4-20-2015

Authentication Protocols Based on Advanced Encryption Standard (AES)

Michael Smith

Brandon Barker

Liljana Babinkostova

BOISE STATE UNIVERSITY
COLLEGE OF ENGINEERING

Authentication Protocols Based on Advanced Encryption Standard

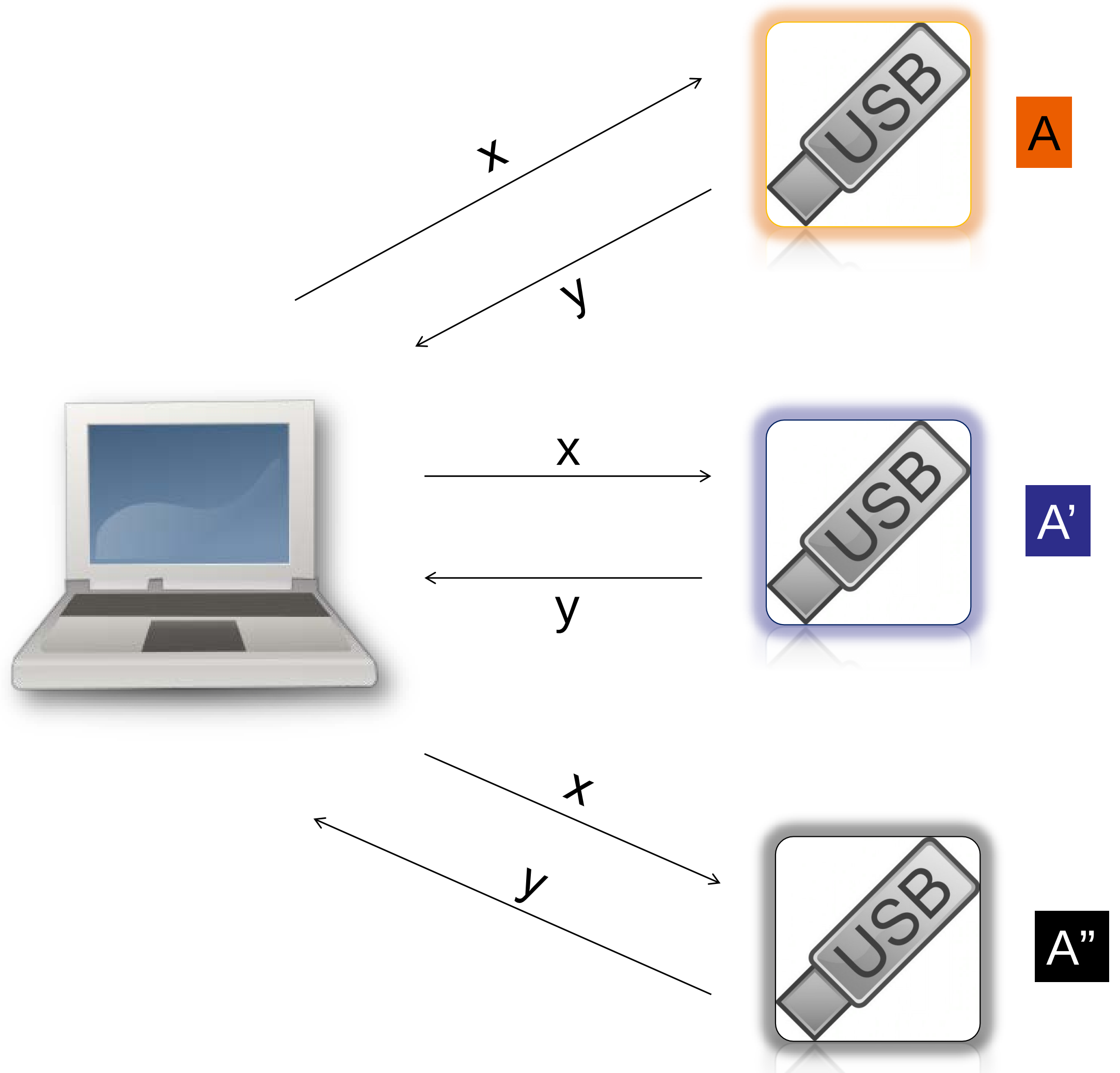
Michael Smith[1], Brandon Barker[1], and Liljana Babinkostova[2], Ph.D
Computer Science Department, [2]Department of Mathematics

Abstract

For the last three decades, hash functions have been an essential element of the cryptography used for securing computers and electronic communications. The SmartDongle is a flash drive produced by MicroWorks and is meant to secure data and assure authorized use of software. In this project, we investigate the security of certain cryptographic techniques used in the current implementation of the SmartDongle's authentication protocol. In particular, we analyze how the use of Merkle-Damgård hash functions based on a simplified version of the Advanced Encryption Standard (AES) can affect the SmartDongle's security. Our study involves extensive computational experimentation and analysis that produced a range of conjectures about the security of the SmartDongle.

Merkle-Damgård Schema

The Merkle-Damgård Schema is a function that can be used in producing a collision-resistant hash function. The message (M) is split into blocks of a fixed size which are then computed through AES individually. The output from the previous computation is then used as the key in the following computation, until all blocks have been computed. The final output is then the hashed text.

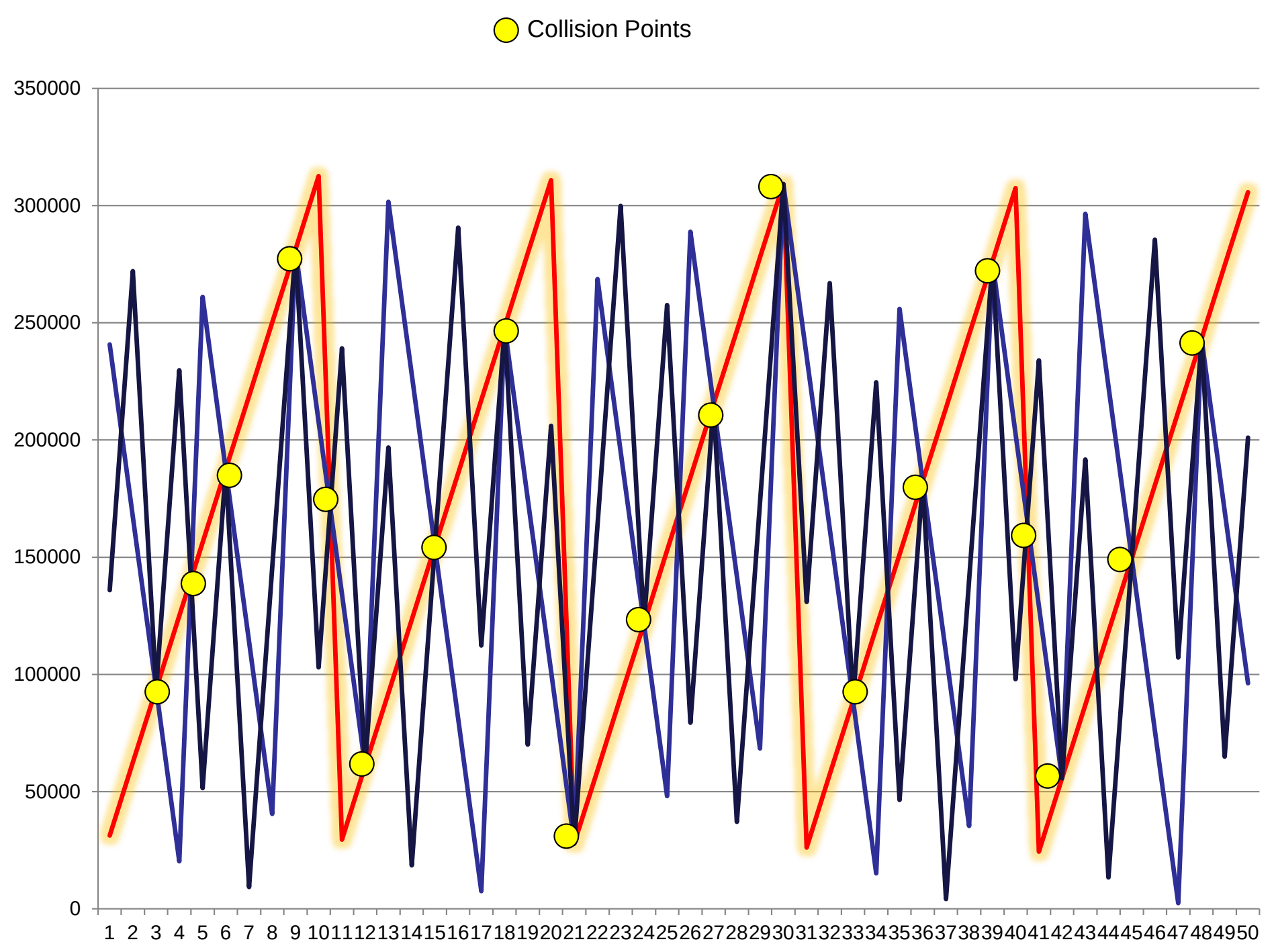


Project Summary

We investigated the security of the SmartDongle Device produced by MicroWorks. The fundamental security idea is that the device computes and communicates 'x, y' pairs from a given equation. The security was based on the computational difficulty of determining the parameter 'A' of the equation $y = Ax \bmod n$ given these 'x' and 'y' values.

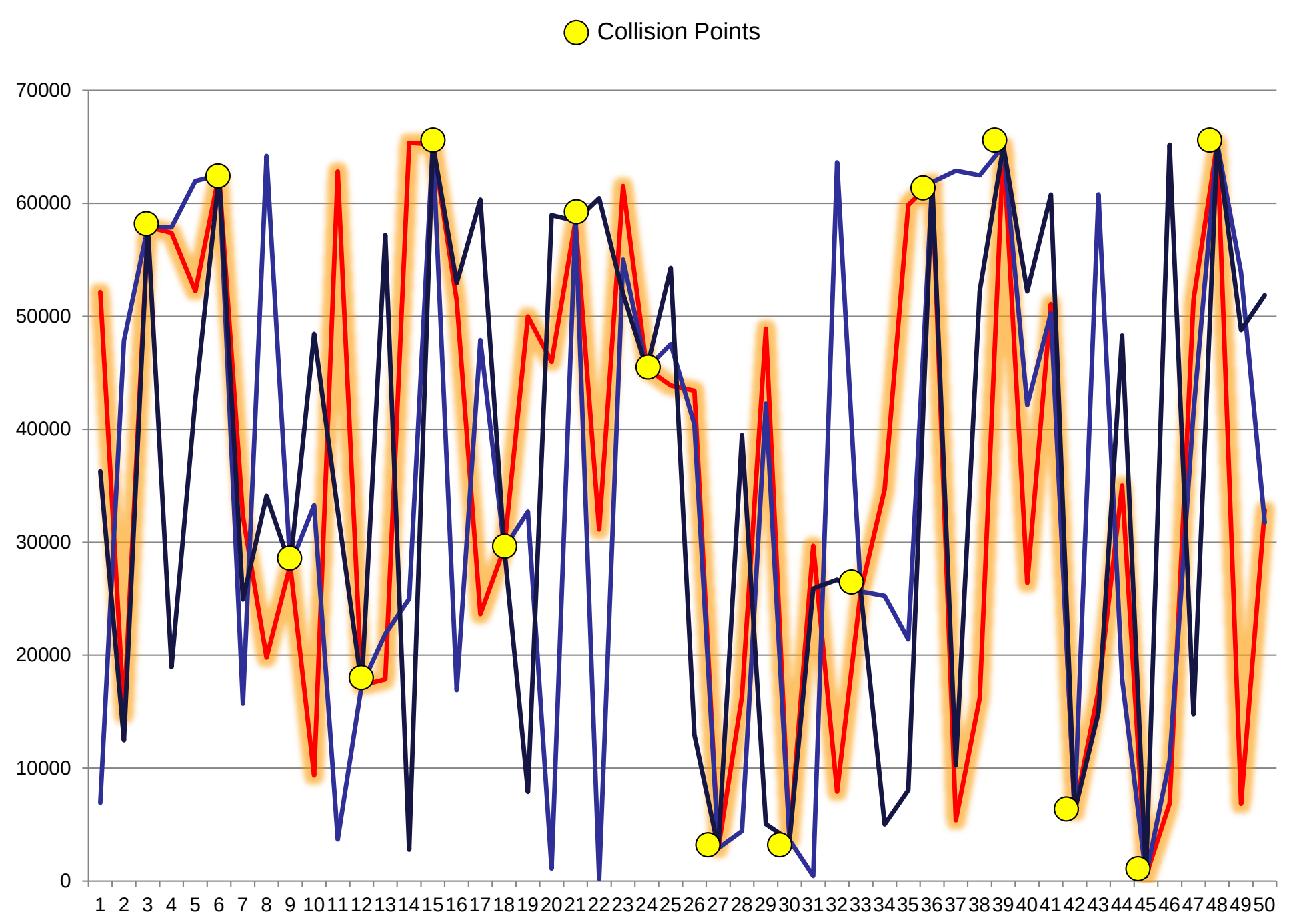
- We developed Java software to compute the possible 'A' values that would satisfy the given 'x' and 'y' values.
- This software would export large amounts of data to Excel which would then be used to create graphs of the 'A' values.
- Using these graphs, we discovered patterns that assisted in finding the correct 'A' values when they were unknown
- We created a hash function based on AES and the Merkle-Damgård Schema to hash the 'y' values.
- We developed software that would compute the values of the original protocol and the hashed protocol
- These values would then be exported to Excel and graphed for the purpose of comparison.
- The pattern previously followed by the correct 'A' value was disguised, though all possible 'A' values converged where 'x' was equal to a factor of 'n'.
- Rehashing 'y' a number of times based on A *modulo* the factor of n eliminated these collisions.

Regular Protocol



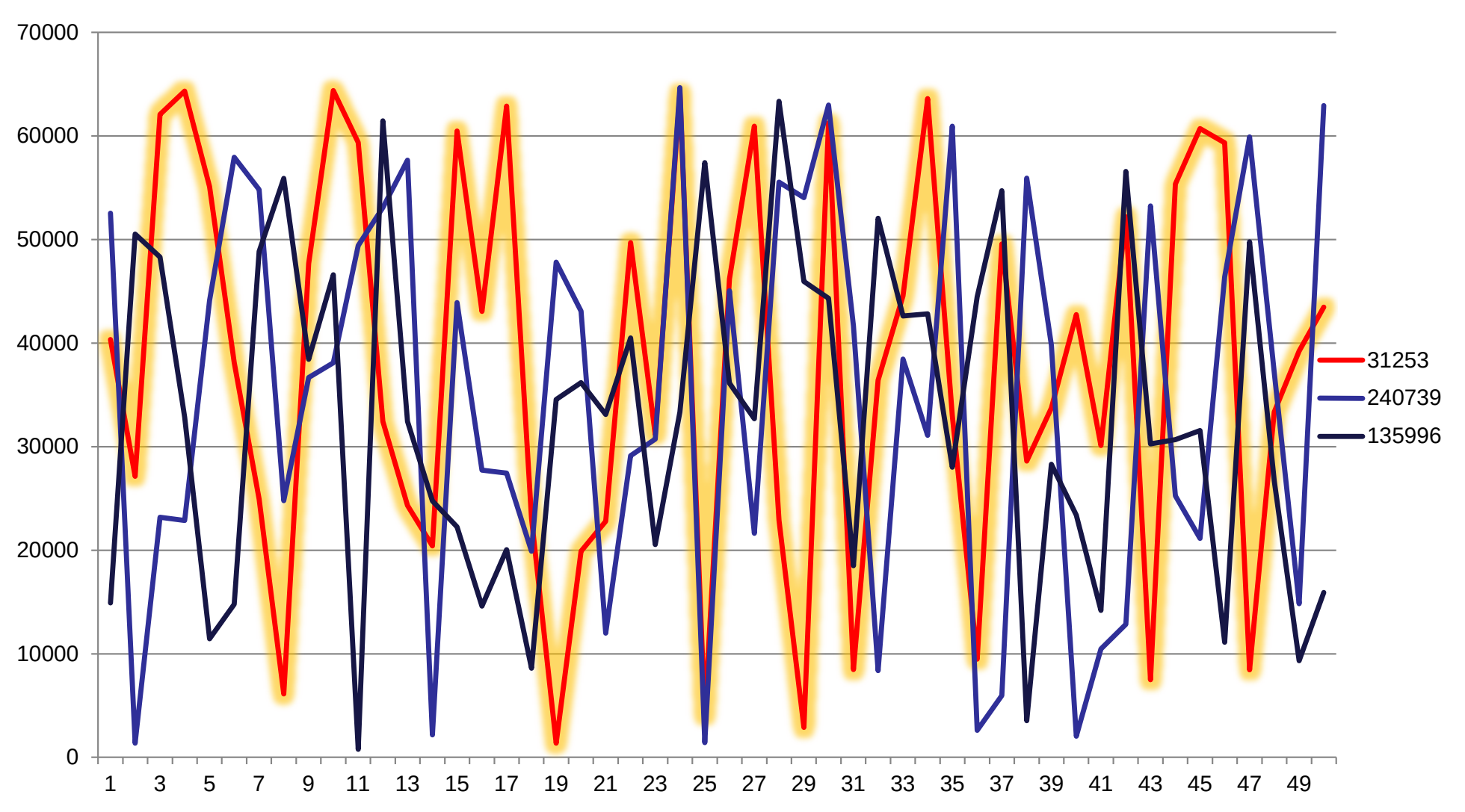
The pattern of the regular protocol is easily identifiable and vulnerable to attack due to its predictability with just a small amount of data collection. There are also several collision points, making it possible for imposter dongles to authenticate.

AES Hashed Protocol



The pattern of 'A' value is disguised by AES encryption of the 'y' value. However, all possible 'A' values converge to the same value where x is equal to a factor of 'n.' This is an algebraic vulnerability of the function that exists with or without hashing any values.

Main Result



Though some encryption schemes run into a problem of reversing the encryption in repeated applications of the encryption, AES does not appear to suffer from this, allowing us to eliminate collisions by rehashing 'y' a number of times based on a modulus of A that is equal to the factor of n (in this example, A *mod 3*). We successfully disguise A while ensuring that 'y' is not hashed to the same value at a given 'x' for any other possible A.

References

1. L. Babinkostova, K. W. Bombardier, M. C. Cole, T. A. Morrell, C. B. Scott, *Algebraic properties of generalized Rijndael-like ciphers*, **Groups Complexity Cryptology** 6(1): 37-54 (2014).
2. I. Damgård, *A Design Principle for Hash Functions*, **Advances in Cryptology - CRYPTO '89 Proceedings, Lecture Notes in Computer Science Vol. 435**, G. Brassard, ed, Springer-Verlag, 416-427 (1989).
3. R.C. Merkle. *A Certified Digital Signature*, **Advances in Cryptology - CRYPTO '89 Proceedings, Lecture Notes in Computer Science Vol. 435**, G. Brassard, ed, Springer-Verlag, 218-238 (1989)
4. R. C.-W. Phan, *Mini Advanced Encryption Standard (Mini-AES): A Testbed for Cryptanalysis Students*, **Cryptologia** 26:4, 283-306 (2002)

Future Work

The computation time for rehashing with the large primes that would typically be used as factors of n could be large enough to bring the practicality of this into question. However, other functions might be used to generate unique keys for rehashing and drastically reduce the amount of times a y values would need to be rehashed to avoid collisions.

ACKNOWLEDGEMENTS

This research was supported by the NSF Idaho Science Talent Expansion Program (STEP) under the grant DUE 0856815 and the Student Research Program at Boise State University.