CS 463 | Crypotography for Cybersecurity

Course Project

5 December 2025

Cody Ruhlen
01332300
CS463:16868

Instructor: J. Takeshita
Institution: Old Dominion University

[7]:
```
import time
import tracemalloc
import gc
from Crypto.Cipher import AES
import ascon
from simon import SimonCipher
from speck import SpeckCipher
import matplotlib.pyplot as plt

message_sizes = [64, 256, 1024, 4096]
runs = 6

key_aes = b"thequickbrownfox"
key_ascon = b"thequickbrownfox"
key_simon = int.from_bytes(b"thequickbrownfox"[:8], "big")
key_speck = int.from_bytes(b"thequickbrownfox"[:8], "big")
nonce = b"andthreelazydogs"
block_size_bytes = 8
def the_message(size):
    return (b"jumpedoverazebra" * ((size // 16) + 1))[:size]

def measure_peak(func):
    gc.collect()
    tracemalloc.start()
    start_time = time.perf_counter()
    func()
    end_time = time.perf_counter()
    current, peak = tracemalloc.get_traced_memory()
    tracemalloc.stop()
    peak_mb = peak / 1_000_000
    runtime = end_time - start_time
    return runtime, peak_mb

def print_avg(cipher, size, avg_time, avg_mem):
    print(f"{cipher} {size} byte | Average Time: {avg_time:.6f} sec |␣
 ↪Average Memory Used: {avg_mem:.6f} MB")

def print_total(cipher, tottime, peak_bytes):
    print(f"{cipher} Totals | Time: {tottime:.6f} sec | Peak Memory:␣
 ↪{peak_bytes/1_000_000:.6f} MB")
```

```
def run_aes(size):
    times, mem = [], []
    message = the_message(size)

    for run in range(1, runs + 1):
        def task():
            cipher = AES.new(key_aes, AES.MODE_ECB)
            ct = cipher.encrypt(message.ljust((len(message)//16 + 1)*16,␣
 ↪b'\0'))
            pt = cipher.decrypt(ct).rstrip(b'\0')
            assert pt == message

        runtime, peak_mb = measure_peak(task)

        if run > 1:
            times.append(runtime)
            mem.append(peak_mb)
            print(f"Run {run-1}")
            print(f"  Time {runtime:.6f} sec | Mem {peak_mb:.6f} MB")

    return times, mem

def run_ascon(size):
    times, mem = [], []
    message = the_message(size)

    for run in range(1, runs + 1):
        def task():
            ct = ascon.encrypt(key_ascon, nonce, b"", message,␣
 ↪variant="Ascon-128")
            pt = ascon.decrypt(key_ascon, nonce, b"", ct,␣
 ↪variant="Ascon-128")
            assert pt == message

        runtime, peak_mb = measure_peak(task)

        if run > 1:
            times.append(runtime)
            mem.append(peak_mb)
```

```
            print(f"Run {run-1}")
            print(f"  Time {runtime:.6f} sec | Mem {peak_mb:.6f} MB")

    return times, mem

def run_simon(size):
    times, mem = [], []
    message = the_message(size)

    for run in range(1, runs + 1):
        def task():
            cipher = SimonCipher(key_simon)
            blocks = []
            for i in range(0, len(message), block_size_bytes):
                b = message[i:i+block_size_bytes].ljust(block_size_bytes,␣
 ↪b"\0")
                blocks.append(cipher.encrypt(int.from_bytes(b, "big")))
            out = b""
            for c in blocks:
                out += cipher.decrypt(c).to_bytes(block_size_bytes, "big")
            out = out[:len(message)]
            assert out == message

        runtime, peak_mb = measure_peak(task)

        if run > 1:
            times.append(runtime)
            mem.append(peak_mb)
            print(f"Run {run-1}")
            print(f"  Time {runtime:.6f} sec | Mem {peak_mb:.6f} MB")

    return times, mem

def run_speck(size):
    times, mem = [], []
    message = the_message(size)

    for run in range(1, runs + 1):
        def task():
            cipher = SpeckCipher(key_speck)
```

```
            blocks = []
            for i in range(0, len(message), block_size_bytes):
                b = message[i:i+block_size_bytes].ljust(block_size_bytes,␣
 ↪b"\0")
                blocks.append(cipher.encrypt(int.from_bytes(b, "big")))
            out = b""
            for c in blocks:
                out += cipher.decrypt(c).to_bytes(block_size_bytes, "big")
            out = out[:len(message)]
            assert out == message

        runtime, peak_mb = measure_peak(task)

        if run > 1:
            times.append(runtime)
            mem.append(peak_mb)
            print(f"Run {run-1}")
            print(f"  Time {runtime:.6f} sec | Mem {peak_mb:.6f} MB")

    return times, mem

totals = {
    "AES-128": {"total_time": 0.0, "peak_mem_bytes": 0},
    "Ascon":   {"total_time": 0.0, "peak_mem_bytes": 0},
    "Simon":   {"total_time": 0.0, "peak_mem_bytes": 0},
    "Speck":   {"total_time": 0.0, "peak_mem_bytes": 0},
}

tracemalloc.start()

per_size_results = {size: {} for size in message_sizes}

for size in message_sizes:
    print(f"\n{size} Byte Tests")

    for cipher_name in ["AES-128", "Ascon", "Simon", "Speck"]:
        print(f"\n{cipher_name}: {size} bytes")

        if cipher_name == "AES-128":
            times, mem = run_aes(size)
```

```
        elif cipher_name == "Ascon":
            times, mem = run_ascon(size)

        elif cipher_name == "Simon":
            times, mem = run_simon(size)

        elif cipher_name == "Speck":
            times, mem = run_speck(size)

        totals[cipher_name]["total_time"] += sum(times)
        totals[cipher_name]["peak_mem_bytes"] = max(
            totals[cipher_name]["peak_mem_bytes"],
            max(mem) * 1_000_000
        )

        avg_time = sum(times) / (runs-1)
        avg_mem = sum(mem) / (runs-1)

        print("-" * 82)
        print_avg(cipher_name, size, avg_time, avg_mem)

        per_size_results[size][cipher_name] = {
            "avg_time": avg_time,
            "avg_mem": avg_mem,
        }

    print("\n" + "-" * 82)
    print(f"{size} byte Winners sorted by Efficiency Ratio")
    print("-" * 82)

    winners = []
    for cipher_name in ["AES-128", "Ascon", "Simon", "Speck"]:
        pdata = per_size_results[size][cipher_name]
        peak_mem_size = pdata["avg_mem"]
        total_time_size = pdata["avg_time"]
        ratio = (peak_mem_size / total_time_size) if total_time_size > 0␣
↪else float("inf")
        winners.append((
            cipher_name,
```

```
            ratio,
            pdata["avg_time"],
            pdata["avg_mem"],
        ))

    winners_sorted = sorted(winners, key=lambda x: x[1])

    print(f"{'Rank':<6} {'Cipher':<10} {'Time Used':>14} {'Memory Used':
↪>13} {'Efficiency Ratio':>21}")
    print("-" * 82)

    for rank, item in enumerate(winners_sorted, start=1):
        name, ratio, avg_time, avg_mem = item
        print(f"{rank:<6} {name:<10} {avg_time:10.6f} sec {avg_mem:10.6f}␣
↪MB {ratio:14.6f} MB/sec")

print("\n" + "*" * 82)
print("Cipher Totals & Efficiency Ratios (across all sizes, combined␣
↪runs)")
print("*" * 82)

final_results = []
for name, data in totals.items():
    total_time = float(data["total_time"])
    peak_bytes = int(data["peak_mem_bytes"])
    peak_mb = peak_bytes / 1_000_000
    ratio = peak_mb / total_time if total_time > 0 else float("inf")
    final_results.append((name, total_time, peak_mb, ratio))

print(f"{'Cipher':<10} {'Time Used':>16} {'Memory Used':>18} {'Efficiency␣
↪Ratio':>25}")
print("-" * 72)
for name, total_time, peak_mb, ratio in final_results:
    print(f"{name:<10} {total_time:12.6f} sec {peak_mb:15.6f} MB {ratio:
↪18.6f} MB/sec")

print("\n" + "=" * 82)
print("Efficiency Rankings")
print("=" * 82)
```

```
ranked = sorted(final_results, key=lambda x: x[3])

print(f"\n{'Rank':<6} {'Cipher':<10} {'Efficiency Ratio':>27}")
print("-" * 45)
for i, (name, _, _, ratio) in enumerate(ranked, 1):
    print(f"{i:<6} {name:<10} {ratio:20.6f} MB/sec")

tracemalloc.stop()

ciphers = ["AES-128", "Ascon", "Simon", "Speck"]

times_plot = {cipher: [] for cipher in ciphers}
mem_plot   = {cipher: [] for cipher in ciphers}
ratio_plot = {cipher: [] for cipher in ciphers}

for size in message_sizes:
    for cipher in ciphers:
        t = per_size_results[size][cipher]["avg_time"]
        m = per_size_results[size][cipher]["avg_mem"]
        r = m / t if t > 0 else float("inf")
        times_plot[cipher].append(t)
        mem_plot[cipher].append(m)
        ratio_plot[cipher].append(r)

plt.figure(figsize=(10,6))
for cipher in ciphers:
    plt.plot(message_sizes, times_plot[cipher], marker='o', label=cipher)

plt.xlabel("Message Size (bytes)")
plt.ylabel("Average Runtime (seconds)")
plt.title("Cipher Runtime vs Message Size")
plt.grid(True)
plt.legend()
plt.xticks(message_sizes)
plt.tight_layout()
plt.show()

plt.figure(figsize=(10,6))
for cipher in ciphers:
    plt.plot(message_sizes, mem_plot[cipher], marker='o', label=cipher)
```

```
plt.xlabel("Message Size (bytes)")
plt.ylabel("Average Memory Used (MB)")
plt.title("Cipher Memory Usage vs Message Size")
plt.grid(True)
plt.legend()
plt.xticks(message_sizes)
plt.tight_layout()
plt.show()

plt.figure(figsize=(10,6))
for cipher in ciphers:
    plt.plot(message_sizes, ratio_plot[cipher], marker='o', label=cipher)

plt.xlabel("Message Size (bytes)")
plt.ylabel("Efficiency Ratio (MB / sec)")
plt.title("Cipher Efficiency Ratio vs Message Size")
plt.grid(True)
plt.legend()
plt.xticks(message_sizes)
plt.tight_layout()
plt.show()
```

```
64 Byte Tests

AES-128: 64 bytes
Run 1
  Time 0.000182 sec | Mem 0.005476 MB
Run 2
  Time 0.000161 sec | Mem 0.004900 MB
Run 3
  Time 0.000158 sec | Mem 0.004900 MB
Run 4
  Time 0.000158 sec | Mem 0.004900 MB
Run 5
  Time 0.000155 sec | Mem 0.004900 MB
--------------------------------------------------------------------------------
--
AES-128 64 byte | Average Time: 0.000163 sec | Average Memory Used: 0.
 ↪005015 MB
```

```
Ascon: 64 bytes
Run 1
  Time 0.010063 sec | Mem 0.002627 MB
Run 2
  Time 0.010715 sec | Mem 0.002482 MB
Run 3
  Time 0.010458 sec | Mem 0.001620 MB
Run 4
  Time 0.009586 sec | Mem 0.001620 MB
Run 5
  Time 0.009615 sec | Mem 0.001620 MB
--------------------------------------------------------------------------------
--
Ascon 64 byte | Average Time: 0.010087 sec | Average Memory Used: 0.001994␣
 ↪MB

Simon: 64 bytes
Run 1
  Time 0.011743 sec | Mem 0.005044 MB
Run 2
  Time 0.012139 sec | Mem 0.005132 MB
Run 3
  Time 0.012363 sec | Mem 0.005132 MB
Run 4
  Time 0.012068 sec | Mem 0.005004 MB
Run 5
  Time 0.011917 sec | Mem 0.004972 MB
--------------------------------------------------------------------------------
--
Simon 64 byte | Average Time: 0.012046 sec | Average Memory Used: 0.005057␣
 ↪MB

Speck: 64 bytes
Run 1
  Time 0.004073 sec | Mem 0.003548 MB
Run 2
  Time 0.004085 sec | Mem 0.003548 MB
Run 3
  Time 0.004121 sec | Mem 0.003548 MB
```

```
Run 4
  Time 0.005586 sec | Mem 0.008048 MB
Run 5
  Time 0.004047 sec | Mem 0.003276 MB
-------------------------------------------------------------------------------
--
Speck 64 byte | Average Time: 0.004383 sec | Average Memory Used: 0.004394␣
 ↪MB

-------------------------------------------------------------------------------
--
64 byte Winners sorted by Efficiency Ratio
-------------------------------------------------------------------------------
--
Rank   Cipher          Time Used   Memory Used      Efficiency Ratio
-------------------------------------------------------------------------------
--
1      Ascon        0.010087 sec   0.001994 MB       0.197654 MB/sec
2      Simon        0.012046 sec   0.005057 MB       0.419789 MB/sec
3      Speck        0.004383 sec   0.004394 MB       1.002496 MB/sec
4      AES-128      0.000163 sec   0.005015 MB      30.824830 MB/sec

256 Byte Tests

AES-128: 256 bytes
Run 1
  Time 0.000352 sec | Mem 0.005182 MB
Run 2
  Time 0.000160 sec | Mem 0.005341 MB
Run 3
  Time 0.000169 sec | Mem 0.005341 MB
Run 4
  Time 0.000165 sec | Mem 0.005341 MB
Run 5
  Time 0.000161 sec | Mem 0.005917 MB
-------------------------------------------------------------------------------
--
AES-128 256 byte | Average Time: 0.000201 sec | Average Memory Used: 0.
 ↪005424 MB
```

```
Ascon: 256 bytes
Run 1
  Time 0.030553 sec | Mem 0.003283 MB
Run 2
  Time 0.030551 sec | Mem 0.003790 MB
Run 3
  Time 0.029250 sec | Mem 0.002388 MB
Run 4
  Time 0.029933 sec | Mem 0.002388 MB
Run 5
  Time 0.029719 sec | Mem 0.002388 MB
--------------------------------------------------------------------------------
--
Ascon 256 byte | Average Time: 0.030001 sec | Average Memory Used: 0.
 ↪002847 MB

Simon: 256 bytes
Run 1
  Time 0.045225 sec | Mem 0.006129 MB
Run 2
  Time 0.043658 sec | Mem 0.006186 MB
Run 3
  Time 0.045539 sec | Mem 0.006186 MB
Run 4
  Time 0.041762 sec | Mem 0.010481 MB
Run 5
  Time 0.044801 sec | Mem 0.005970 MB
--------------------------------------------------------------------------------
--
Simon 256 byte | Average Time: 0.044197 sec | Average Memory Used: 0.
 ↪006990 MB

Speck: 256 bytes
Run 1
  Time 0.016929 sec | Mem 0.005326 MB
Run 2
  Time 0.018024 sec | Mem 0.004638 MB
Run 3
  Time 0.016372 sec | Mem 0.004638 MB
Run 4
```

```
  Time 0.017126 sec | Mem 0.004638 MB
Run 5
  Time 0.018505 sec | Mem 0.004638 MB
-------------------------------------------------------------------------------
--
Speck 256 byte | Average Time: 0.017391 sec | Average Memory Used: 0.
 ↪004776 MB

-------------------------------------------------------------------------------
--
256 byte Winners sorted by Efficiency Ratio
-------------------------------------------------------------------------------
--
Rank   Cipher          Time Used   Memory Used      Efficiency Ratio
-------------------------------------------------------------------------------
--
1      Ascon        0.030001 sec   0.002847 MB       0.094909 MB/sec
2      Simon        0.044197 sec   0.006990 MB       0.158165 MB/sec
3      Speck        0.017391 sec   0.004776 MB       0.274596 MB/sec
4      AES-128      0.000201 sec   0.005424 MB      26.920103 MB/sec

1024 Byte Tests

AES-128: 1024 bytes
Run 1
  Time 0.000268 sec | Mem 0.007335 MB
Run 2
  Time 0.000184 sec | Mem 0.007615 MB
Run 3
  Time 0.000154 sec | Mem 0.007615 MB
Run 4
  Time 0.000160 sec | Mem 0.008191 MB
Run 5
  Time 0.000164 sec | Mem 0.007615 MB
-------------------------------------------------------------------------------
--
AES-128 1024 byte | Average Time: 0.000186 sec | Average Memory Used: 0.
 ↪007674
MB
```

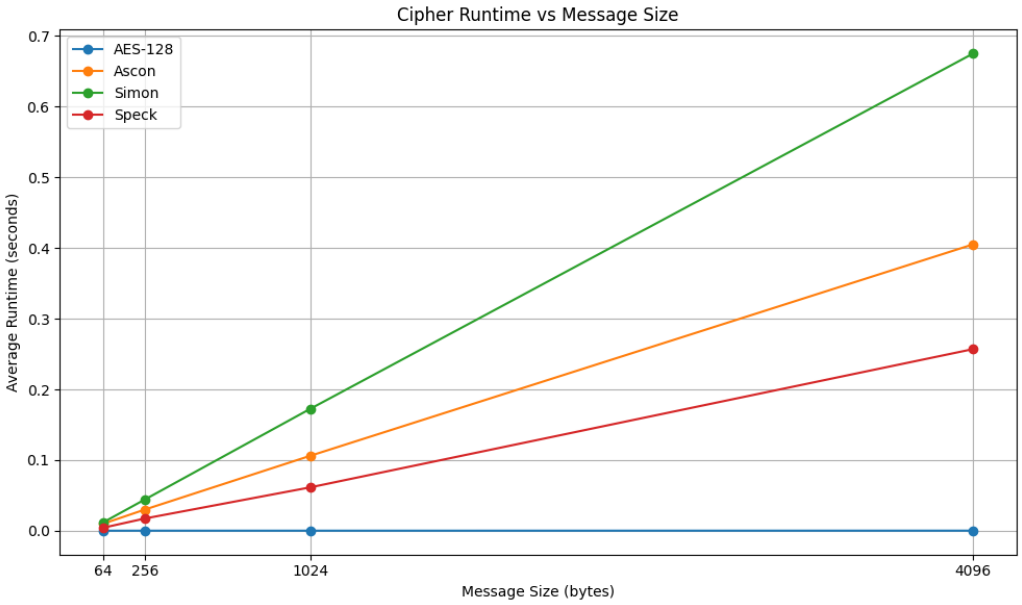
```
Ascon: 1024 bytes
Run 1
  Time 0.104208 sec | Mem 0.005745 MB
Run 2
  Time 0.103090 sec | Mem 0.006026 MB
Run 3
  Time 0.107013 sec | Mem 0.008686 MB
Run 4
  Time 0.105747 sec | Mem 0.006026 MB
Run 5
  Time 0.110922 sec | Mem 0.009830 MB
--------------------------------------------------------------------------------
--
Ascon 1024 byte | Average Time: 0.106196 sec | Average Memory Used: 0.
 ↪007263 MB

Simon: 1024 bytes
Run 1
  Time 0.168203 sec | Mem 0.016960 MB
Run 2
  Time 0.163771 sec | Mem 0.013910 MB
Run 3
  Time 0.186825 sec | Mem 0.016068 MB
Run 4
  Time 0.176170 sec | Mem 0.013008 MB
Run 5
  Time 0.168491 sec | Mem 0.013542 MB
--------------------------------------------------------------------------------
--
Simon 1024 byte | Average Time: 0.172692 sec | Average Memory Used: 0.
 ↪014698 MB

Speck: 1024 bytes
Run 1
  Time 0.060341 sec | Mem 0.011468 MB
Run 2
  Time 0.062946 sec | Mem 0.012419 MB
Run 3
  Time 0.061348 sec | Mem 0.012362 MB
Run 4
```

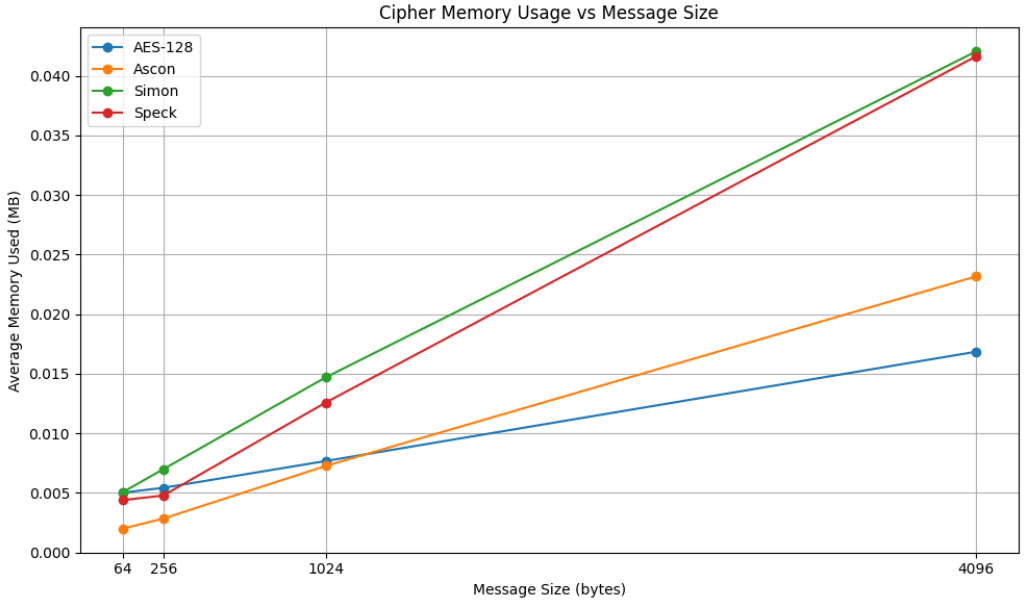
```
  Time 0.061052 sec | Mem 0.011468 MB
Run 5
  Time 0.061956 sec | Mem 0.015208 MB
------------------------------------------------------------------------------
--
Speck 1024 byte | Average Time: 0.061529 sec | Average Memory Used: 0.
 ↪012585 MB

------------------------------------------------------------------------------
--
1024 byte Winners sorted by Efficiency Ratio
------------------------------------------------------------------------------
--
Rank   Cipher          Time Used   Memory Used      Efficiency Ratio
------------------------------------------------------------------------------
--
1      Ascon        0.106196 sec   0.007263 MB       0.068389 MB/sec
2      Simon        0.172692 sec   0.014698 MB       0.085109 MB/sec
3      Speck        0.061529 sec   0.012585 MB       0.204539 MB/sec
4      AES-128      0.000186 sec   0.007674 MB      41.245835 MB/sec

4096 Byte Tests

AES-128: 4096 bytes
Run 1
  Time 0.000215 sec | Mem 0.016727 MB
Run 2
  Time 0.000215 sec | Mem 0.016727 MB
Run 3
  Time 0.000223 sec | Mem 0.017303 MB
Run 4
  Time 0.000226 sec | Mem 0.016727 MB
Run 5
  Time 0.000264 sec | Mem 0.016719 MB
------------------------------------------------------------------------------
--
AES-128 4096 byte | Average Time: 0.000229 sec | Average Memory Used: 0.
 ↪016841
MB
```

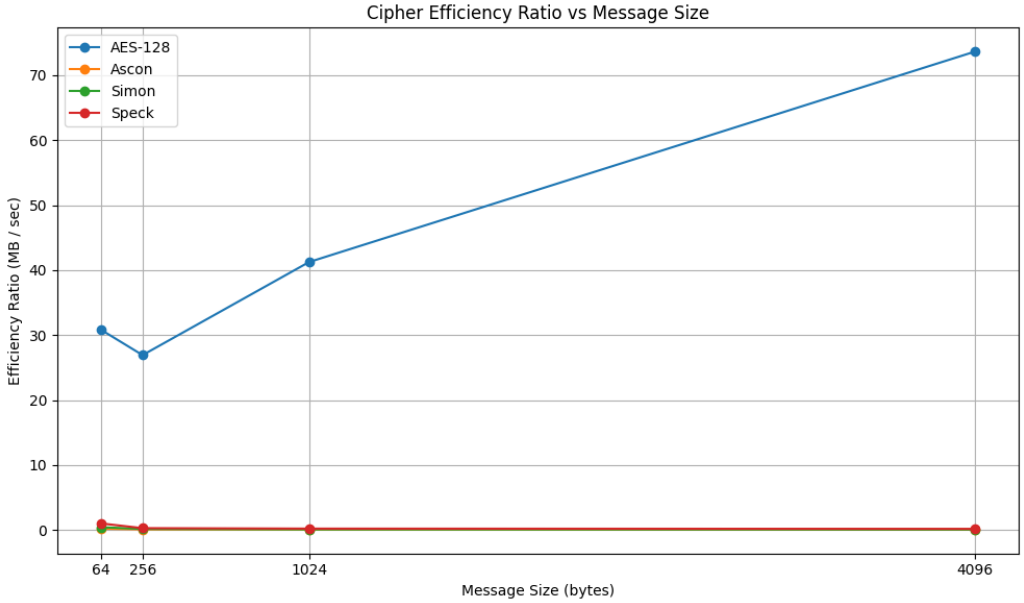
```
Ascon: 4096 bytes
Run 1
  Time 0.399768 sec | Mem 0.021386 MB
Run 2
  Time 0.401453 sec | Mem 0.023602 MB
Run 3
  Time 0.408520 sec | Mem 0.023658 MB
Run 4
  Time 0.412228 sec | Mem 0.023602 MB
Run 5
  Time 0.404337 sec | Mem 0.023546 MB
--------------------------------------------------------------------------------
--
Ascon 4096 byte | Average Time: 0.405261 sec | Average Memory Used: 0.
 ↪023159 MB

Simon: 4096 bytes
Run 1
  Time 0.664167 sec | Mem 0.040720 MB
Run 2
  Time 0.682687 sec | Mem 0.043000 MB
Run 3
  Time 0.666204 sec | Mem 0.041720 MB
Run 4
  Time 0.673488 sec | Mem 0.043000 MB
Run 5
  Time 0.689801 sec | Mem 0.041720 MB
--------------------------------------------------------------------------------
--
Simon 4096 byte | Average Time: 0.675269 sec | Average Memory Used: 0.
 ↪042032 MB

Speck: 4096 bytes
Run 1
  Time 0.256954 sec | Mem 0.039123 MB
Run 2
  Time 0.260896 sec | Mem 0.041405 MB
Run 3
  Time 0.257251 sec | Mem 0.042888 MB
Run 4
```

```
  Time 0.254014 sec | Mem 0.041515 MB
Run 5
  Time 0.256265 sec | Mem 0.043177 MB
------------------------------------------------------------------------------
--
Speck 4096 byte | Average Time: 0.257076 sec | Average Memory Used: 0.
↪041622 MB

------------------------------------------------------------------------------
--
4096 byte Winners sorted by Efficiency Ratio
------------------------------------------------------------------------------
--
Rank   Cipher          Time Used   Memory Used      Efficiency Ratio
------------------------------------------------------------------------------
--
1      Ascon        0.405261 sec   0.023159 MB       0.057145 MB/sec
2      Simon        0.675269 sec   0.042032 MB       0.062245 MB/sec
3      Speck        0.257076 sec   0.041622 MB       0.161904 MB/sec
4      AES-128      0.000229 sec   0.016841 MB      73.649088 MB/sec

******************************************************************************
**
Cipher Totals & Efficiency Ratios (across all sizes, combined runs)
******************************************************************************
**
Cipher            Time Used        Memory Used          Efficiency Ratio
------------------------------------------------------------------------
AES-128        0.003895 sec        0.017303 MB           4.442818 MB/sec
Ascon          2.757729 sec        0.023658 MB           0.008579 MB/sec
Simon          4.521021 sec        0.043000 MB           0.009511 MB/sec
Speck          1.701894 sec        0.043177 MB           0.025370 MB/sec

==============================================================================
==
Efficiency Rankings
==============================================================================
==

Rank   Cipher                Efficiency Ratio
------------------------------------------
```

```
1       Ascon                   0.008579 MB/sec
2       Simon                   0.009511 MB/sec
3       Speck                   0.025370 MB/sec
4       AES-128                 4.442818 MB/sec
```




Cipher Memory Usage vs Message Size
AES-128
Ascon
Simon
Speck
Average Memory Used (MB)
0.000
0.005
0.010
0.015
0.020
0.025
0.030
0.035
0.040
64
256
1024
4096
Message Size (bytes)


Cipher Efficiency Ratio vs Message Size
AES-128
Ascon
Simon
Speck
Efficiency Ratio (MB / sec)
0
10
20
30
40
50
60
70
64
256
1024
4096
Message Size (bytes)

[ ]:

[ ]: