https://doi.org/10.48009/2_iis_2016_193-200

*Issues in Information Systems*
*Volume 17, Issue II, pp. 193-200, 2016*

# DETERMINING RSA PRIVATE KEY USING MICROSOFT EXCEL SOLVER

***Abhijit Sen, Kwantlen Polytechnic University, abhijit.sen@kpu.ca***

## ABSTRACT

*Microsoft Excel offers number of data analysis tools commonly known as what if analysis. These tools are extensively used to solve many business problems. However in Engineering and Computing Science these easy to use features are seldom used to solve appropriate problems. Instead most of the problems are solved using specific tools such as MATLAB or specific Computer programs. In situation where Microsoft Excel is used, only rudimentary features such as formula, mathematical functions are utilized. The Rivest-Shamir-Adleman (RSA) algorithm is one of the most well-known and secure public-key encryption methods used for secured data transmission. To determine the private key for RSA algorithm one has to use complex mathematical calculations. Many students of Introductory Computer Security courses without appropriate Mathematical background find it difficult to use RSA algorithm to determine appropriate solution for private decryption key. In this paper author attempts to show how RSA algorithm to determine private key can be modelled for Microsoft Excel Spread sheet, and how Excel Solver can easily and effectively be used subsequently to determine private keys for RSA algorithm. The author uses several examples to show viability of usage of EXCEL solver to quickly find private RSA keys. The author finds that the EXCEL Solver can be easily used in the class room to demonstrate how RSA algorithm can be used. The author also discusses the limitations and practical issues related to using EXCEL solver.*

**Keywords:** EXCEL Solver, Private Key, Public Key, RSA

## INTRODUCTION

Spreadsheets software is widely used to formulate optimization models in business to determine how limited resources can be utilized to achieve the major objective or stated goals of the systems. In particular, Excel Solver is used extensively for solving and analyzing optimization models. Solver is a Microsoft Excel add-in which solves problems by modifying a cell or group of cells to achieve specific goal, such as minimizing, maximizing or attaining some target value. The solution is achieved by modifying a number of input cells according to a set of user defined criteria or constraints.

The objective of this paper is to show how one can use Excel solver to determine private decryption key using well known RSA algorithm [Forouzan, 2013]. RSA is a widely used public-key cryptographic algorithm involving two keys- public and private key. Public key is made public and is used for encrypting messages, whereas private key is held secret and is used for decrypting messages. Private and public keys are mathematically related. One of the key elements of RSA Algorithm is generation of private key given a public key.

Many students have difficulties in understanding the basis of RSA algorithm because of mathematics involved. The mathematics is especially challenging for students in Introductory Cryptography course. In the pursuit of finding simpler ways of teaching RSA algorithm, the author researched on different software tools such as MATLAB, and programming languages such as C#, Java, etc. Because of simplicity and ease of use Microsoft EXCEL is selected as a possible candidate for Cryptography course. Brief introduction to Excel Solver in class is sufficient to solve this problem. In this paper, the author has demonstrated a novel way of the using Microsoft EXCEL Add-in Excel Solver to generate RSA keys.

## LITERATURE SURVEY

The use of spreadsheet modeling and Excel Solver in solving linear and nonlinear programming problems in an introductory Operations Research course is discussed in literatures [Chandrakantha, 2011]. Application of Solver through examples to different applications areas such as manufacturing, transportation, financial planning, capital investment, shortest path determination, and Task Assignment is demonstrated in literatures [Chandrakantha, 2011]. The fundamental concepts of optimization, and usage of Microsoft EXCE techniques to specific application areas of business and marketing are detailed in [Winston, 2014].

In specialized areas of computer security where EXCEL is applied to solve specific problems, one finds only the usage of basic functionalities of EXCEL such as use of simple mathematical formulas and operations.

In this paper the author investigates the use of EXCEL Solver to calculate private decryption key and demonstrates the feasibility of using this tool to compute the key easily. This tool is easy to use and will be suitable for users without substantial mathematical background to implement a solution.

## OVERVIEW OF RSA ALGORITHM

RSA is a cryptosystem for public-key encryption widely used for securing sensitive data, being transmitted over insecure channel such as the Internet [Forouzan, 2013]. In RSA two different keys are used, one public and the other private. The public key can be shared with everyone, whereas the private key must be kept secret. These two keys are mathematically related. The most complex part of RSA cryptography is the public and the private key-generation algorithm, which is summarized below.

**RSA Key Generation Steps [**Forouzan, 2013**]:**

Each user generates a public/private key using the following well known RSA key generation Algorithm [Forouzan, 2013]:

- Select two large primes at random: p, q such that $p \neq q$
- Compute $n = p * q$
- Compute $ø(n) = (p-1) * (q-1)$
- Select encryption key e such that $1 < e < ø(n)$, $\gcd(e, ø(n)) = 1$
- Find decryption key d such that $e * d = 1 \bmod ø(n)$ and $0 \leq d \leq n$
  Most common algorithm used to calculate d is Extended Euclidean Algorithm [Forouzan, 2013]
- Publish their public encryption key {e, n}
- Keep secret private decryption key: {d, n}

**Encryption/ Decryption Steps [**Forouzan, 2013**]:**

The encrypted text $C = M^e \bmod n$, where M is the plaintext message.
The Plaintext (decrypted) text $M = C^d \bmod n$

## SPREADSHEET MODELING OF RSA KEY GENERATION ALGORITHM

Solver is part of a suite of commands in EXCEL sometimes called what-if analysis tools. With Solver, one can find an optimal (maximum or minimum) value for a formula in one cell — called the objective cell — subject to constraints, or limits, on the values of other formula cells on a worksheet. Solver works with a group of cells, called decision variables or simply variable cells that participate in computing the formulas in the objective and constraint cells. Solver adjusts the values in the decision variable cells to satisfy the limits on constraint cells and produce the result you want for the objective cell. In this section, the author shows how the key generation algorithm can be modelled using EXCEL Solver.

**Problem Statement**: Find the private decryption key using RSA Algorithm.

**Objective Function:**

In this section ,we show how the theoretical foundations of RSA algorithm as described in the previous section is translated to actual formulae and constrains so that problem can be implemented in EXCEL Solver. Objective is to find the value of private decryption key d for a given value of public encryption key e such that the following rule is satisfied:

$e * d = 1 \bmod ø(n)$
Constraints are the following:

$0 <= d <=n$
$1 < e < ø(n)$,
$\gcd (e, ø(n)) = 1$

The above optimization problem can be represented using the following spreadsheet model:

Cell A7: value of the first Prime number p
Cell B7: value of the second Prime number q
Cell C7: value of n using formula n = p * q
Cell D7: value of ø(n) using formula ø(n) = (p-1) * (q-1)
Cell E7: selected value of public encryption key e
Cell F7: Greatest Common Divisor gcd (e, ø(n)) and must be one 1
Cell G7: decision variable d the private key to be determined
Cell H7; represents the objective function e * d * mod ø (n) where e, d, and ø(n) are taken from cells E7, G7, and D7 respectively

Constraints are:

$E$7 <= $D$7 -1
$E$7 >= 2
$F$7 =1
$G$7 <= $C$7
$G$7: integer
$G$7 >= 0

The complete set of constraints, target cell (objective function cell), variable cells (changing cells) and the value of the objective function are identified in the Solver parameters box.

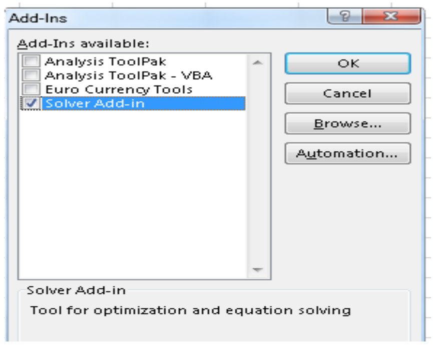
## DETERMINING PRIVATE KEYS USING MICROSOFT EXCEL SOLVER

The following step shows how to invoke Microsoft EXCEL Solver to determine Private Decryption Key using the previously stated Objective function and appropriate constraints.

First enable solver by using the following steps:

- Click the File tab, and then click Options below the Excel tab.
- In the Excel Options dialog box, click Add-Ins.
- In the Manage drop-down box, select Excel Add-ins, and then click Go.

- In the Add-Ins dialog box, select Solver Add-in, and then click OK.



After the Solver add-in is enabled, Excel will auto-install the Add-in if it is not already installed, and the Solver command will be added to the Analysis group on the Data tab in the ribbon.

To apply solver we need to define a series of requirements, rules and constraints. These requirements, rules and constraints guide solver and set limits which allow solver to quickly narrow in on the answer.

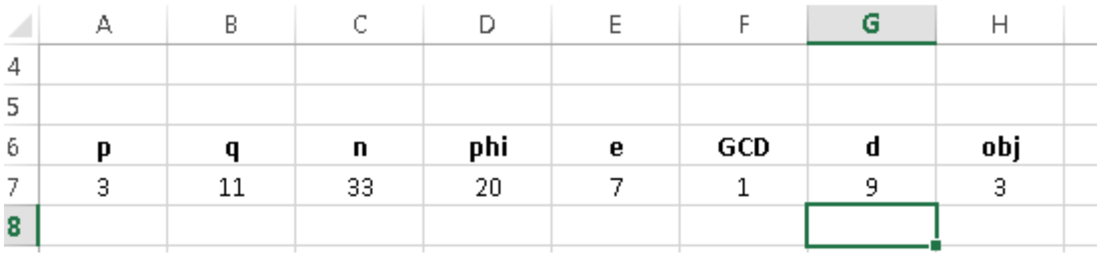
- In the **Set Objective** box, enter a cell reference ($H$7) for the objective cell that contains the formula. **Set Objective** box is the cell which you are trying to solve the problem for.
- To: section defines what we want to do with our **Set Objective cell**. Click **Min/Max** if you want the value of the objective cell to be as small/large as possible. If you want the objective cell to be a certain value, click **Value of**, and then type the value (1) in the box.
- In the **By Changing Variable Cells** box, enter reference for each decision variable cell range ($G$7). **Changing Variable Cells**: refers to the cells which will be modified by Solver to try and solve the problem.
- In the **Subject to the Constraints** box, enter any constraints that you want to apply. Constraints are the rules which define the limits of the possible solutions to the problem
- To accept the constraint and add another, click **Add**.
- To accept the constraint and return to the **Solver Parameters** dialog box, click **OK**.
- Select a **Solving Method** GRG Nonlinear
- Click **Solve** and to keep the solution values on the worksheet, click **Keep Solver Solution**, in the **Solver Results** dialog box. If Solver has found a solution, the worksheet cells will be changed to show the solution

In the following section we will show the results of executing Excel Solver to find the private key d.

## RESULTS

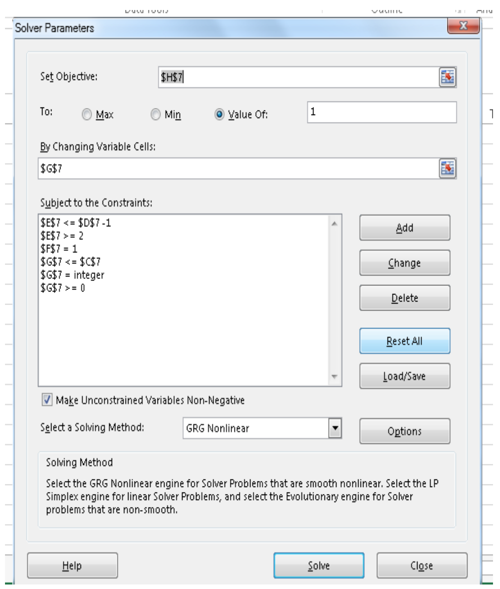
Figure 1 shows the initial values of p and q. The values of n and phi (ø(n)) are calculated using appropriate formula. User selects a specific value of e (7 in our example). Greatest Common Divisor gcd(e,ø(n)) and must be one 1 as shown in GCD column. So encryption key is a valid key. We select an initial value of decryption key d to be 7 as shown. Spreadsheet computes e*d mod (ø(n)) in obj column. As seen from obj column the result is 3 which indicates dis not the correct value for decryption key.

To get the valid solution, we will invoke EXCEL Solver which contains the objectives, rules and constraints. Once Solver is executed the correct encryption key is found as shown in Figure 3. The value of decryption key d is 3 which satisfies the objective and meets all constraints as shown Figure 4.



|  | A | B | C | D | E | F | G | H |
|---|---|---|---|---|---|---|---|---|
| 4 |  |  |  |  |  |  |  |  |
| 5 |  |  |  |  |  |  |  |  |
| 6 | **p** | **q** | **n** | **phi** | **e** | **GCD** | **d** | **obj** |
| 7 | 3 | 11 | 33 | 20 | 7 | 1 | 9 | 3 |
| 8 |  |  |  |  |  |  |  |  |

**Figure 1.** Initial Implementation of Spread Sheet



Solver Parameters

Set Objective: $H$7

To: Max Min [✓ Value Of:] 1

By Changing Variable Cells: $G$7

Subject to the Constraints:

```
$E$7 <= $D$7 -1
$E$7 >= 2
$F$7 = 1
$G$7 <= $C$7
$G$7 = integer
$G$7 >= 0
```

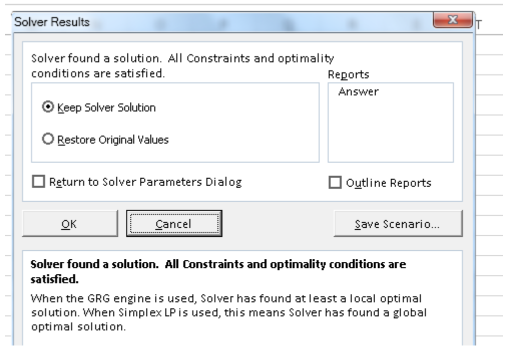
[✓] Make Unconstrained Variables Non-Negative

Select a Solving Method: GRG Nonlinear

Solving Method

Select the GRG Nonlinear engine for Solver Problems that are smooth nonlinear. Select the LP Simplex engine for linear Solver Problems, and select the Evolutionary engine for Solver problems that are non-smooth.

**Figure 2.** Solver Implementation with All the Constraints



Solver Results

Solver found a solution. All Constraints and optimality conditions are satisfied.

[✓ Keep Solver Solution]

Restore Original Values

Reports: Answer

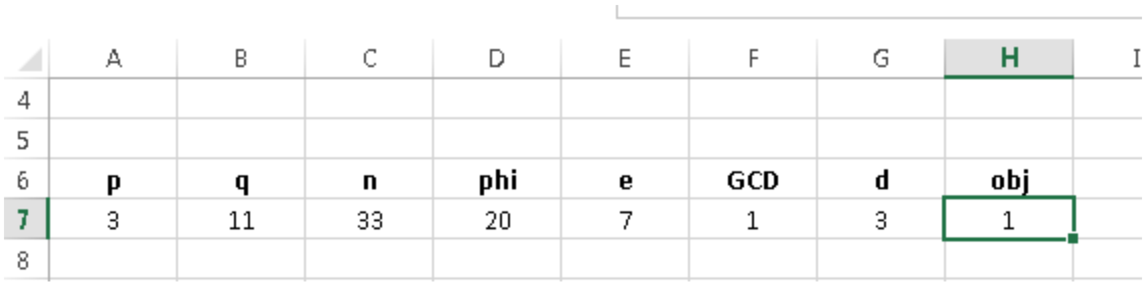
☐ Return to Solver Parameters Dialog

☐ Outline Reports

**Solver found a solution. All Constraints and optimality conditions are satisfied.**

When the GRG engine is used, Solver has found at least a local optimal solution. When Simplex LP is used, this means Solver has found a global optimal solution.

**Figure 3.** Solver Solution Successful



| | A | B | C | D | E | F | G | H |
|---|---|---|---|---|---|---|---|---|
| 4 | | | | | | | | |
| 5 | | | | | | | | |
| 6 | **p** | **q** | **n** | **phi** | **e** | **GCD** | **d** | **obj** |
| 7 | 3 | 11 | 33 | 20 | 7 | 1 | 3 | 1 |
| 8 | | | | | | | | |

**Figure 4.** Final Implementation of Spread Sheet with Optimal Solution

The final value of d is found to be 3 whereas initial value of d was 9. Using the values of e,d,n from Figure 4 above, one can generate ciphertext C from Message M. Assume plaintext message M is number 9

Ciphertext C generated by sender using public key {e, n} $=M^e$ mod n = $9^7$ mod 33 = 15

Receiver decrypts Ciphertext C using private key {d, n} to generate Plaintext M = $C^d$ mod n = $15^3$ mod 33 = 9

The tests are conducted using different values of p, q, e and d. The following seven tests are done as shown in Figure 5.

| Tests | p | q | n | ø(n) | e | GCD | d | obj |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| 1 | 3 | 11 | 33 | 20 | 7 | 1 | 9 | 3 |
| 2 | 3 | 11 | 33 | 20 | 7 | 1 | 6 | 2 |
| 3 | 11 | 13 | 143 | 120 | 7 | 1 | 111 | 57 |
| 4 | 11 | 13 | 143 | 120 | 7 | 1 | 31 | 97 |
| 5 | 137 | 157 | 21509 | 21216 | 59 | 1 | 10000 | 17168 |
| 6 | 257 | 337 | 86609 | 86016 | 17 | 1 | 10000 | 83984 |
| 7 | 257 | 337 | 86609 | 86016 | 17 | 1 | 20000 | 81952 |

**Figure 5.** Initial Values with Arbitrary d

Using the initial values from Figure 5 the following table shows the decryption key d for each of the test cases corresponding to Figure 6 as generated by the EXCEL Solver

| Tests | p | q | n | ø(n) | e | GCD | **d** | obj |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| 1 | 3 | 11 | 33 | 20 | 7 | 1 | **3** | 1 |
| 2 | 3 | 11 | 33 | 20 | 7 | 1 | **3** | 1 |
| 3 | 11 | 13 | 143 | 120 | 7 | 1 | **103** | 1 |
| 4 | 11 | 13 | 143 | 120 | 7 | 1 | **103** | 1 |
| 5 | 137 | 157 | 21509 | 21216 | 59 | 1 | **11507** | 1 |
| 6 | 257 | 337 | 86609 | 86016 | 17 | 1 | **65777** | 1 |
| 7 | 257 | 337 | 86609 | 86016 | 17 | 1 | **65777** | 1 |

**Figure 6.** Solver Output: Final Values with Valid Decryption Key d

Figures 5 and 6 also show how different initial value of d was resulted in correct solution for d. As is seen from Figure 6, for each case the objective function value of 1 is met.

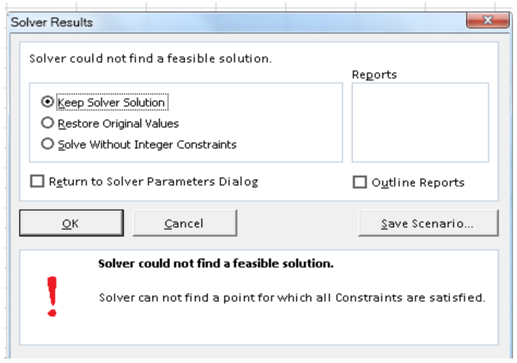
As can be seen for test cases (1, 2), (3, 4), (6, 7) in Figure 5 different values of d are used as initial values, However solver generated the same decryption keys : 3 for case (1,2), 103 for case (3,4), and 65777 for case (6,7) as shown in Figure 6.
These examples demonstrate the viability of using EXCEL Solver to generate private key for given valid public encryption key.

## ISSUES AND LIMITATIONS OF EXCEL SOLVER

The EXCEL Solver may not come up with correct solution in every situation. With the initial value of d = 28600 as shown in Test 8 Figure 7, solver could not find a feasible solution as shown in Figure 8.

| Test | p | q | n | ø(n) | e | GCD | d | obj |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| 8 | 137 | 311 | 42607 | 42160 | 53 | 1 | 28600 | 40200 |

**Figure 7.** No Solution Found for d = 28600


Solver Results

Solver could not find a feasible solution.

Reports

(● Keep Solver Solution)
○ Restore Original Values
○ Solve Without Integer Constraints

☐ Return to Solver Parameters Dialog
☐ Outline Reports

OK | Cancel | Save Scenario...

**Solver could not find a feasible solution.**

Solver can not find a point for which all Constraints are satisfied.

Figure 8. No Feasible Solution

If error occurs, and if there are no errors in modelling it is suggested [12] that one should reinvoke the solver with a new initial value. By changing d from 28600 to 42607 as shown in Test 9 Figure 9, solver correctly generated a feasible solution d= 28637 as shown in Figure 10. .

| Test | p | q | n | ø(n) | e | GCD | d | obj |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| 9 | 137 | 311 | 42607 | 42160 | 53 | 1 | 42607 | 40200 |

**Figure 9.** Value of d Changed to 42607 from 28600

| Test | p | q | n | ø(n | e | GCD | d | obj |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| 10 | 137 | 311 | 42607 | 42160 | 53 | 1 | 28637 | 1 |

**Figure 10.** Feasible Solution Attained by Changing Initial Value of d

It is good Solver practice to test Solver's answer using slightly different numbers and ensuring if the answer makes sense. The issues and limitations of EXCEL Solver are discussed in details [Evans, 2008], which also suggested possible remedial steps. Also, it is practically impossible to implement the RSA algorithm in a spreadsheet using very large integer values. Excel's mod(x, n) does not work if x > 2^32 - 1 (i.e., longer than 32 bits), although one can write software routines to handle large integers. However EXCEL is a very simple tool that can be used to demonstrate the principles of RSA using numbers that EXCEL can handle. The general complexity of convergence to optimal solution is discussed in [McCullough, 1999].

## CONCLUSIONS

This paper discusses how some of the Microsoft Excel features so widely used in business applications can also be easily used Cryptography. The author choose widely used RSA algorithm and demonstrates Excel's Solver techniques to determine private key. The users in many situations do not have to be knowledgeable in sophisticated programs like MATLAB and other Simulation software to solve many Engineering and Computer Science problems. Many students have difficulties of understanding introductory cryptography concepts. Instructors teaching Introductory courses in Computer Security and Cryptography courses can save valuable time by using simple software like Microsoft EXCEL. EXCEL offers a quick way to build a model, and instructors can easily explain the model and its results to students who struggle with complexities and rigors of underlying mathematical principles. Application programs written in a programming language and sophisticated simulation language may offer higher performance and flexibility, but for general user community interested in the applied aspect of cryptography, it may be worthwhile to try out simple easily available tools to convey the concepts.

## REFERENCES

Chandrakantha, L. (2011). Using Excel Solver in Optimization Problems. Proceedings of the *Twenty-third Annual International Conference on Technology in Collegiate Mathematics,* Denver, Colorado, March 17-20, pp. 42-49. Available: http://archives.math.utk.edu/ICTCM/i/23/C006.html

Evans, J.R. (2008). Teaching Note—Some Practical Issues with Excel Solver: Lessons for Students and Instructors. INFORMS. *Transactions on Education 8*(2), pp. 89-95. Available: http://dx.doi.org/10.1287/ited.1070.0006

Forouzan, Behrouz A. (2013). Cryptography and Network Security, McGraw-Hill. Available: http://www.mheducation.com/highered/product.M0073376221.html

McCullough, B. D., and Vinod, H. D. (1999). The Numerical Reliability of Econometric Software. *Journal of Economic Literature, 37*(2), pp. 633-665

Winston, W. (2014). 'Marketing Analytics: Data-Driven Techniques with Microsoft Excel', Wiley. Available: http://ca.wiley.com/WileyCDA/WileyTitle/productCd-111837343X.html

Winston, W. (2014). 'Microsoft Excel 2013 Data Analysis and Business Modeling', Microsoft Press. Available: https://www.microsoftpressstore.com/store/microsoft-excel-2013-data-analysis-and-business-modeling-9780735669130