Analysis of the Security Design, Engineering, and Implementation of the SecureDNA System

Alan T. Sherman,
Cyber Defense Lab
University of Maryland, Baltimore County (UMBC)

February 26, 2026
Network and Distributed System Security Symposium (NDSS)

Joint work with Jeremy J. Romanik Romano, Enis Golaszewski,
Edward Zieglar, Jonathan D. Fuchs, William E. Byrd

UMBC

UAB

SecureDNA Screens Synthesis Requests against Hazards

SecureDNA

SecureDNA is the most promising screening system

Main Results

- Formal-Methods analysis of SCEP and query protocols

SCEP aims to provide mutual authentication

- Two structural weaknesses in SecureDNA protocols

1. *Custom SCEP protocol provides only one-way authentication*
   => rate-limiting attack by corrupt *K* or *H*

Keyserver *K*

2. *Inadequate bindings of responses from H*
   => *replay/swapping attack could change H's responses*

   If S reconnects over same TLS session (which the implementation disallows)

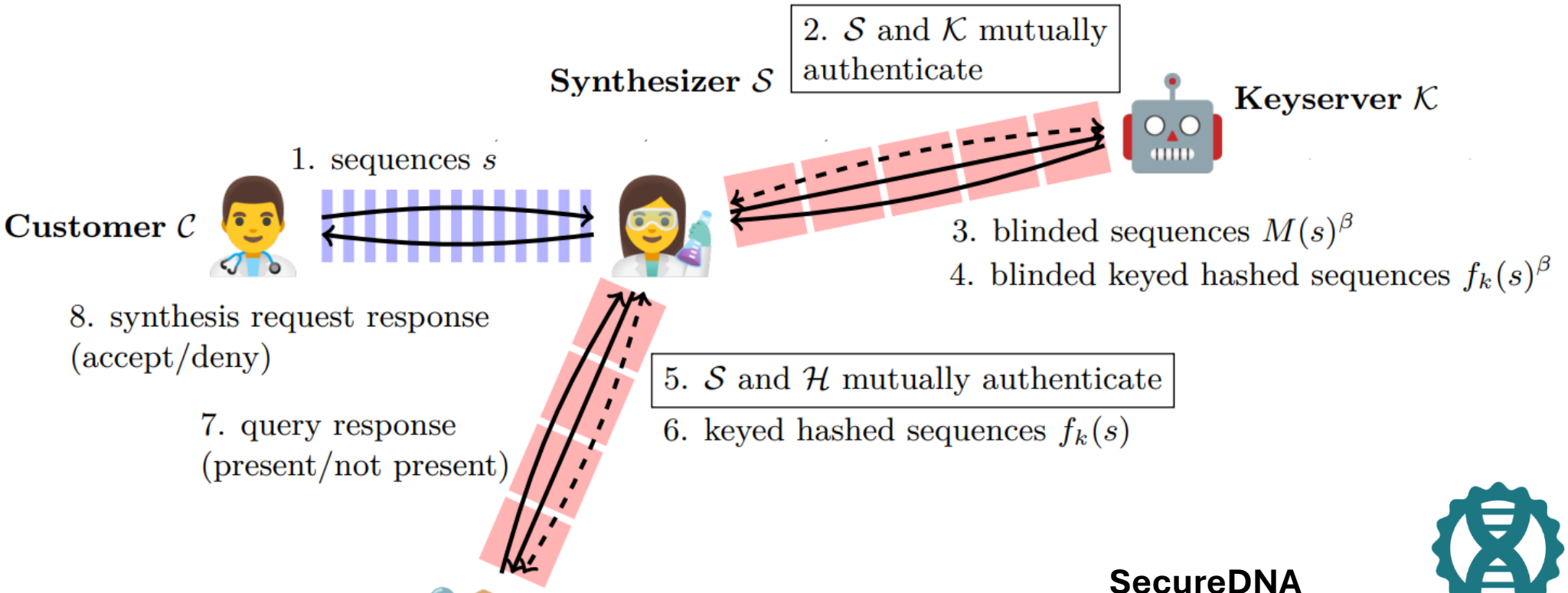
Hashed Database *H*

- Improved version SCEP+

  SecureDNA implemented our recommended fix

It would be stronger security engineering to eliminate these weaknesses—other attacks might be possible

SecureDNA's Basic Query



SecureDNA

Launched in late 2022.
Enables honest synthesizers to screen requests.
Goals: keep request and hazards secret.
Version 1.0.8 has 64,000 lines of Rust code.

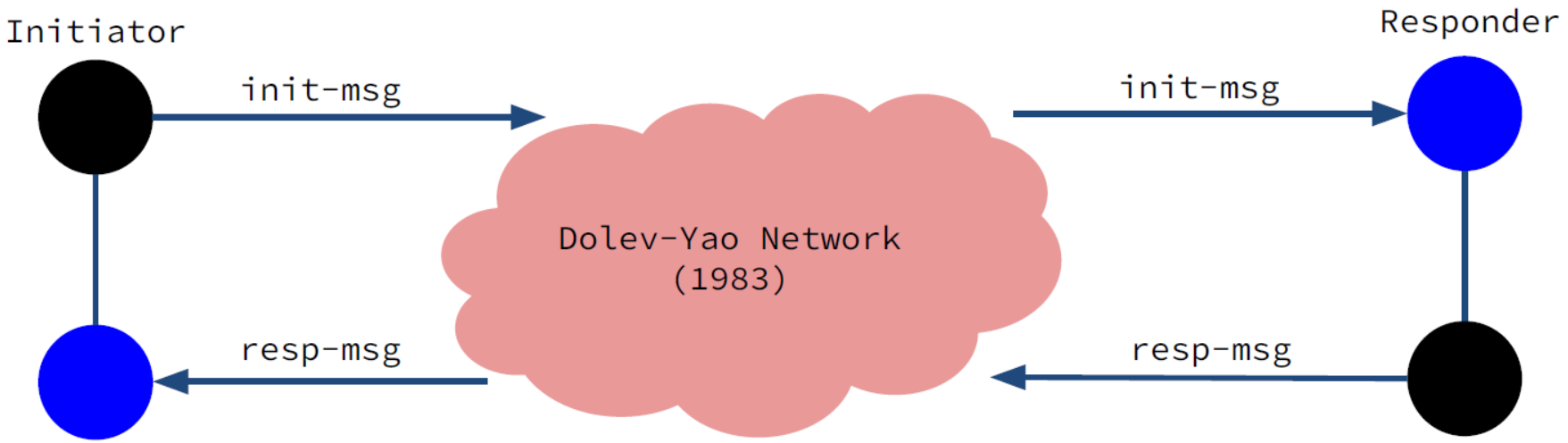
Security Goals

SG1	Keep database secret
SG2	Keep synthesis request secret
SG3	Return correct answers for basic requests
SG4	Return correct answers for exemption requests

Non-Security Goals: Speed, accuracy, easy to use, adoption

Adversarial Model

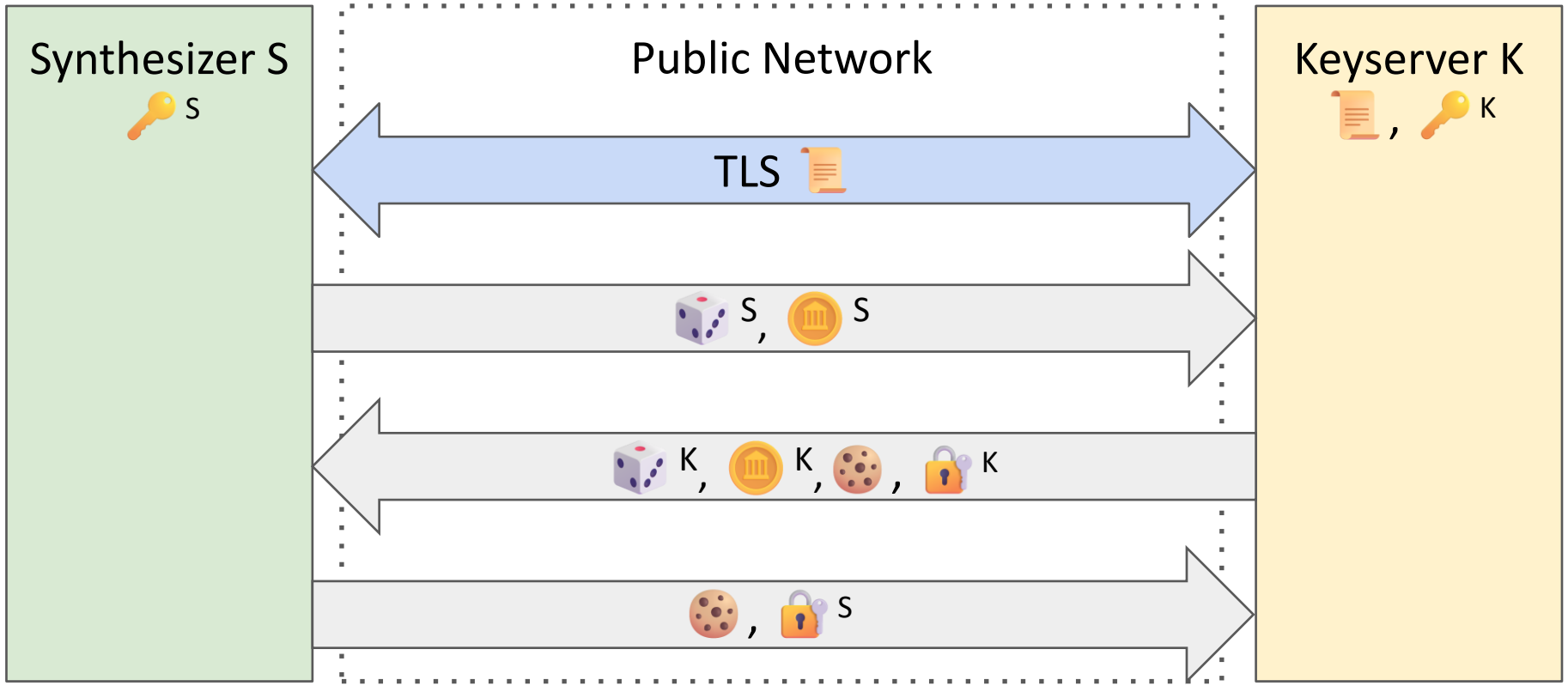
- Dolev-Yao (DY) network intruder
  - Full control of messages on network
  - Does not have to follow protocol
  - Can manipulate unbounded number of protocol sessions
  - May be a legitimate communicant
  - Cannot break cryptographic primitives



Challenges

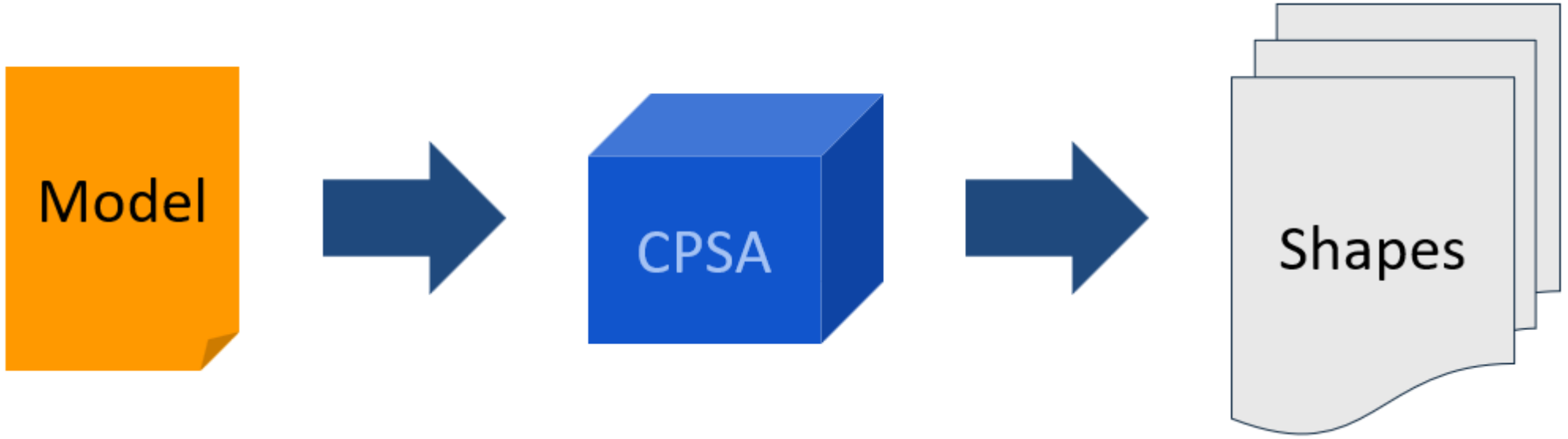
- Documentation describes protocols inadequately
  => We discerned protocols from code review
- Protocol analysis is inherently complex
  => We used CPSA
- SecureDNA was slow to engage with us

SCEP Authentication Protocol



- Keyserver Certificate
- K Keyserver Token
- S Synthesizer Token
- K Keyserver Nonce
- S Synthesizer Nonce
- K Keyserver Signing Key
- S Synthesizer Signing Key
- Cookie (used for subsequent authentications)
- K Keyserver Verifier: **enc**((S, K, K), K)
- S Synthesizer Verifier: **enc**((S, K, K), S)

Formal-Methods Analysis using CPSA

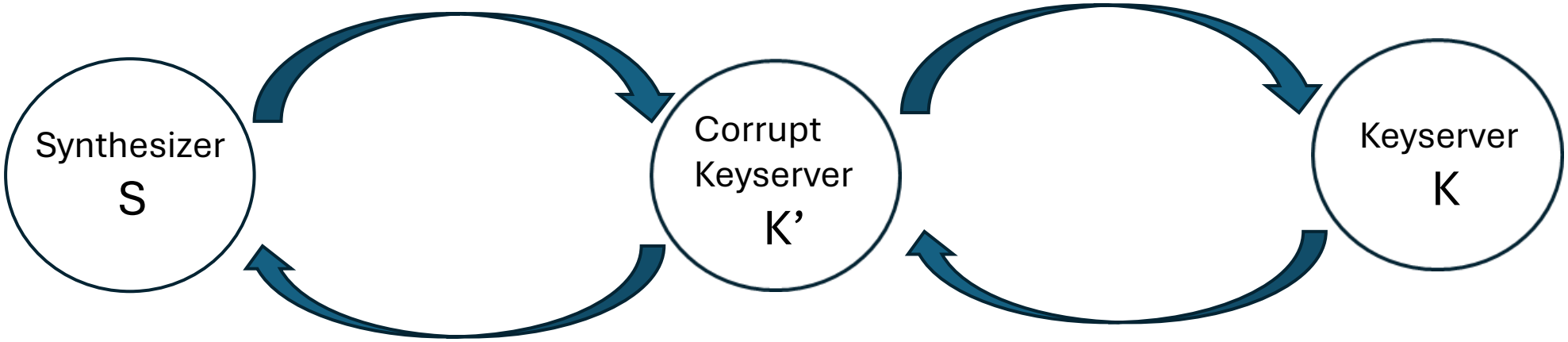


Cryptographic Protocol Shapes Analyzer

Developed by MITRE, funded by NSA

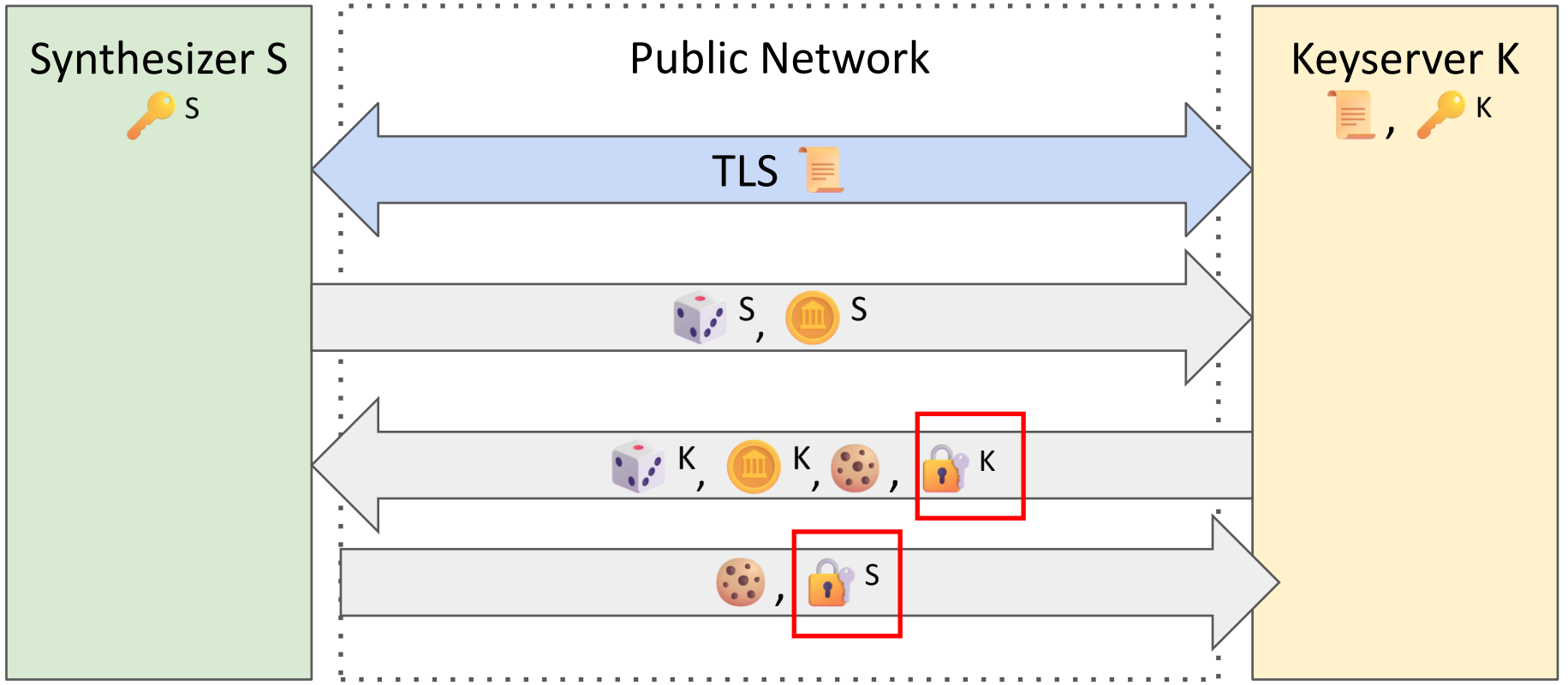
Completeness, M. Liskov (2011): If CPSA terminates, it has explored all essentially different possible executions given initial assumptions.

Attack 1: Rate Limiting by Corrupt Keyserver



K' masquerades as *S* to *K*.
Similar to Lowe's attack on Needham-Schroeder.
Also works substituting *H* for *K*.

SCEP+ Authentication Protocol



Improve bindings

- Keyserver Certificate
- K Keyserver Token
- S Synthesizer Token
- K Keyserver Nonce
- S Synthesizer Nonce

- K Keyserver Signing Key
- S Synthesizer Signing Key
- Cookie (used for subsequent authentications)
- K Keyserver Verifier: **enc**((S, K, K, S,), K)
- S Synthesizer Verifier: **enc**((S, K, K, S,), S)

Summary of Formal-Methods Analysis

Security Goal	**SCEP**	**SCEP+**
Confidential (*S*, Cookie)	✓	✓
Confidential (*K*, Cookie)	✗	✓
Agreement (*S*, *K*, [*S*, *K*, Nonce(*S*), Nonce(*K*), Cookie])	✓	✓
Agreement (*K*, *S*, [*S*, *K*, Nonce(*S*), Nonce(*K*), Cookie])	✗	✓

S = Server, *K* = Keyserver

Findings

- SCEP achieves only one-way authentication
- Inadequate bindings (e.g., responses from *H*)
- SecureDNA depends critically on auditing to mitigate rate-limiting attacks and other malicious behaviors.
- Blinding protects secrecy of requests

Recommendations

- Improve SCEP (replace with mTLS or add proper bindings to SCEP)

  => SecureDNA implemented our SCEP+
- Strengthen cryptographic bindings in all protocols
- Perform security review of TLS implementation and integration
- Take greater care in system design (e.g., store keys in TPMs)

Conclusion

- Secure systems need more than abstract cryptography.
- Better to use formal methods throughout design.