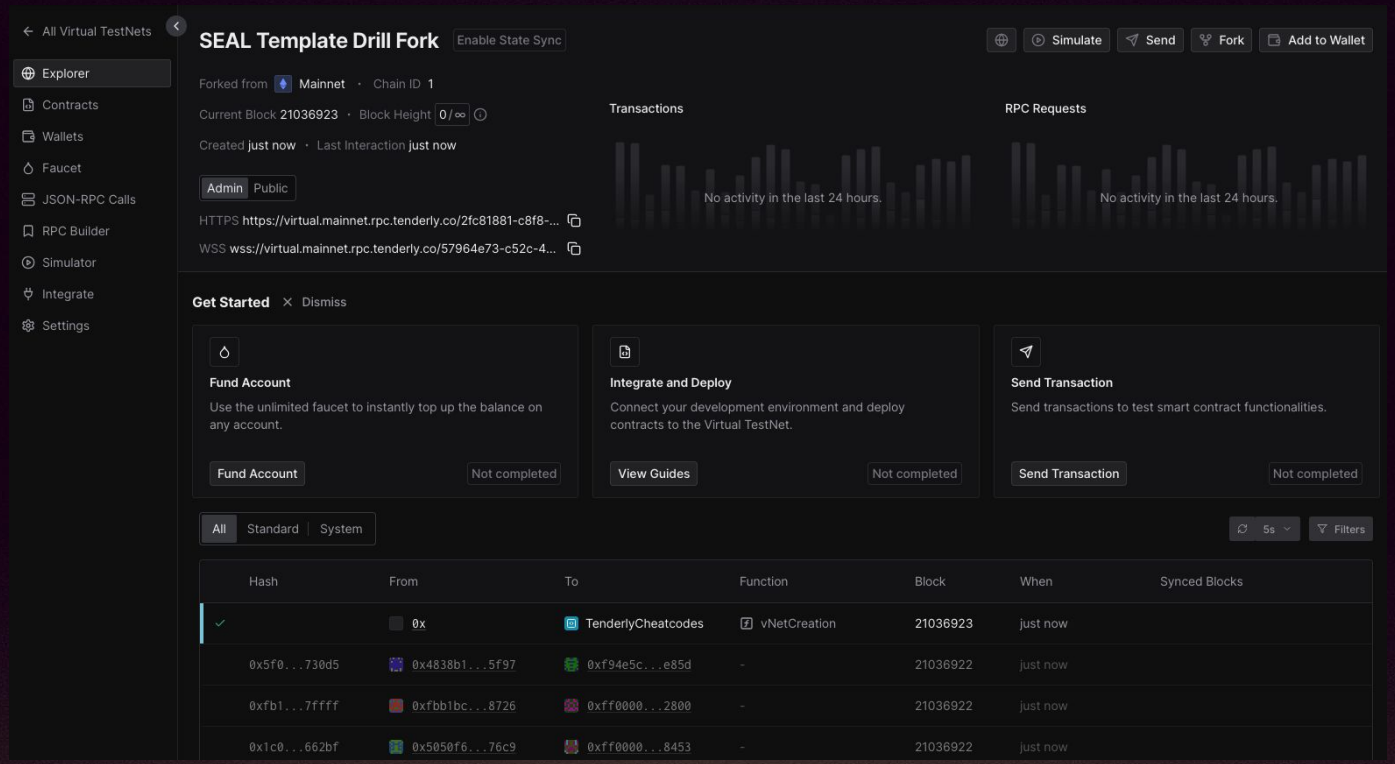
Live Drill

Live Drill Setup Steps

1. Create network fork
2. Run bots & monitoring
3. Brief team
4. Team triages incident
5. Team prepares response & comms
6. Team executes incident response
7. Postmortem

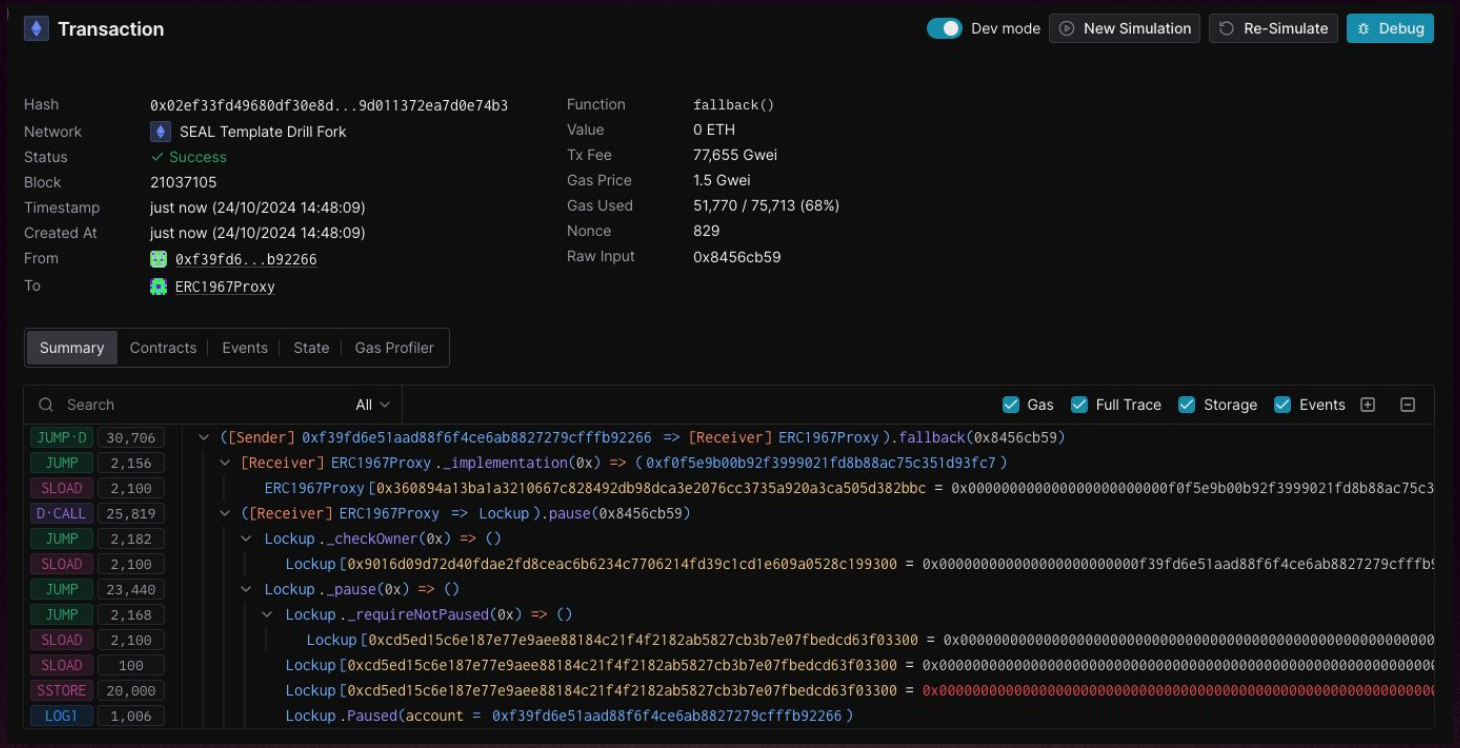
Run The Live Drill

Live Fork Explorer & RPC



Bots & Monitoring

Incident Response



Resources: ‘LET THE GAMES BEGIN’

- Join live Wargames event (previously November 10 between DSS and Devcon in Bangkok with Tenderly)
- SEAL Wargames Drill Scenario Template (OS toolkit): https://github.com/security-alliance/drill-template
- Join the waitlist for a SEAL sponsored drill: https://securityalliance.org/
- Paid track for projects on an accelerated timeline: https://www.shield3.com/
- Chat with Isaac re. Running your own wargames

Thank you!

Isaac Patka

Security Alliance (SEAL) | Shield3

isaac@shield3.com

@isaacpatka on Telegram