ASCON: Analisi prestazionale del nuovo standard internazionale per la crittografia lightweight

Laurea Triennale in Informatica

Mattia Oldani (966668)

18 Aprile 2024

UNIVERSITÀ DEGLI STUDI DI MILANO

Indice

Obiettivi del lavoro di tirocinio e tecnologie utilizzate

- Raccolta di (1) **tempi di esecuzione** e (2) **spazio utilizzato** degli algoritmi lightweight proposti nella famiglia ASCON

Obiettivi del lavoro di tirocinio e tecnologie utilizzate

- Raccolta di (1) **tempi di esecuzione** e (2) **spazio utilizzato** degli algoritmi lightweight proposti nella famiglia ASCON
- **Analisi dei dati raccolti** per mostrare quali algoritmi sono i migliori e, per ogni algoritmo, quali implementazioni sono le migliori

Obiettivi del lavoro di tirocinio e tecnologie utilizzate

- Raccolta di (1) **tempi di esecuzione** e (2) **spazio utilizzato** degli algoritmi lightweight proposti nella famiglia ASCON
- **Analisi dei dati raccolti** per mostrare quali algoritmi sono i migliori e, per ogni algoritmo, quali implementazioni sono le migliori

Obiettivi del lavoro di tirocinio e tecnologie utilizzate

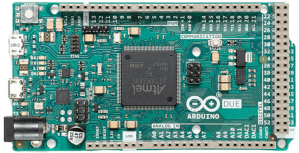
- Raccolta di (1) **tempi di esecuzione** e (2) **spazio utilizzato** degli algoritmi lightweight proposti nella famiglia ASCON
- **Analisi dei dati raccolti** per mostrare quali algoritmi sono i migliori e, per ogni algoritmo, quali implementazioni sono le migliori

Obiettivi del lavoro di tirocinio e tecnologie utilizzate

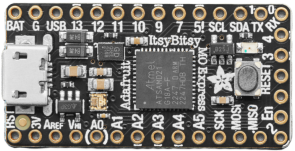
- Raccolta di (1) **tempi di esecuzione** e (2) **spazio utilizzato** degli algoritmi lightweight proposti nella famiglia ASCON
- **Analisi dei dati raccolti** per mostrare quali algoritmi sono i migliori e, per ogni algoritmo, quali implementazioni sono le migliori



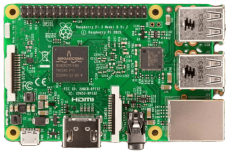

Dispositivi utilizzati



(a) Arduino Due.



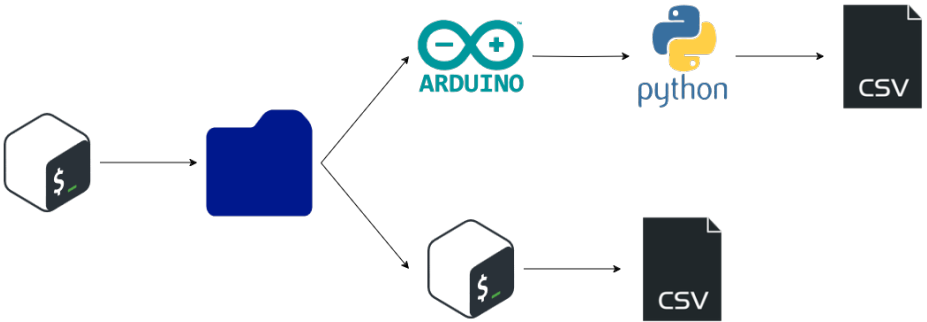
(b) Adafruit ItsyBitsy M0 Express.



(c) Raspberry Pi 3 Model B.

Workflow

1 Introduzione al problema



Indice

2 Testing e analisi dei risultati

Sotto-famiglie di ASCON

2 Testing e analisi dei risultati

Hash
Algoritmi hash e XOF

AEAD
Algoritmi di cifratura autenticata

Auth
Algoritmi MAC e PRF

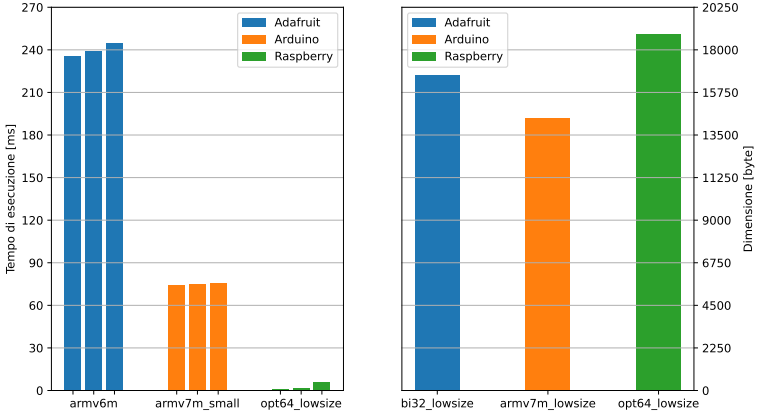
Per ogni sotto-famiglia qui citata è stato scelto un algoritmo e analizzato per tempi di esecuzione e spazio utilizzato; i risultati di tale analisi sono riportati in due tipi di grafico

Tempi di esecuzione
Contiene gruppi di tre colonne che rappresentano, per ogni implementazione, le misurazioni minima, media e massima

Spazio utilizzato
Contiene una sola colonna per ogni implementazione e ognuna indica la dimensione dell'eseguibile

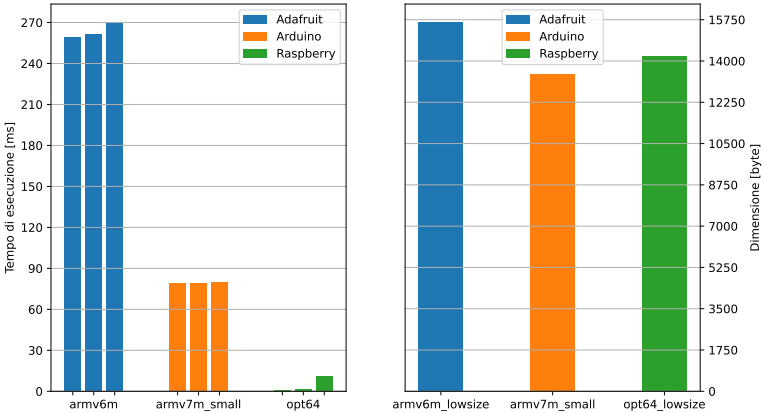
AEAD: ascon128a

2 Testing e analisi dei risultati



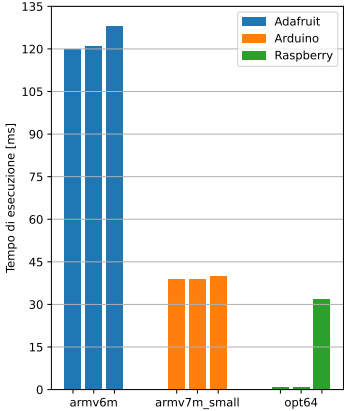
Hash: asconhasha

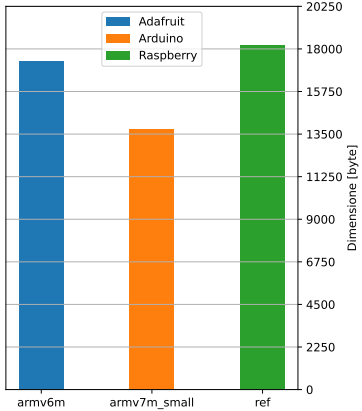
2 Testing e analisi dei risultati



Auth: asconmaca

2 Testing e analisi dei risultati





Migliori implementazioni per ogni dispositivo

2 Testing e analisi dei risultati

Dispositivo	Ottimizzazione proposta		
	Tempo	Spazio	Ibrida
Adafruit	armv6m	lowsize	armv6m
Arduino	armv7m_small	armv7m_small	armv7m_small
Raspberry	opt64	opt64_lowsize	opt64

Indice

3 Conclusioni

Risultati ottenuti

3 Conclusioni

1. **Ottimi tempi di esecuzione** — Gli algoritmi analizzati sono molto veloci considerando l'ambiente limitato nel quale sono eseguiti

Risultati ottenuti

3 Conclusioni

1. **Ottimi tempi di esecuzione** — Gli algoritmi analizzati sono molto veloci considerando l'ambiente limitato nel quale sono eseguiti
2. **Poco spazio utilizzato** — Alcune implementazioni riducono drasticamente la grandezza dell'eseguibile, rendendoli ottimi in ambiti dove la memoria è limitata

Risultati ottenuti

3 Conclusioni

1. **Ottimi tempi di esecuzione** — Gli algoritmi analizzati sono molto veloci considerando l'ambiente limitato nel quale sono eseguiti
2. **Poco spazio utilizzato** — Alcune implementazioni riducono drasticamente la grandezza dell'eseguibile, rendendoli ottimi in ambiti dove la memoria è limitata
3. **Duttilità** — Ci sono implementazioni che ottimizzano – o avvicinano all'ottimo – entrambi gli aspetti analizzati, come la `armv7m_small` del dispositivo Arduino

Sviluppi futuri

3 Conclusioni

1. **Raccolta dati** — Analisi dei cicli della CPU

Sviluppi futuri

3 Conclusioni

1. **Raccolta dati** — Analisi dei cicli della CPU
2. **Board** — Architetture IoT non testate (ARMv6, ARM neon, ESP32, AVR, eccetera)

Sviluppi futuri

3 Conclusioni

1. **Raccolta dati** — Analisi dei cicli della CPU
2. **Board** — Architetture IoT non testate (ARMv6, ARM neon, ESP32, AVR, eccetera)
3. **Plaintext** — File di grandezza maggiore di 1024 byte (immagini o video)

Sviluppi futuri

3 Conclusioni

1. **Raccolta dati** — Analisi dei cicli della CPU
2. **Board** — Architetture IoT non testate (ARMv6, ARM neon, ESP32, AVR, eccetera)
3. **Plaintext** — File di grandezza maggiore di 1024 byte (immagini o video)
4. **Testing automatico** — Realizzazione di script che automatizzino il testing usando, ad esempio, l'Arduino IDE dal terminale e non tramite la GUI

Grazie per l'attenzione!

Processo di standardizzazione

3 Conclusioni