



# Compact Memory Implementations of the ML-DSA Post-Quantum Digital Signature Algorithm

R. D. Meneses, C. Teixeira, M. A. A. Henriques

16 de Setembro, 2024

# Sumário

01

Criptografia pós-quântica

02

Assinaturas digitais e ML-DSA

03

Otimizações de memória

04

Análise experimental

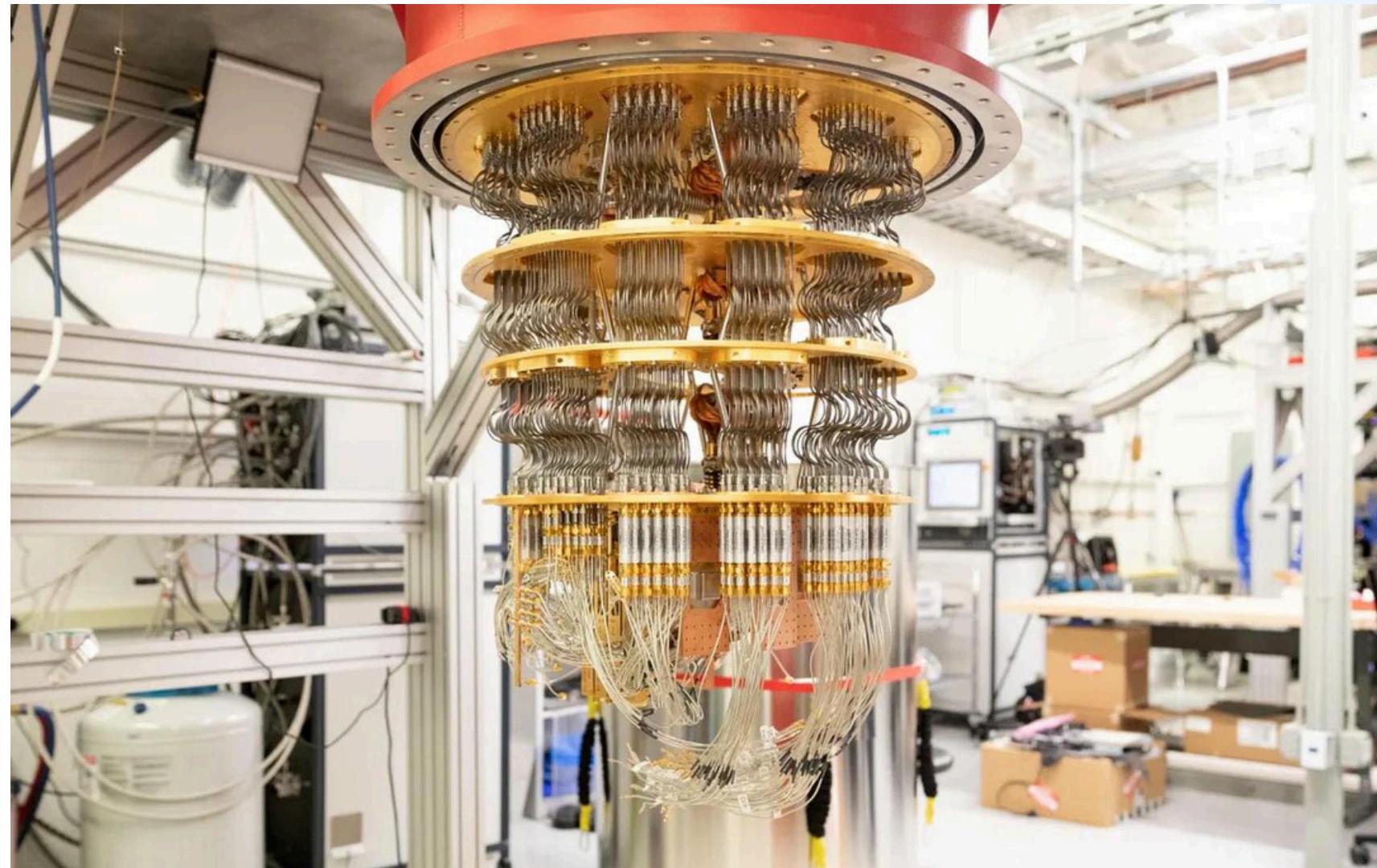
05

Resultados

06

Conclusão

# Criptografia pós-quântica



Computador quântico de 54-qubits da Google

## O que é criptografia pós-quântica?

Algoritmos criptográficos resistentes aos ataques de um computador quântico;

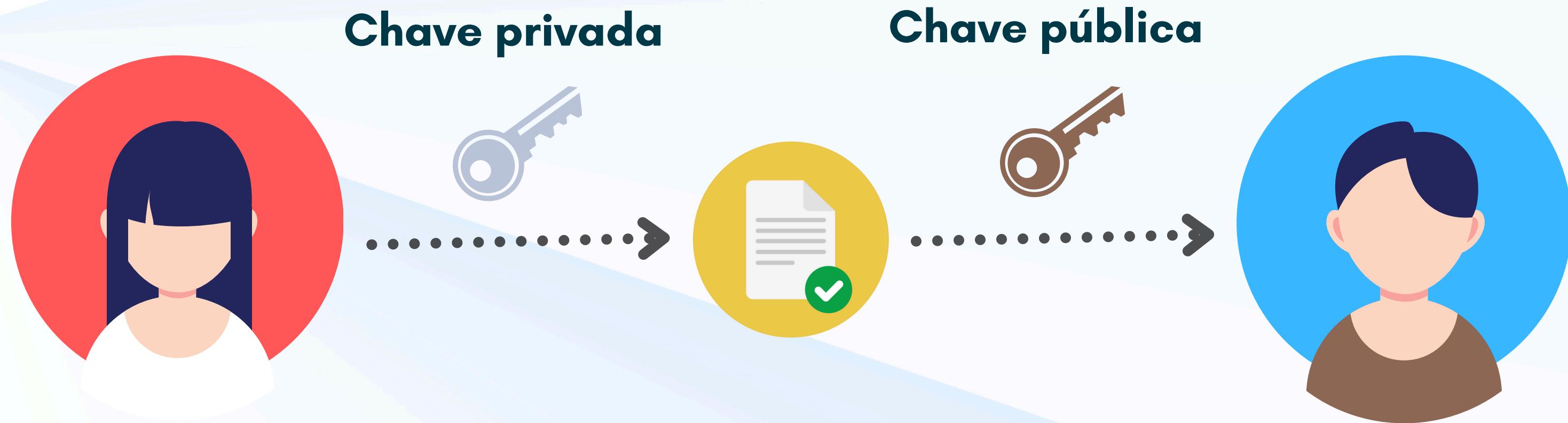
## NIST 2016 - 2024

Processo de padronização para algoritmos criptográficos pós-quânticos.

- ML-KEM (Kyber)
- **ML-DSA (Dilithium)**
- FN-DSA (Falcon)
- SLH-DSA (SPHINCS+)

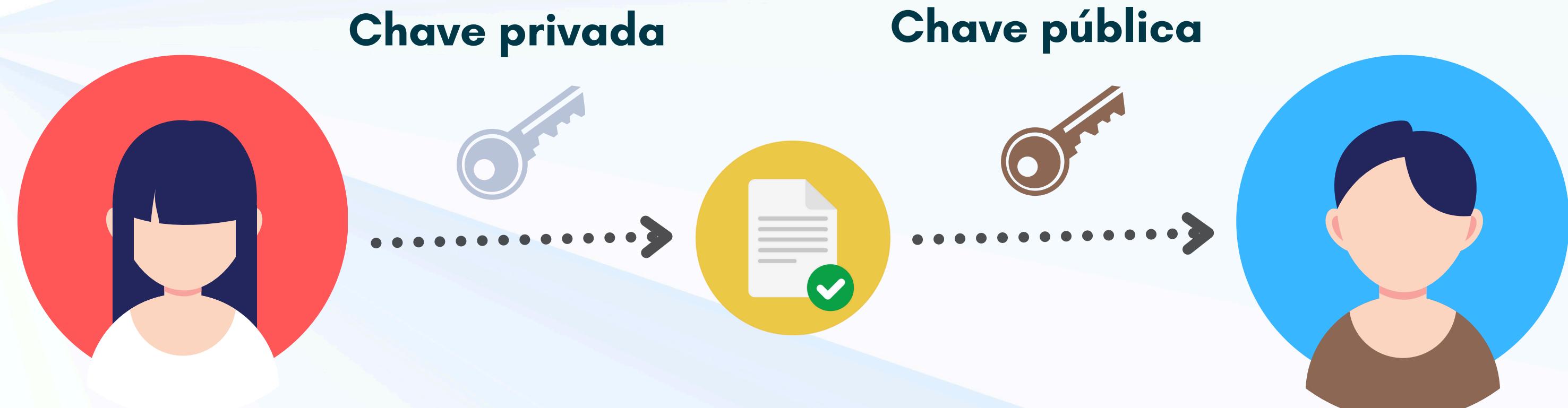
# Assinaturas digitais

- Buscam garantir a autenticidade e integridade de mensagens e documentos digitais.



# Assinaturas digitais

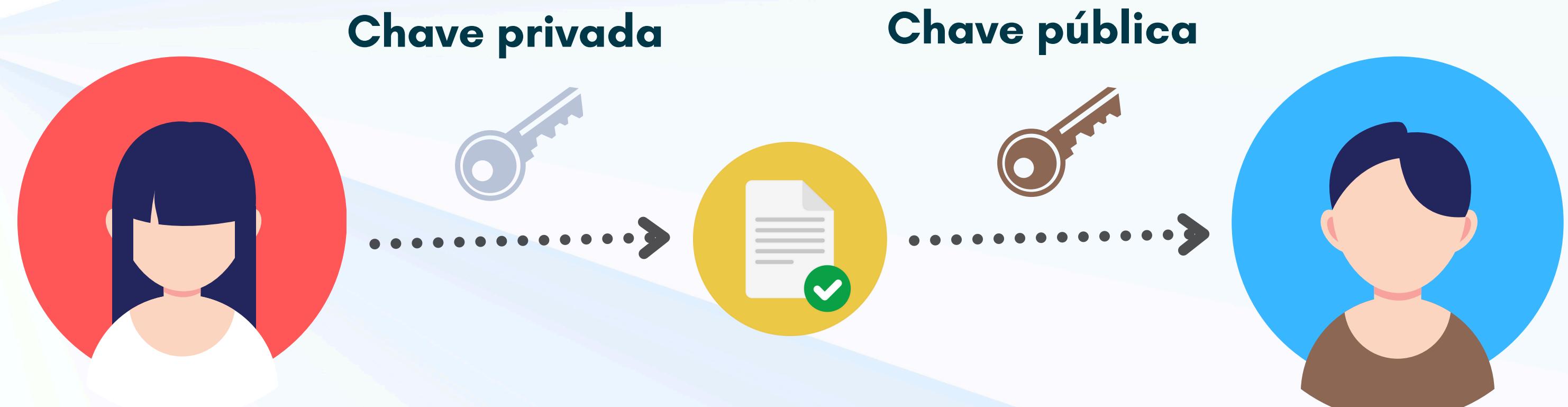
- Buscam garantir a autenticidade e integridade de mensagens e documentos digitais.



1. Geração de chaves (KeyGen);
2. Assinatura (Sign);

# Assinaturas digitais

- Buscam garantir a autenticidade e integridade de mensagens e documentos digitais.

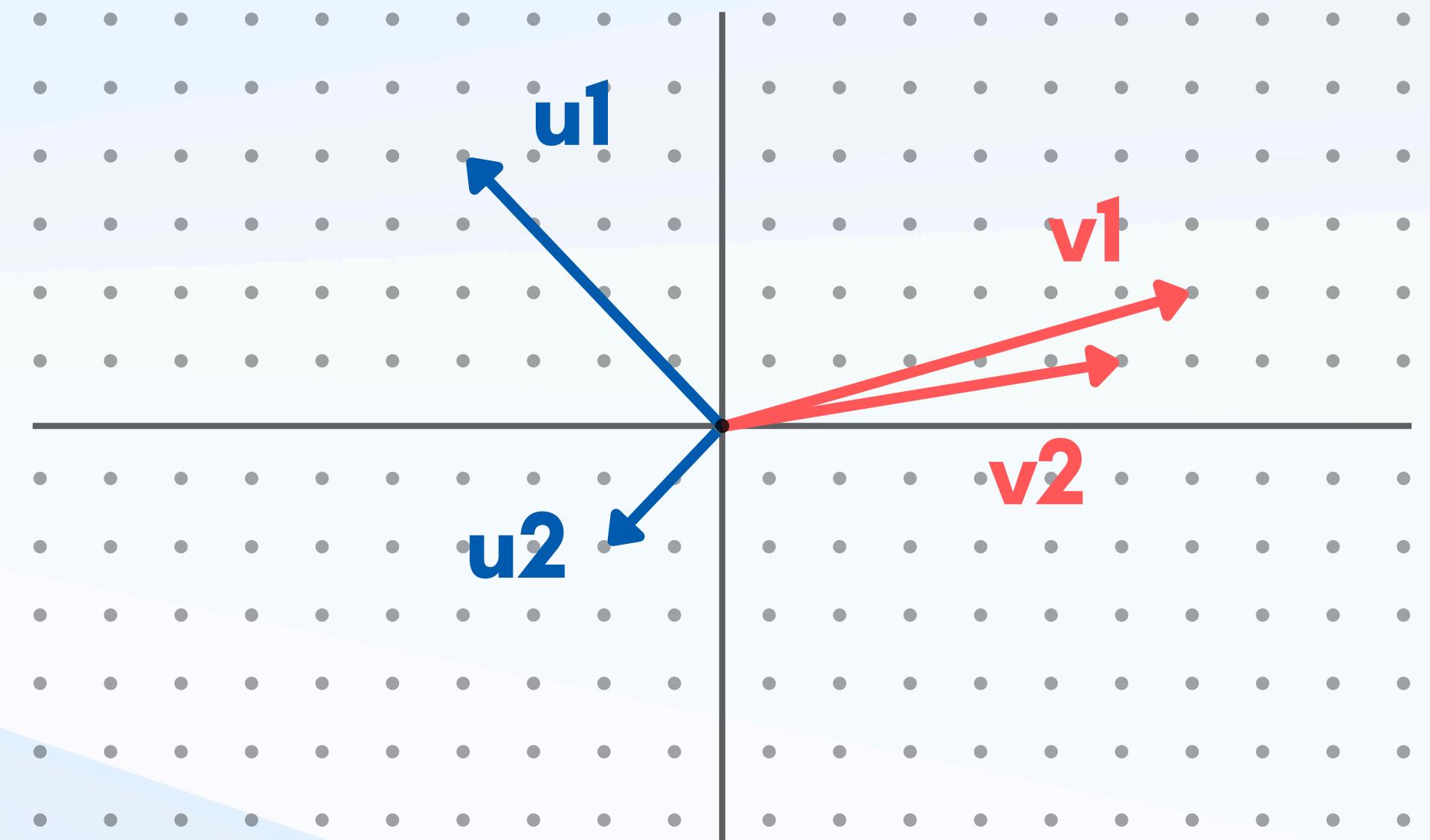


1. Geração de chaves (KeyGen);
2. Assinatura (Sign);
3. Verificação (Verify).

# Problemas sobre reticulados

## O que são reticulados?

- Estruturas algébricas que consistem em um conjunto  $S$  de pontos em um espaço  $n$ -dimensional.
- Os pontos são gerados a partir de uma base vetores contida em  $S$ .



## Problemas relevantes:

SVP, CVP, GapCVP, MSIS, **MLWE**...

# Funcionamento do ML-DSA



Chave pública =  $(A, t)$



Chave privada =  $(s1, s2)$

$$t = A \times s1 + s2$$

The diagram illustrates the formula for generating a polynomial  $t$ . It shows a vertical vector  $t$  on the left, followed by an equals sign. To its right is a matrix  $A$ , represented as a 4x4 grid of white boxes with a brown border. To the right of  $A$  is a multiplication sign ( $\times$ ). To the right of the multiplication sign are two vertical vectors:  $s1$  (represented by a 4x1 column of white boxes with a grey border) and  $s2$  (represented by a 4x1 column of white boxes with a grey border). Between  $s1$  and  $s2$  is a plus sign (+).

- Matriz de polinômios  $A$ ;
- Vetor de polinômios  $t$ ;
- Vetores secretos de polinômios  $s1$  e  $s2$ .

# Funcionamento do ML-DSA



Chave pública =  $(A, t)$



Chave privada =  $(s_1, s_2)$

$$t = A \times s_1 + s_2$$

The diagram illustrates the equation  $t = A \times s_1 + s_2$ . On the left, there is a vertical vector  $t$  composed of four white boxes. An equals sign follows. To the right is a matrix  $A$  with four columns and four rows, also composed of white boxes. A multiplication symbol ( $\times$ ) is placed between  $A$  and the first term of the sum. To the right of the multiplication is a plus sign ( $+$ ). Following the plus sign are two vertical vectors,  $s_1$  and  $s_2$ , each composed of four white boxes.

- Matriz de polinômios  $A$ ;
- Vetor de polinômios  $t$ ;
- Vetores secretos de polinômios  $s_1$  e  $s_2$ .

## Criptografia baseada em reticulados

A dificuldade consiste em, dados  $A$  e  $t$ , determinar os vetores  $s_1$  e  $s_2$ .

Cada entrada nos vetores ou matriz é um elemento no anel polinomial  $Z_q[X]/(X^{256} - 1)$ .

Este problema é chamado **MLWE (Module Learning with Errors)**.

# Otimizações de memória no ML-DSA

## **Problema:**

- Armazenamento de matrizes e vetores de polinômios exige muita memória.

# Otimizações de memória no ML-DSA

## Problema:

- Armazenamento de matrizes e vetores de polinômios exige muita memória.

## Método tradicional:

$$\begin{bmatrix} A & \text{---} \\ \begin{bmatrix} a_{11} & \cdots & a_{1l} \\ \vdots & \ddots & \vdots \\ a_{k1} & \cdots & a_{kl} \end{bmatrix} & \times & \begin{bmatrix} t & \text{---} \\ \begin{bmatrix} t_1 \\ \vdots \\ t_l \end{bmatrix} & = & \begin{bmatrix} v & \text{---} \\ \begin{bmatrix} v_1 \\ \vdots \\ v_k \end{bmatrix} \end{bmatrix} \end{bmatrix}$$

# Otimizações de memória no ML-DSA

## Problema:

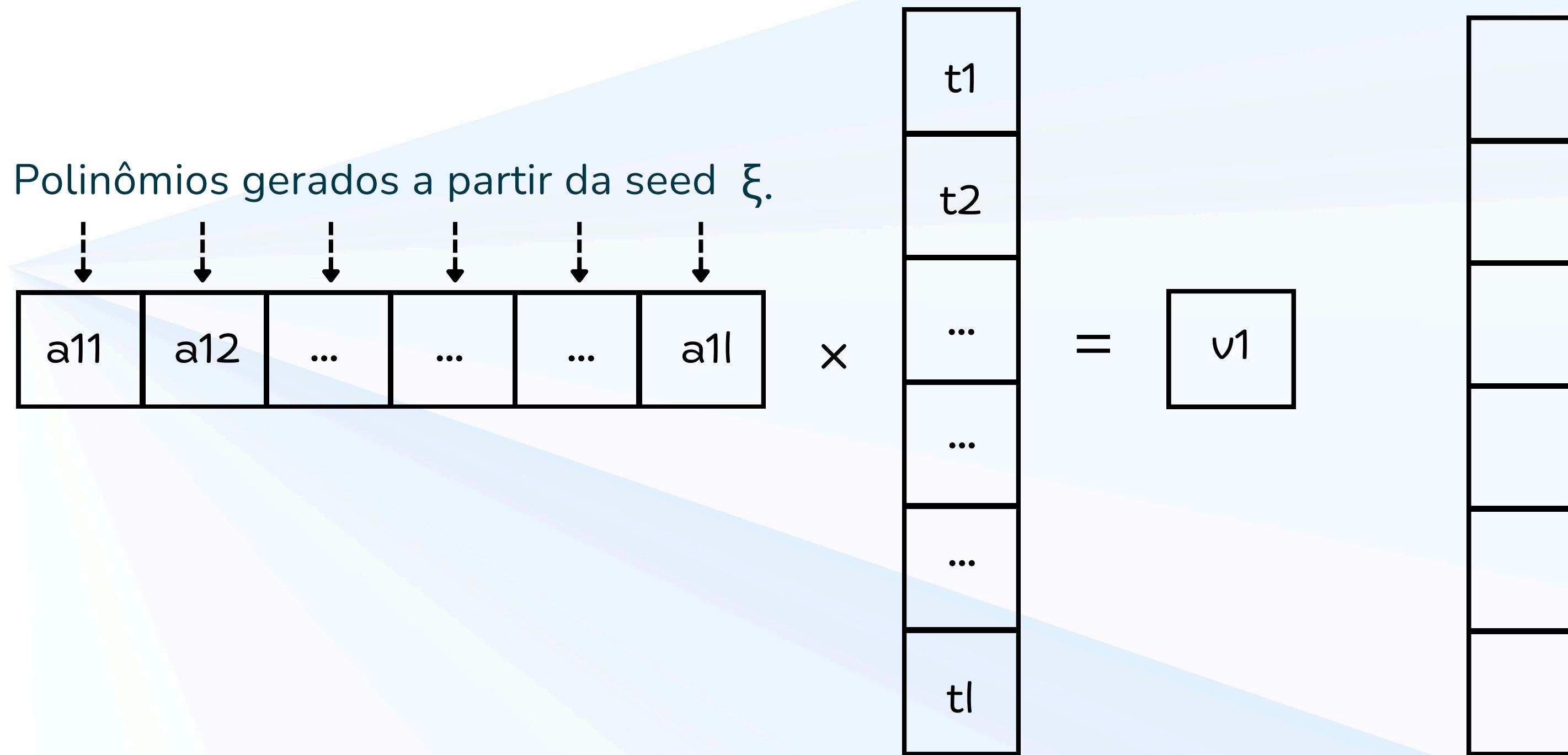
- Armazenamento de matrizes e vetores de polinômios exige muita memória.

## Método tradicional:

$$\begin{bmatrix} \mathbf{A} \\ \vdots \\ \mathbf{a}_{k1} \cdots \mathbf{a}_{kl} \end{bmatrix} \times \begin{bmatrix} \mathbf{t} \\ \vdots \\ \mathbf{t}_l \end{bmatrix} = \begin{bmatrix} \mathbf{v} \\ \vdots \\ \mathbf{v}_k \end{bmatrix}$$

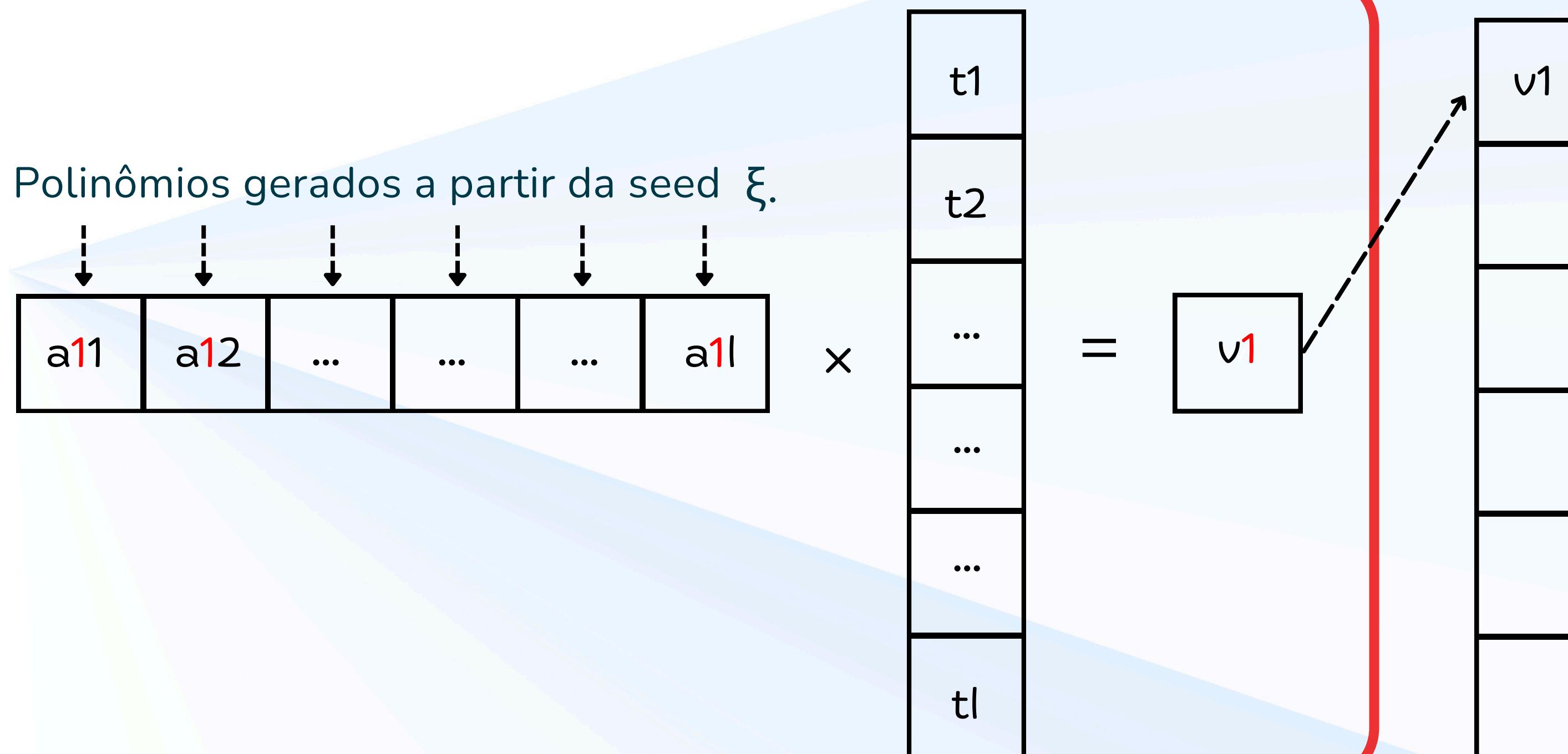
- A matriz pública **A** ocupa  $\{16, 30, 56\}$ -KiB de RAM, tornando inviável a implementação em dispositivos restritos.
- Parâmetros expandidos a partir de uma seed pseudoaleatória  $\xi$ .
- **Realizamos a geração dos parâmetros reutilizando uma mesma região de memória.**

# Geração via vetor de polinômios (ML-DSA-v)



# Geração via vetor de polinômios (ML-DSA-v)

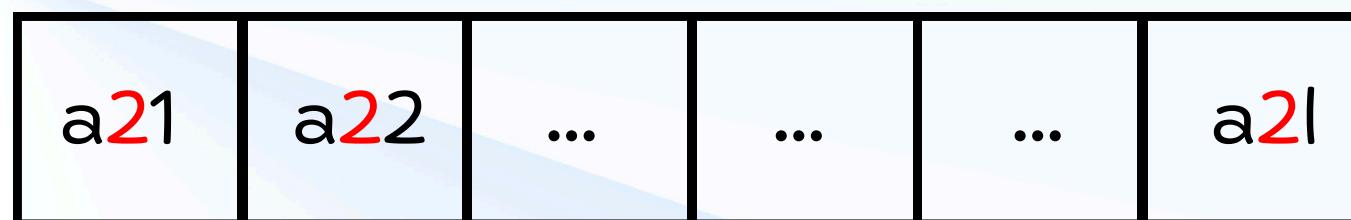
$i = 1$



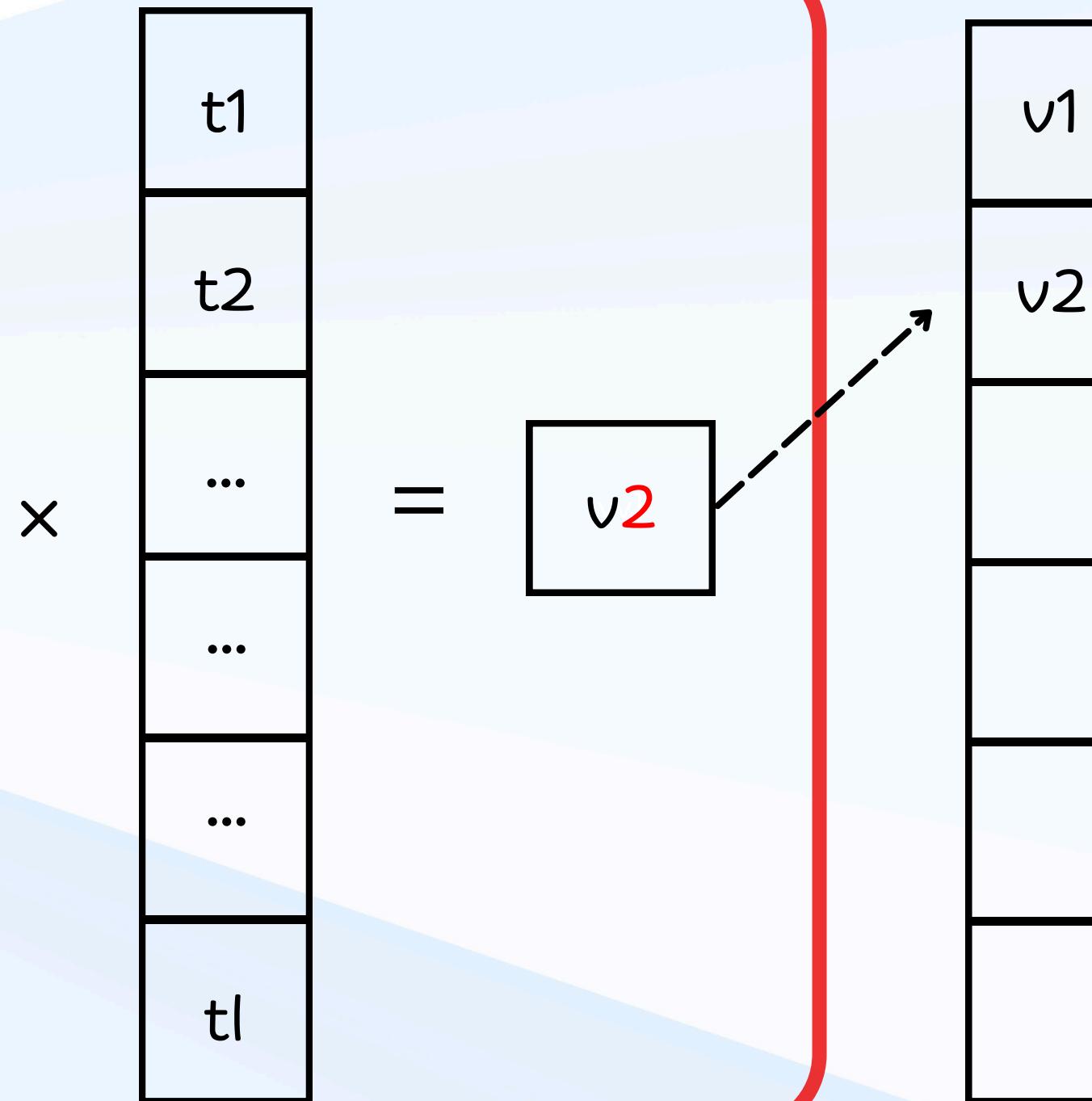
# Geração via vetor de polinômios (ML-DSA-v)

$i = 2$

Polinômios gerados a partir da seed  $\xi$ .



Sobrescrevendo o vetor anterior.

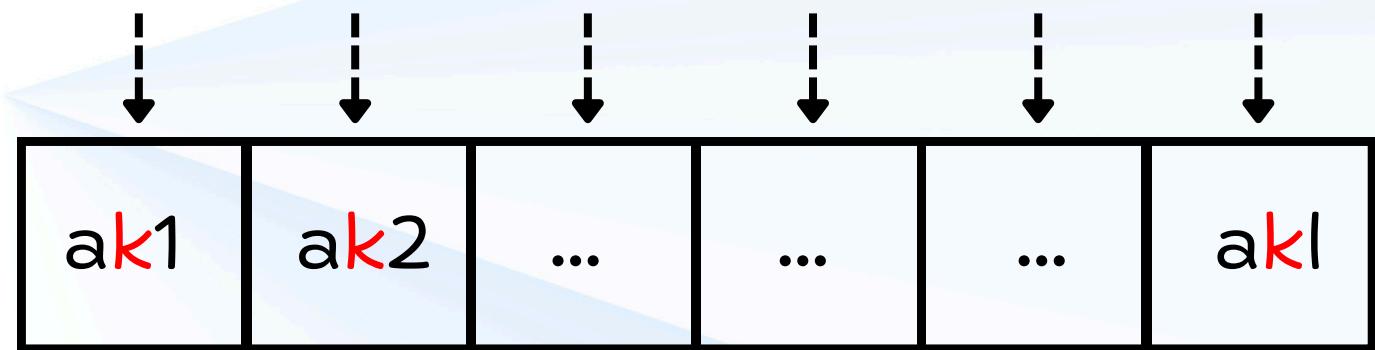


Polinômio  $v_2$  é gerado e armazenado

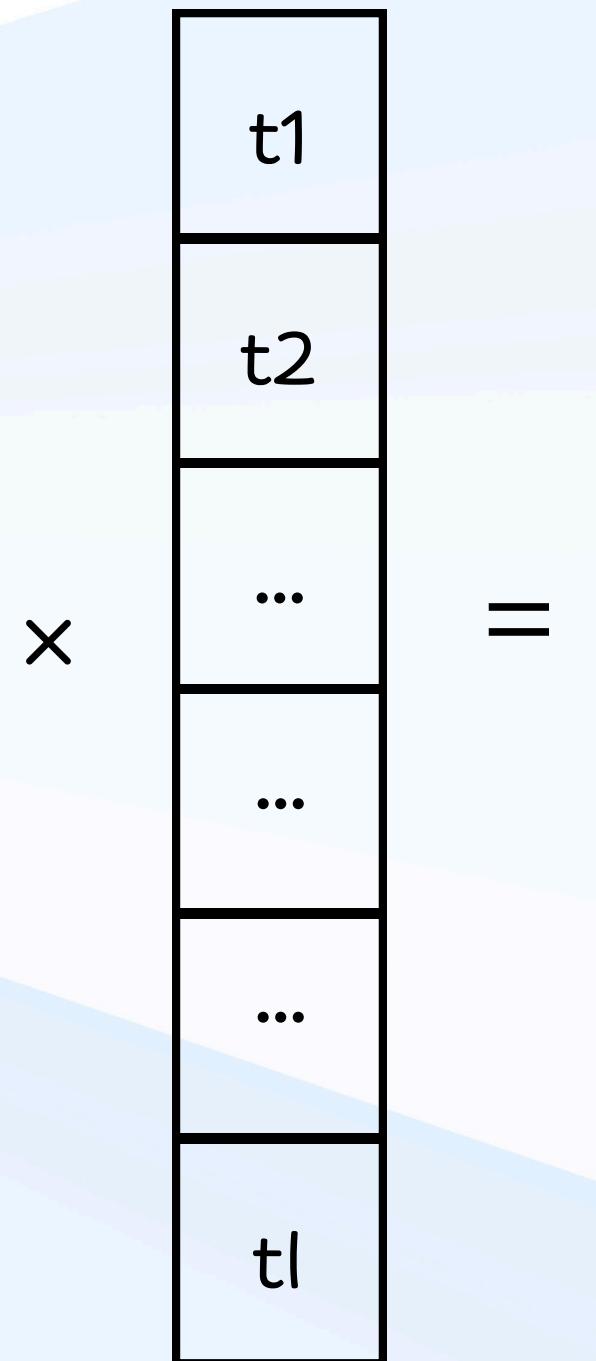
# Geração via vetor de polinômios (ML-DSA-v)

$i = k$

Polinômios gerados a partir da seed  $\xi$ .



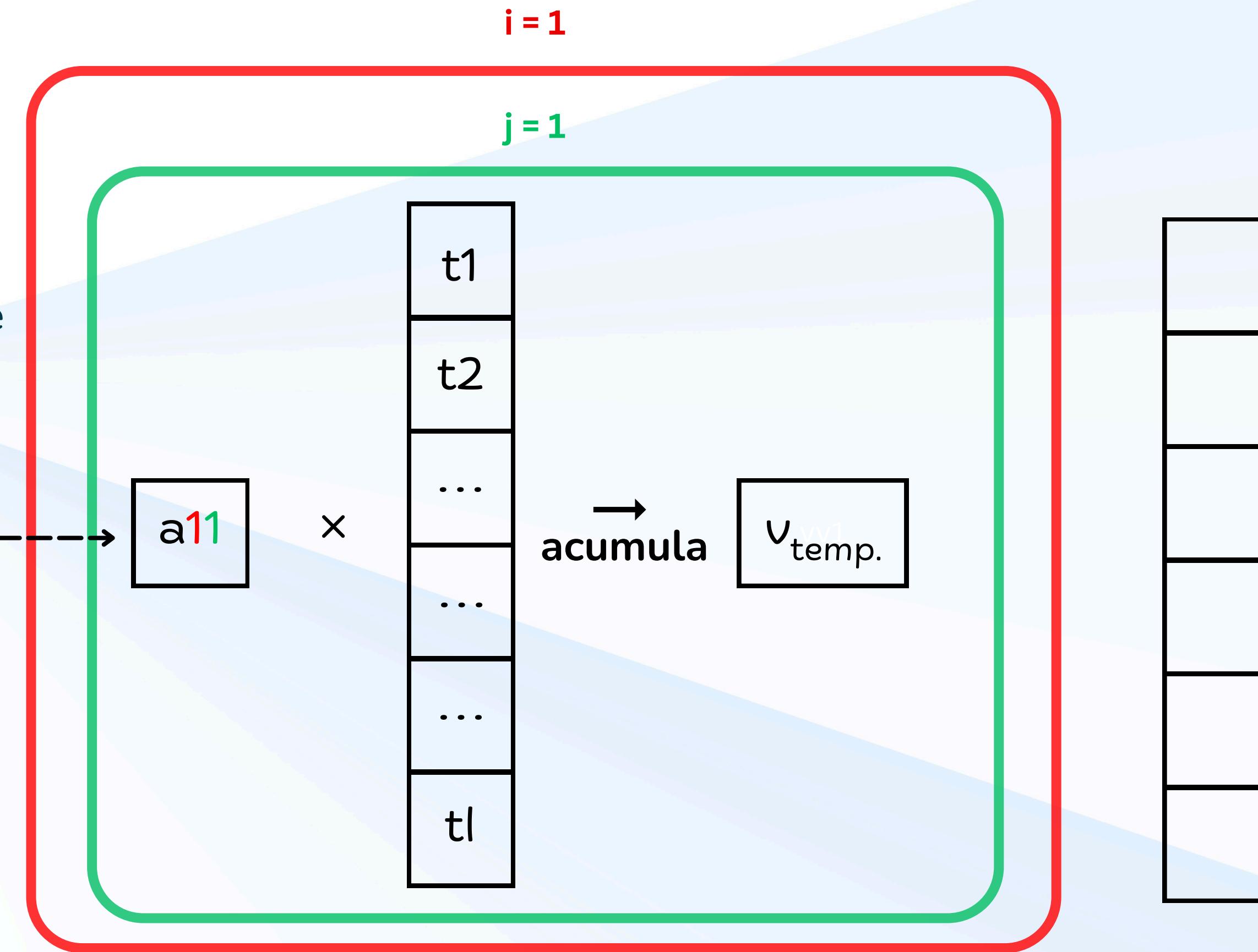
Sobrescrevendo o vetor anterior.



Polinômio  $v_k$  é  
gerado e arma-  
zenado

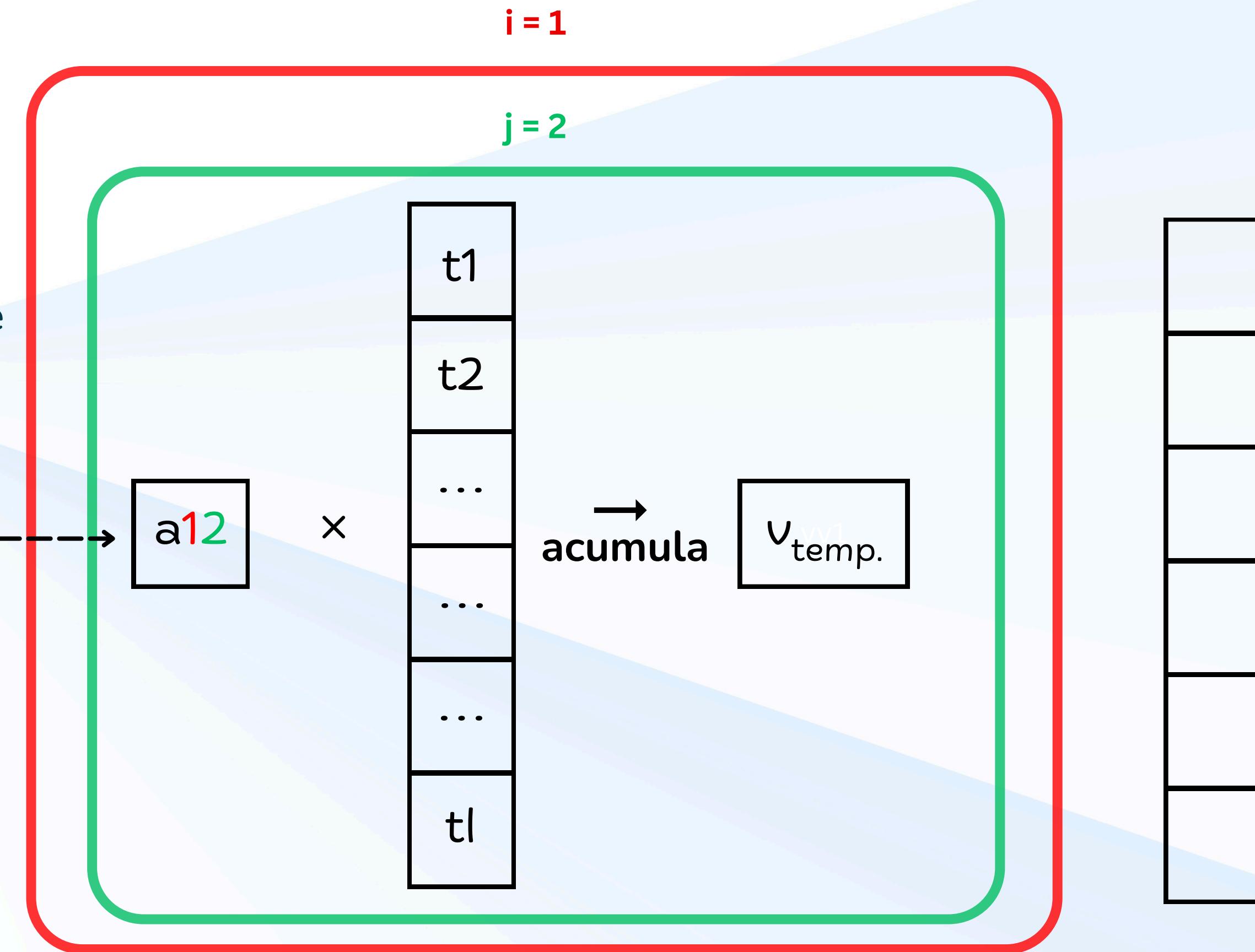
# Geração via um único polinômio (ML-DSA-p)

Polinômio  $a_{11}$  é gerado a partir da seed  $\xi$ .



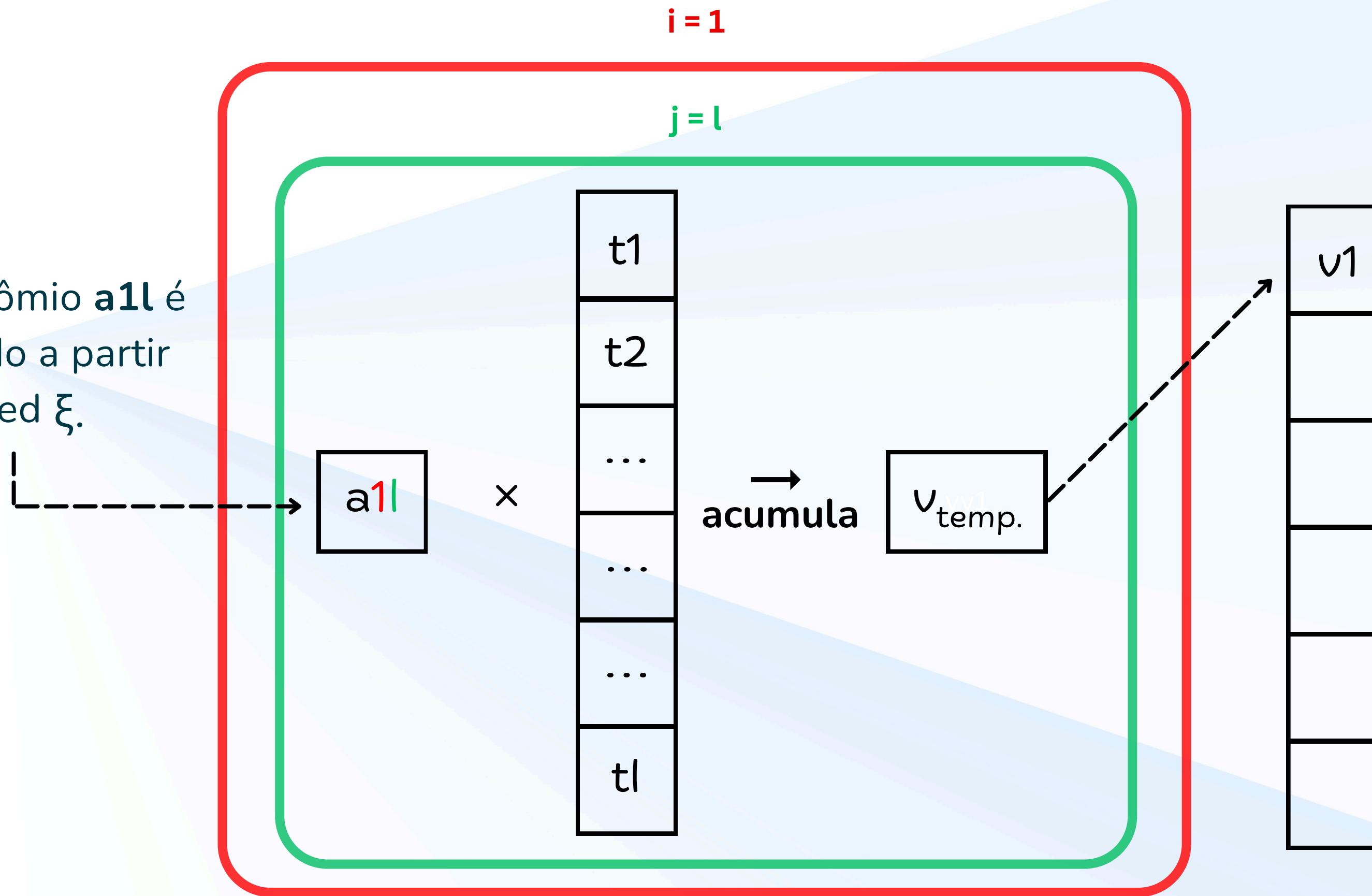
# Geração via um único polinômio (ML-DSA-p)

Polinômio  $a_{12}$  é gerado a partir da seed  $\xi$ .



# Geração via um único polinômio (ML-DSA-p)

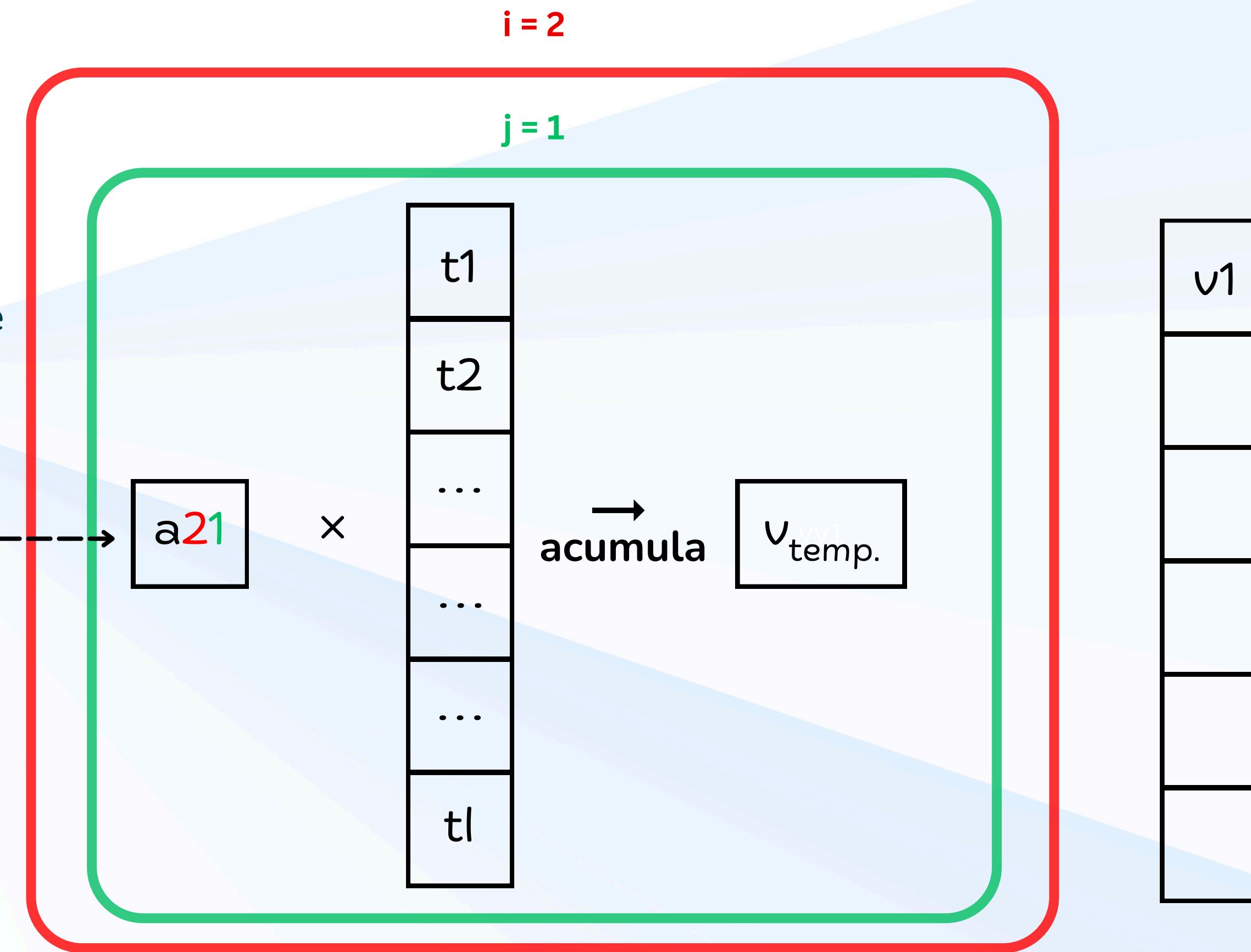
Polinômio  $a_{1l}$  é gerado a partir da seed  $\xi$ .



Polinômio  $v_1$  é gerado e armazenado

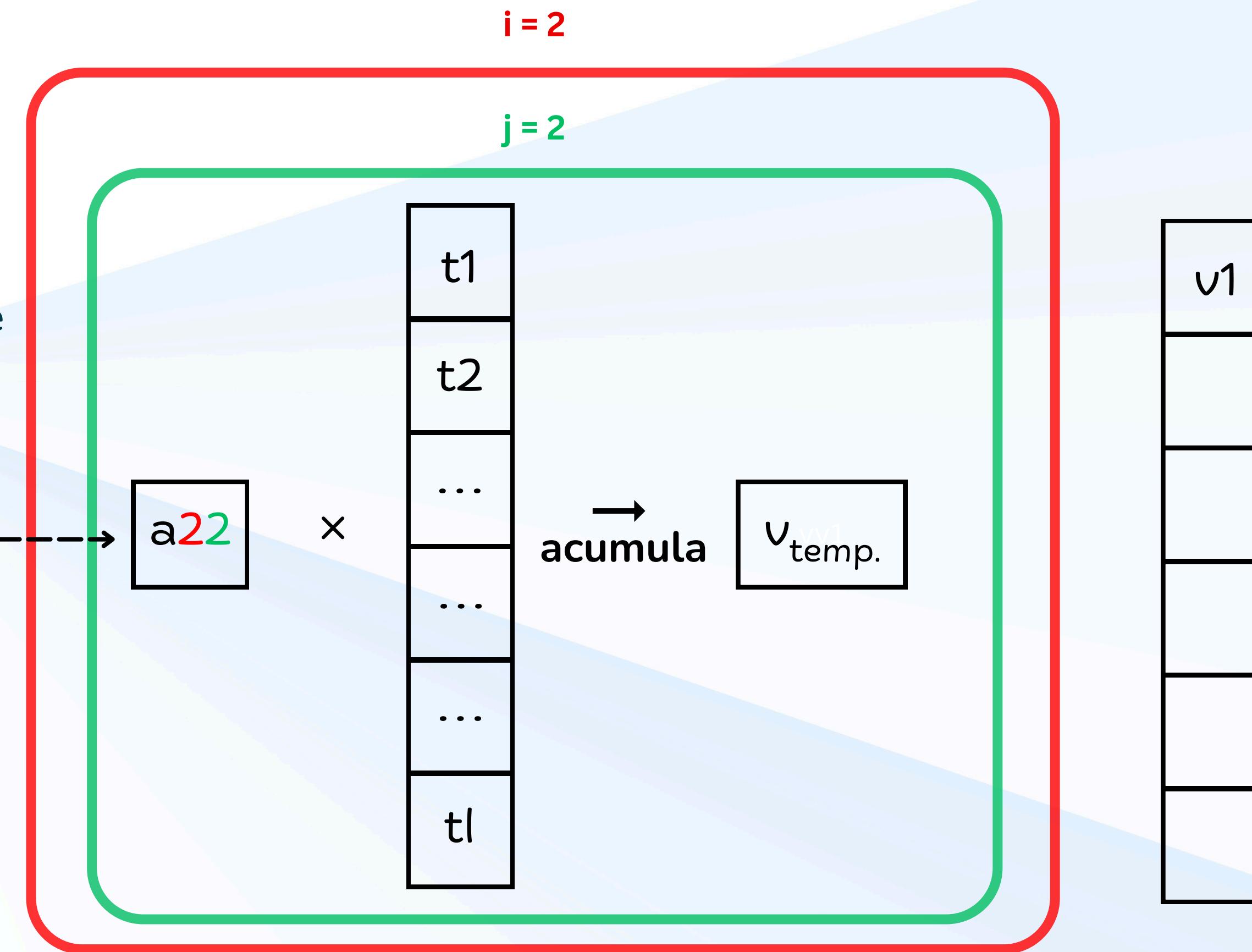
# Geração via um único polinômio (ML-DSA-p)

Polinômio  $a_{21}$  é gerado a partir da seed  $\xi$ .

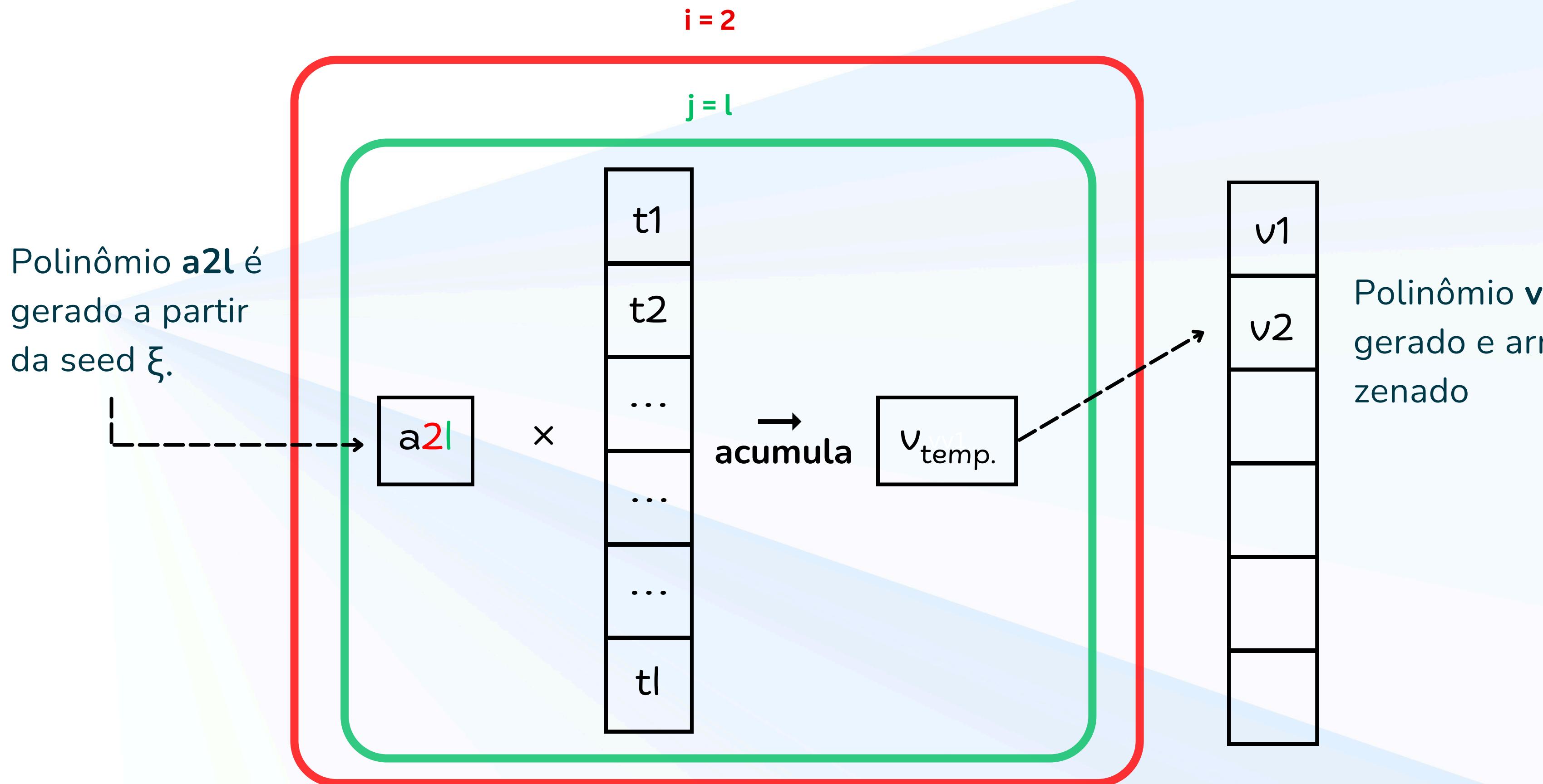


# Geração via um único polinômio (ML-DSA-p)

Polinômio  $a_{22}$  é gerado a partir da seed  $\xi$ .

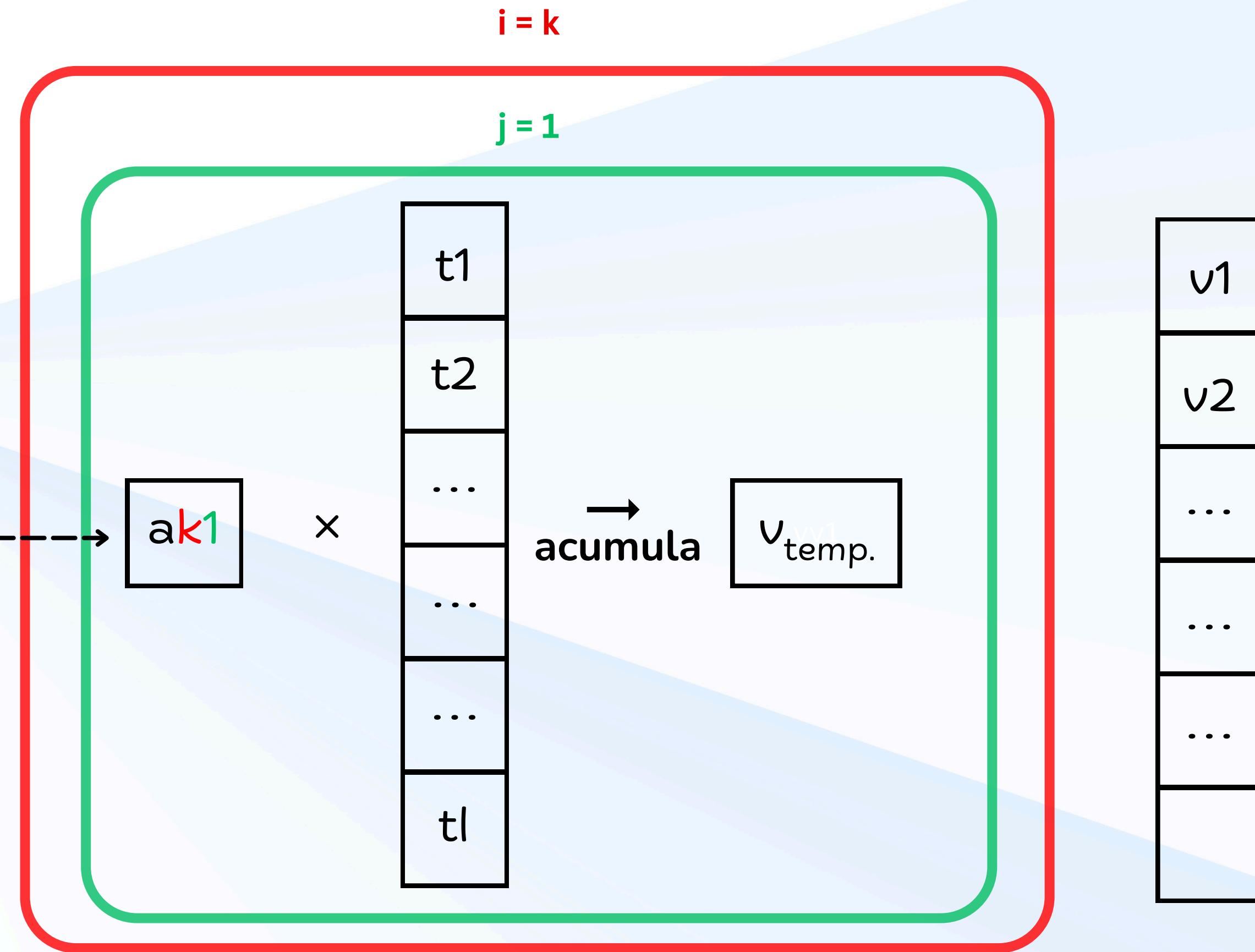


# Geração via um único polinômio (ML-DSA-p)



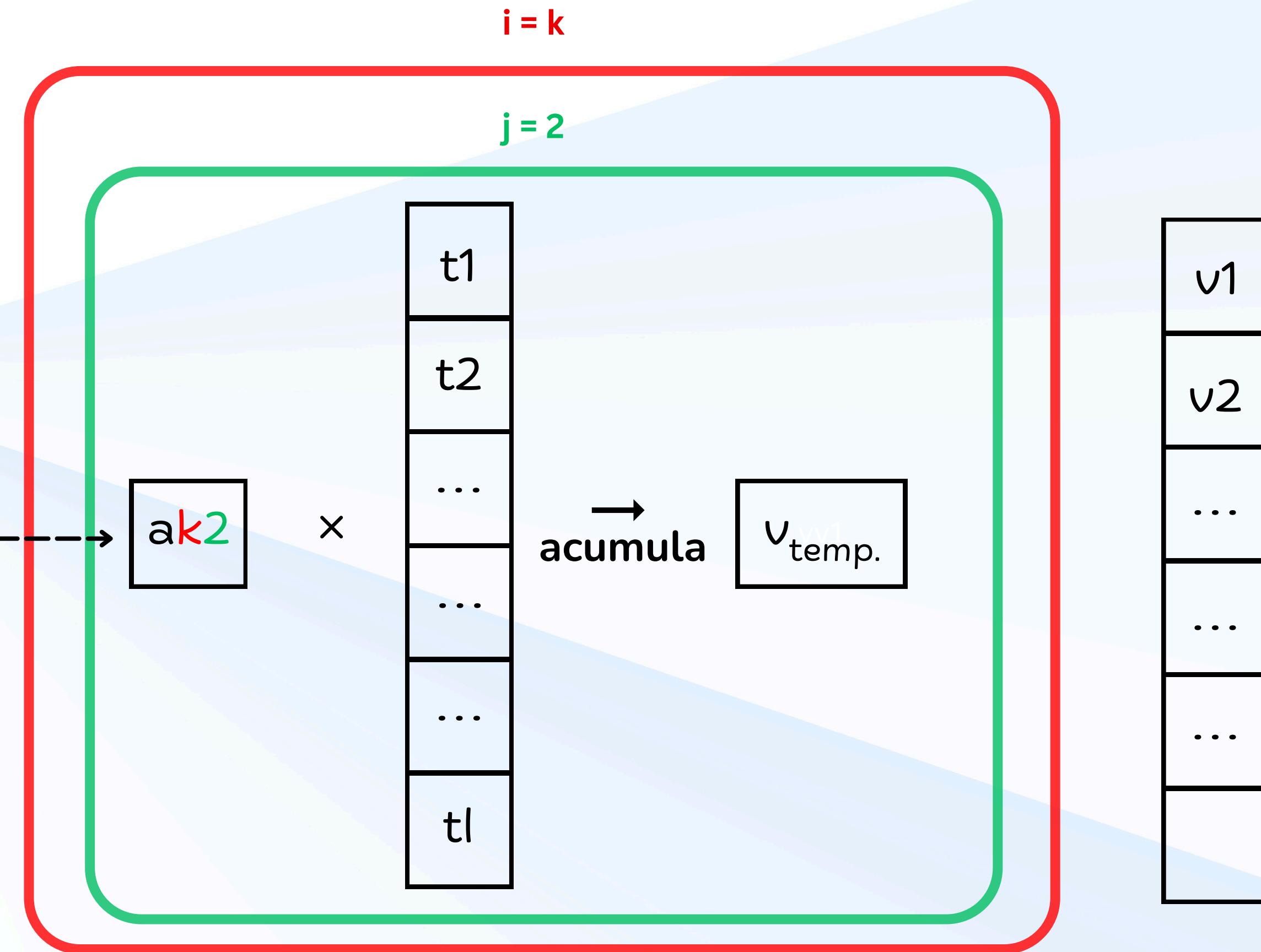
# Geração via um único polinômio (ML-DSA-p)

Polinômio  $ak_1$  é gerado a partir da seed  $\xi$ .



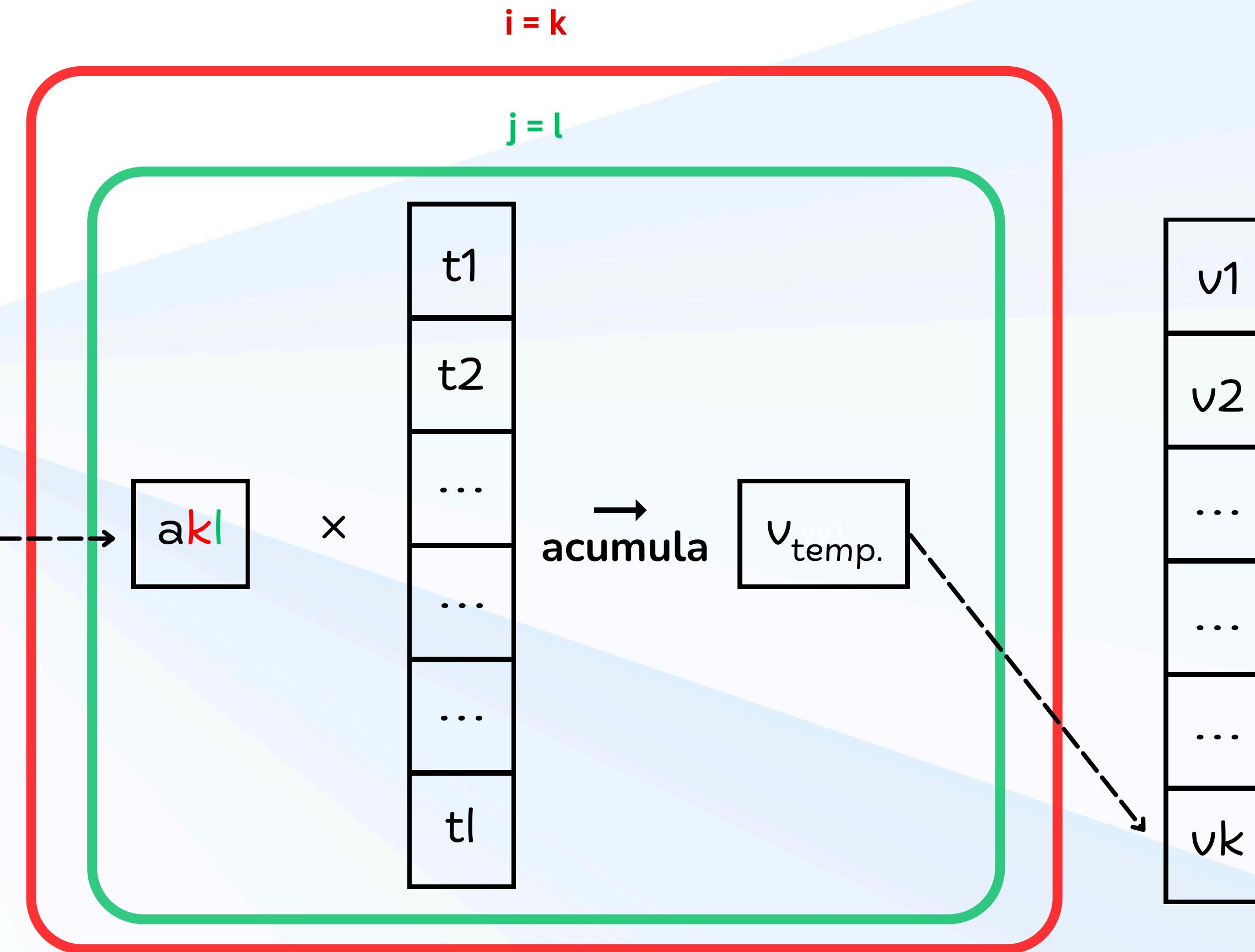
# Geração via um único polinômio (ML-DSA-p)

Polinômio  $ak2$  é gerado a partir da seed  $\xi$ .



# Geração via um único polinômio (ML-DSA-p)

Polinômio  $a_{kl}$  é gerado a partir da seed  $\xi$ .



Polinômio  $v_k$  é gerado e armazenado

# Método de análise

- Realizamos medidas para o uso de memória e ciclos de CPU para cada um dos níveis de segurança do ML-DSA.

# Materiais utilizados

- Utilizamos um laptop com processador Intel Core i7-1185G7, SO Ubuntu 22.04.4 LTS e 16 GiB de memória RAM.

- **PQClean:** implementações de referência em C puro.
- **Valgrind 3.18.1:** utilizado para as medições de uso de memória RAM.

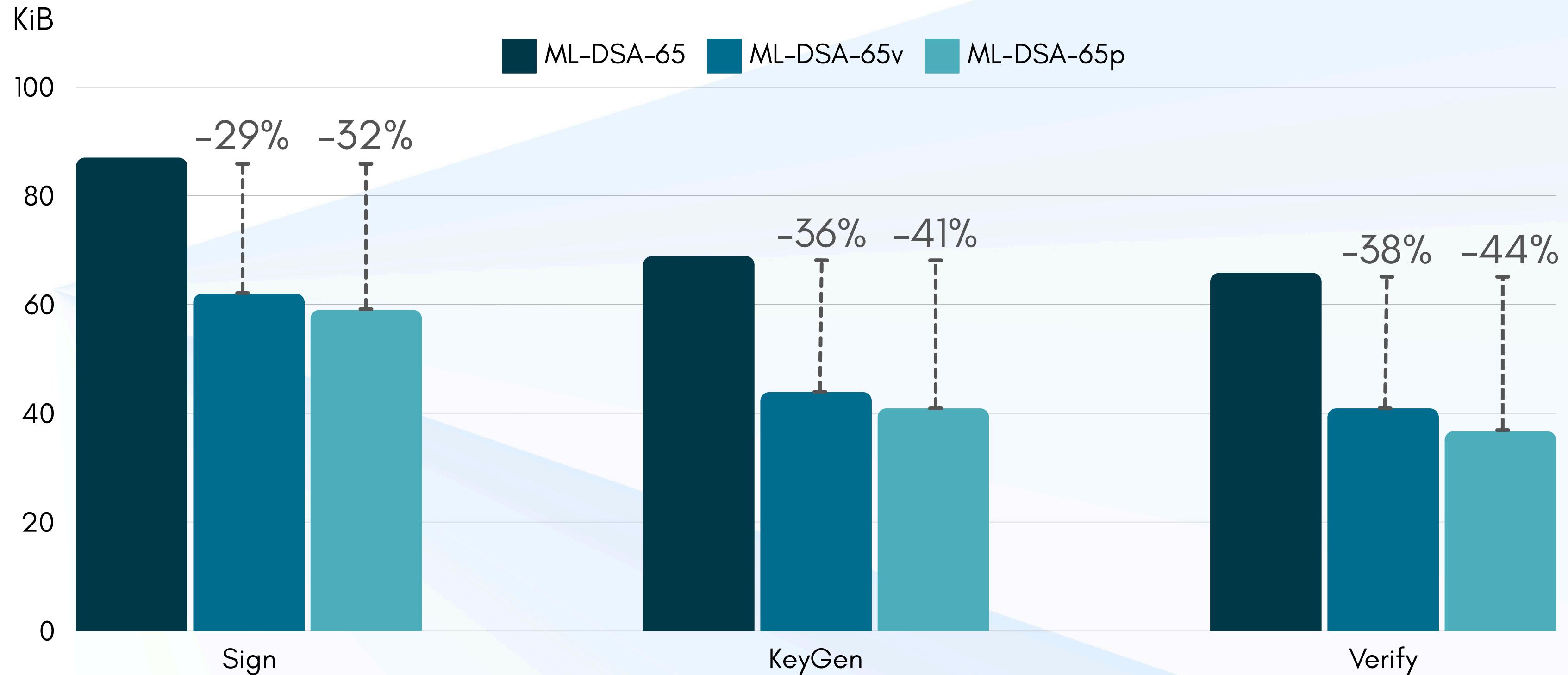
# Níveis de segurança:

- ML-DSA-44
- ML-DSA-65
- ML-DSA-87

# Procedimento experimental

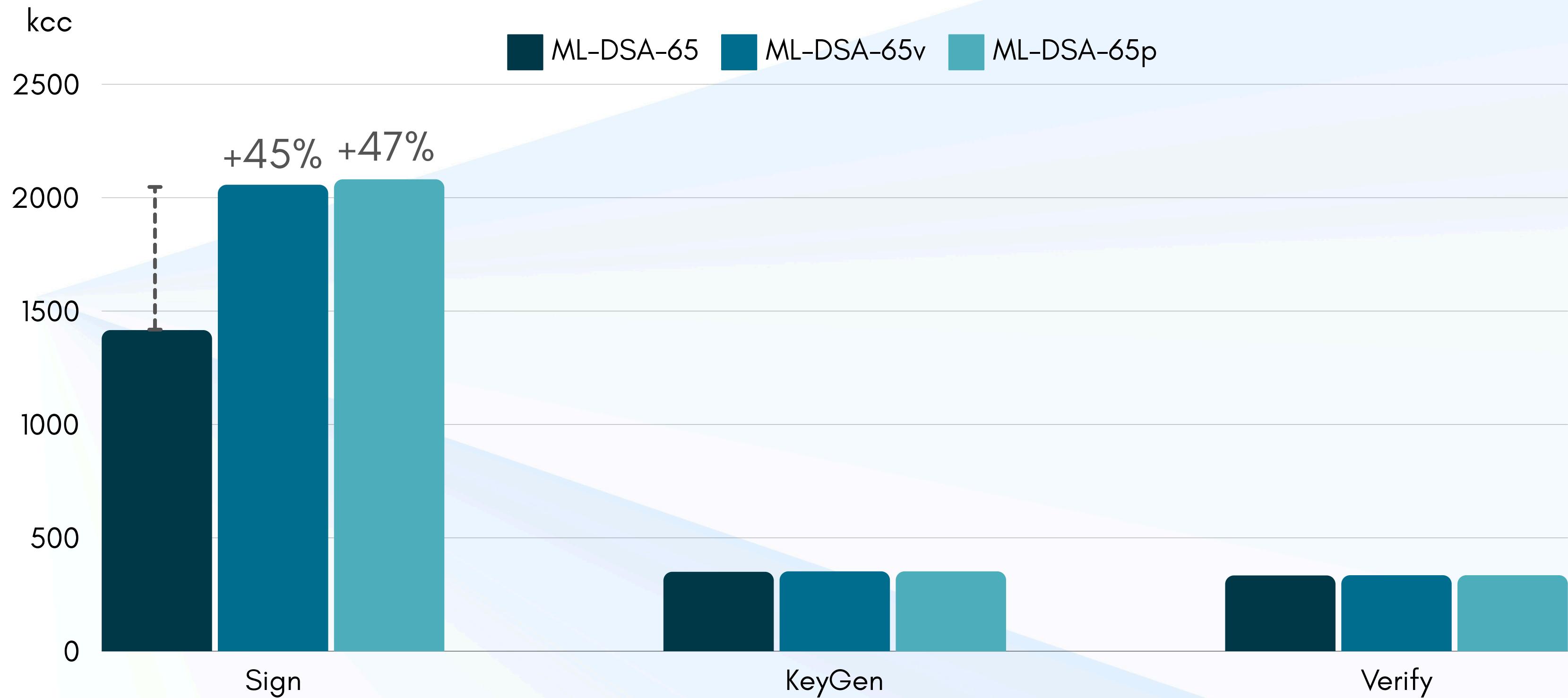
- **Picos de RAM:** análises feitas para uma única execução de KeyGen, Sign e Verify.
- **Ciclos de CPU:** medições feitas para 15000 iterações de KeyGen, Sign e Verify para uma mensagem aleatória de 59 bytes.

# Resultados



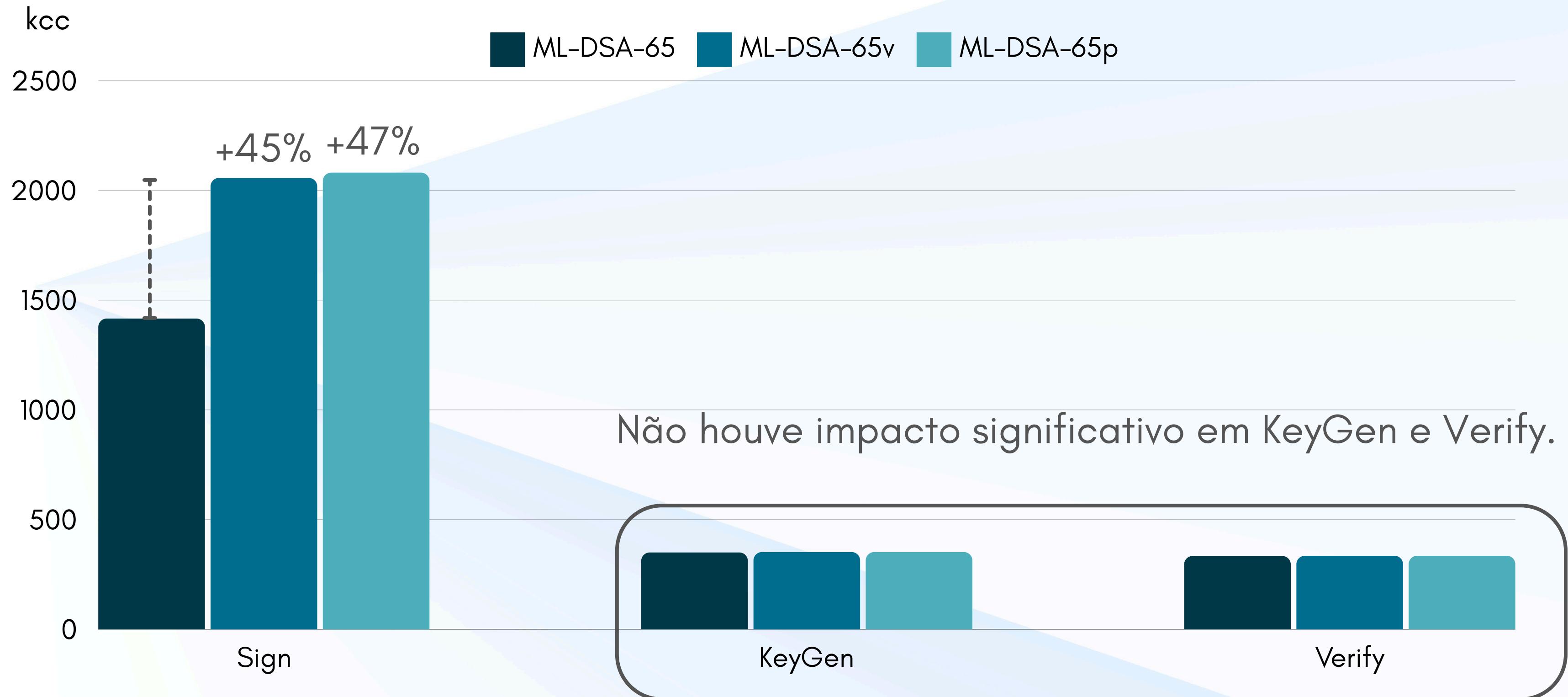
Consumo de memória RAM em ML-DSA-65

# Resultados



Média de ciclos de CPU em ML-DSA-65

# Resultados



Média de ciclos de CPU em ML-DSA-65

# Conclusão e trabalhos futuros

- Redução significativa no uso de memória RAM para todos os níveis de segurança;
- Há um aumento tolerável no número médio de ciclos de processador;
- As otimizações propostas são independentes da arquitetura de processador utilizada.

# Conclusão e trabalhos futuros

- Redução significativa no uso de memória RAM para todos os níveis de segurança;
  - Há um aumento tolerável no número médio de ciclos de processador;
  - As otimizações propostas são independentes da arquitetura de processador utilizada.
- Aplicação das mesmas otimizações para o **ML-KEM (Kyber)**.
  - Avaliar as implementações propostas em ambientes computacionalmente restritos.

# Obrigado!



r197962@dac.unicamp.br



github.com/regras

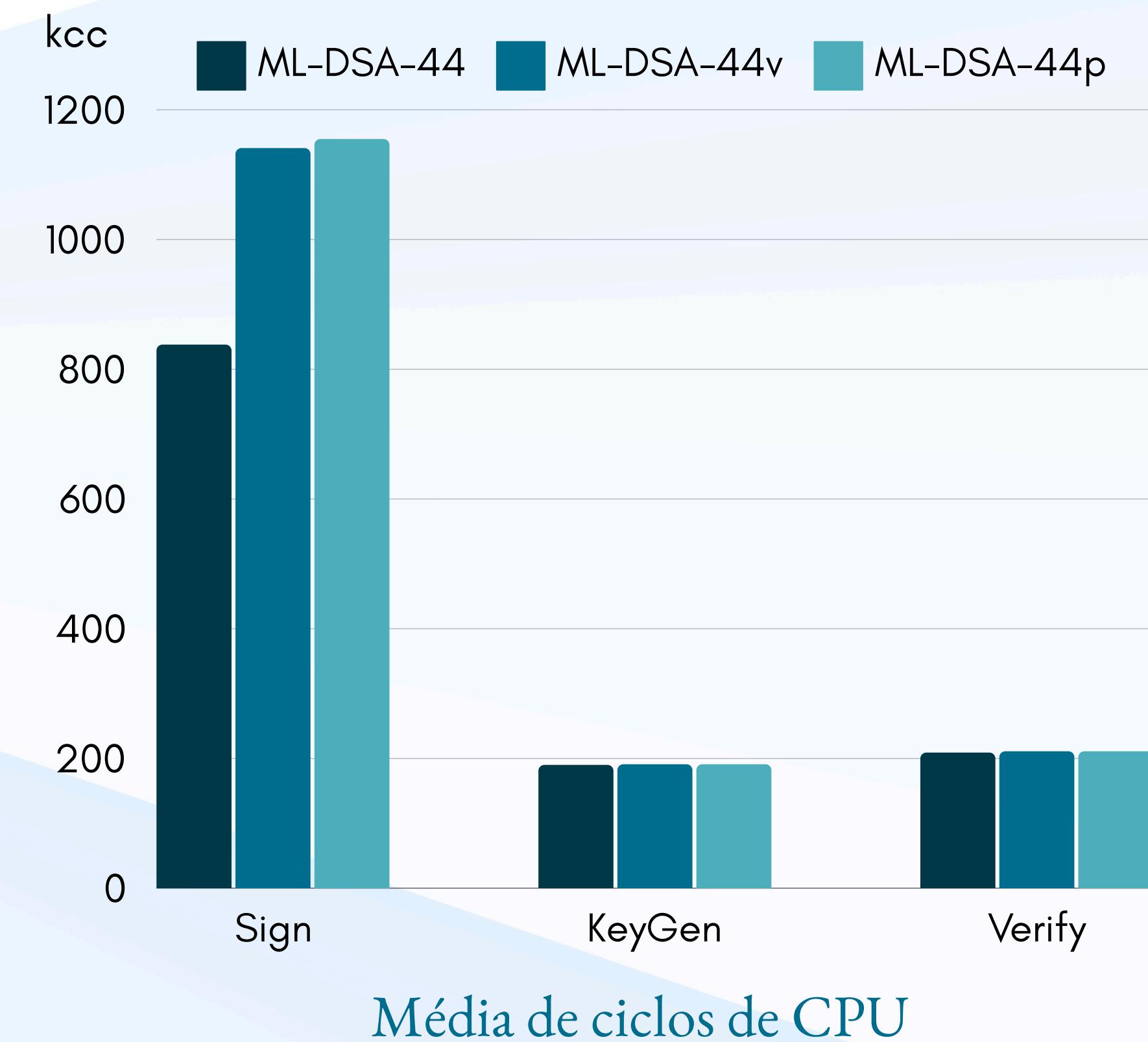
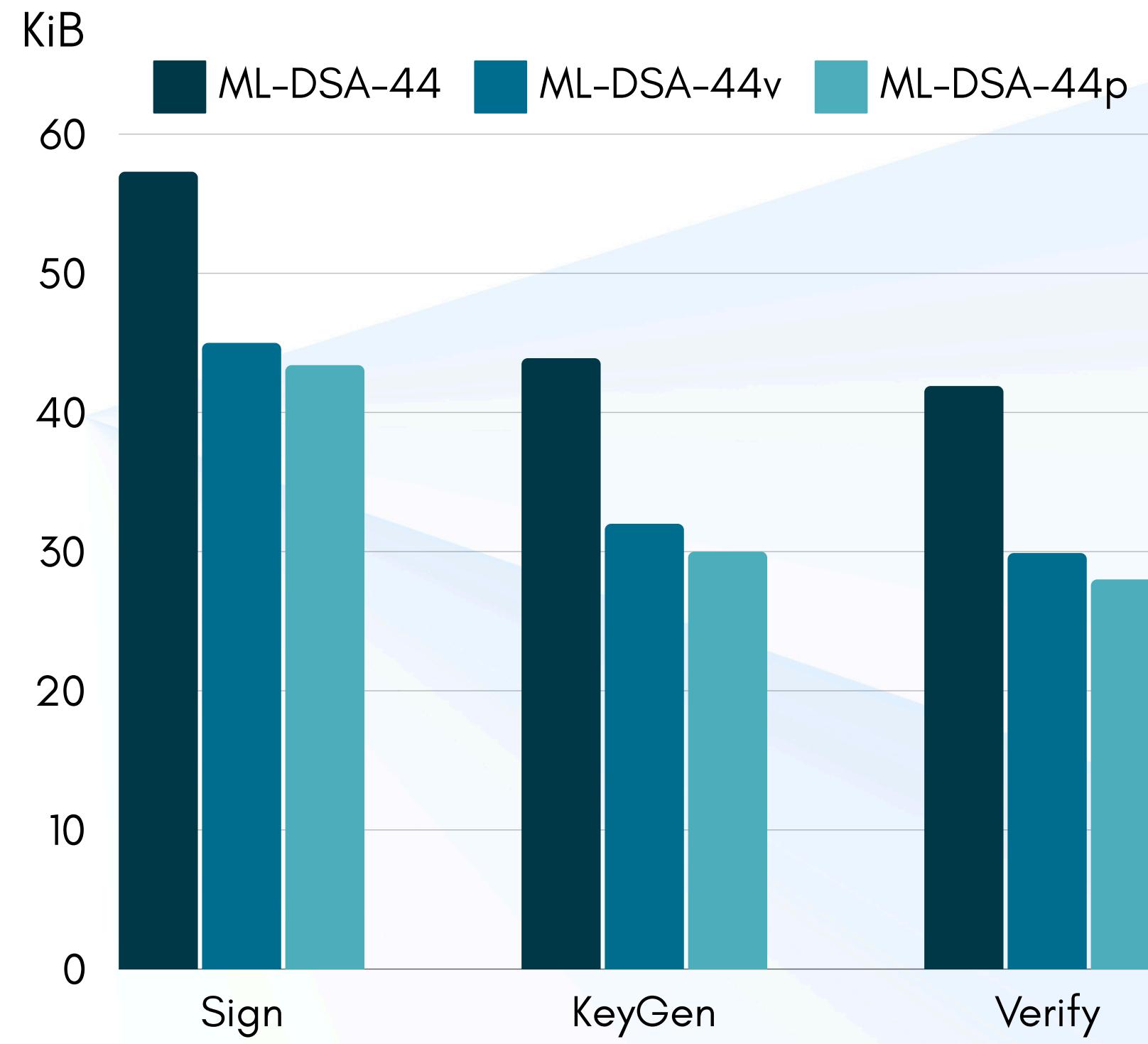


# Compact Memory Implementations of the ML-DSA Post-Quantum Digital Signature Algorithm

R. D. Meneses, C. Teixeira, M. A. A. Henriques

16 de Setembro, 2024

# Apêndice I: resultados para ML-DSA-44



# Apêndice II: resultados para ML-DSA-87

