SecureCrypto.py Cheatsheet

```
Quick Start:
import securecrypto as sc
sc.init() # optional if DLL is alongside
```

Constants

```
sc.ALGORITHMS # ('SHA256', 'SHA512')
sc.HMAC_ALGORITHMS # ('HMACSHA256', 'HMACSHA512')
sc.HMAC_ALGORITHMS_MAP # {'sha256': 'HMACSHA256', 'sha512': 'HMACSHA512'}
```

Symmetric (AES w/ PBKDF2)

```
c = sc.encrypt("Hello", "pw") # Base64 (default)
p = sc.decrypt(c, "pw") # -> "Hello"
c_hex = sc.encrypt("Hello", "pw", "hex") # hex
c_raw = sc.encrypt("Hello", "pw", "raw") # bytes
b = sc.encrypt_bytes(b"data", "pw")
plain = sc.decrypt_bytes(b, "pw")
```

Files

```
sc.encrypt_file("in.pdf", "in.pdf.enc", "pw")
sc.decrypt_file("in.pdf.enc", "restored.pdf", "pw")
```

Hybrid (RSA + AES)

```
pub, priv = sc.generate_keypair()
ct = sc.hybrid_encrypt("Top Secret", pub)
pt = sc.hybrid_decrypt(ct, priv)
```

Signing (RSA)

```
sig = sc.sign_string("hello", priv)
ok = sc.verify_string("hello", sig, pub)
sig_file = sc.sign_file_to("doc.bin", priv)
ok2 = sc.verify_file_from("doc.bin", sig_file, pub)
sig_b64 = sc.sign_file("doc.bin", priv)
ok3 = sc.verify_file("doc.bin", sig_b64, pub)
```

Hash & HMAC

```
h = sc.hash_string("abc", sc.ALGORITHMS[0])
hf = sc.hash_file("doc.bin", "SHA512")
hm = sc.hmac("msg", "key", sc.HMAC_ALGORITHMS[0])
ok = sc.hmac_verify("msg", hm, "key", "HMACSHA256")
```

Keys (I/O)

```
sc.export_key_to_file(pub, "public_key.xml")
sc.export_key_to_file(priv, "private_key.xml")
pub2 = sc.import_key_from_file("public_key.xml")
priv2 = sc.import_key_from_file("private_key.xml")
```

Encoding Utility

```
sc.encode_bytes(b"\x01\x02\xff", "base64") # 'AQL/'
sc.encode_bytes(b"\x01\x02\xff", "hex") # '0102ff'
sc.encode_bytes(b"\x01\x02\xff", "raw") # b'\x01\x02\xff'
```

Signature I/O helpers

```
sc.save_signature("doc.bin", sig)
sig_txt = sc.load_signature("doc.bin.sig")
```

Common Pitfalls

```
- DLL must be in the same folder or call sc.init("path/to/SecureCrypto.dll")
- In pythonnet 3.x, out params return tuples (pub, priv = ...)
- encrypt() returns Base64 by default; use "hex" or "raw" if needed
- File encryption format: [16-byte salt][16-byte IV][ciphertext]
```