Reverse Engineering challenges

Contrived by Dennis Yurichev (yurichev.com).

The website has been inspired by Project Euler and "the matasano crypto challenges".

All challenges/exercises/problems/tasks

1; 2; 3; 4; 5; 6; 7; 8; 9; 10; 11; 12; 13; 14; 15; 16; 17; 18; 19; 20; 21; 22; 23; 24; 25 (black boxes); 26; 27; 28; 29 (obfusction); 30; 31; 32; 33; 34; 35; 36; 37; 38; 39; 40; 41; 42 (unknown cryptoalgorithm); 43; 44; 45 (simple copyprotection); 46 (toy-level web server); 47 (broken data compression algorithm); 48; 49; 50 (4-byte XOR mask); 51 (stack); 52 (stack); 53; 54 (LOOP instruction); 55 (simple patching exercise); 56; 57; 58; 59; 60; 61; 62 (array); 63 (array); 64 (array); 65 (array); 66 (array); 67 (bit field); 68 (bit field); 69 (bit field); 70 (bit field); 71 (structure); 72 (structure); 73.

By level

- Level 1: 1; 2; 3; 4; 5; 6; 7; 8; 9; 10; 11; 12; 13; 14; 15; 16; 17; 18; 19; 20; 21; 22; 23; 26; 27; 28; 30; 31; 32; 33; 34; 35; 36; 37; 38; 39; 41; 43; 48; 49; 51 (stack); 52 (stack); 53; 54 (LOOP instruction); 55 (simple patching exercise); 56; 57; 58; 59; 60; 61; 62 (array); 63 (array); 64 (array); 65 (array); 66 (array); 67 (bit field); 68 (bit field); 69 (bit field); 70 (bit field); 71 (structure); 72 (structure).
- Level 2: 24; 29 (obfusction); 40; 42 (unknown cryptoalgorithm); 44; 45 (simple copyprotection); 46 (toy-level web server).
- Level 3: 25 (black boxes); 47 (broken data compression algorithm).

Disctinction between levels is very blurred and not very strict, so don't rely on them fully. But feel free to send me recommendations about them. I would love to hear comments like "it was so easy/hard" or "I've spent 2 hours for this", so I can gather some statistics about exercises and promote/demote them.

By type

- Piece of assembly listing: 1; 2; 3; 4; 5; 6; 7; 8; 9; 10; 11; 12; 13; 15; 16; 17; 18; 19; 20; 21; 22; 23; 27; 28; 31; 32; 33; 34; 35; 36; 48; 49; 52 (stack); 56; 58; 59; 60; 61; 62 (array); 63 (array); 64 (array); 65 (array); 66 (array); 67 (bit field); 68 (bit field); 69 (bit field); 70 (bit field); 72 (structure).

  There are two questions almost for every exercise like this, if not specified otherwise. 1) What does this function do? Try to give one-sentence answer. 2) Rewrite this function into C/C++.

- Executable file: 12; 24; 29 (obfusction); 30; 38; 39; 40; 41; 42 (unknown cryptoalgorithm); 43; 44; 45 (simple copyprotection); 46 (toy-level web server); 47 (broken data compression algorithm); 71 (structure).

By architecture

- X86: 2; 3; 4; 27; 30; 31; 32; 33; 35; 36; 37; 38; 40; 41; 42 (unknown cryptoalgorithm); 44; 45 (simple copyprotection); 46 (toy-level web server); 47 (broken data compression algorithm); 48; 52 (stack); 54 (LOOP instruction); 56; 58; 59; 60; 62 (array); 63 (array); 64 (array); 65 (array); 66 (array); 67 (bit field); 68 (bit field); 69 (bit field); 70 (bit field); 71 (structure); 72 (structure).
- X64: 1; 5; 6; 7; 8; 9; 10; 11; 12; 13; 15; 16; 17; 18; 19; 20; 21; 22; 23; 24; 28; 31; 36; 38; 41; 46 (toy-level web server); 47 (broken data compression algorithm); 49; 54 (LOOP instruction); 73.

- X86/X64 SSE: 13; 15; 16; 20; 31; 36.
- X86/X64 FPU: 20; 31; 36; 60; 61; 62 (array); 72 (structure).
- MIPS: 2; 3; 4; 5; 6; 7; 8; 9; 10; 11; 19; 20; 21; 22; 24; 28; 30; 32; 33; 34; 35; 36; 37; 39; 40; 42 (unknown cryptoalgorithm); 43; 45 (simple copyprotection); 52 (stack); 56; 58; 59; 61; 62 (array); 63 (array); 64 (array); 65 (array); 66 (array); 67 (bit field); 68 (bit field); 70 (bit field); 71 (structure); 72 (structure).
- ARM: 2; 3; 4; 5; 6; 7; 8; 10; 11; 19; 20; 21; 22; 28; 32; 33; 34; 35; 36; 37; 41; 52 (stack); 56; 58; 59; 62 (array); 63 (array); 64 (array); 65 (array); 66 (array); 67 (bit field); 68 (bit field); 70 (bit field); 72 (structure).
- ARM64: 2; 3; 4; 5; 6; 7; 8; 9; 10; 11; 18; 19; 20; 21; 22; 23; 27; 28; 32; 33; 34; 35; 36; 37; 52 (stack); 56; 58; 59; 61; 62 (array); 63 (array); 64 (array); 65 (array); 66 (array); 67 (bit field); 68 (bit field); 70 (bit field); 72 (structure).
- CLR (.NET): 14; 26.
- JVM (Java Virtual Machine): 14; 26.

By OS

- Windows: 24; 29 (obfusction); 30; 38; 39; 40; 41; 42 (unknown cryptoalgorithm); 43; 44; 45 (simple copyprotection); 46 (toy-level web server); 47 (broken data compression algorithm); 48; 53; 69 (bit field); 73.
- Linux: 12; 24; 30; 38; 39; 40; 41; 42 (unknown cryptoalgorithm); 43; 45 (simple copyprotection); 46 (toy-level web server); 47 (broken data compression algorithm); 49; 53; 71 (structure); 73.
- Mac OS X: 30; 38; 39; 40; 42 (unknown cryptoalgorithm); 43; 46 (toy-level web server); 47 (broken data compression algorithm).
- Raspberry Pi Linux: 24; 35; 41.

Other

- Amateur cryptography: 25 (black boxes); 33; 34; 42 (unknown cryptoalgorithm); 44; 50 (4-byte XOR mask).

  This kind of cryptography is very different from professional one, nevertheless, it is highly popular in various types of software, so practicing reverse engineers has experience with it (or should have).
- Crackme/keygenme: 73.

About the website

Well, "challenges" is a loud word, these are rather just exercises.

Some exercises were in my book for beginners, some were in my blog, and I eventually decided to keep them all in one single place like this website, so be it.

The source code of this website is also available at GitHub: https://github.com/dennis714/challenges.re. I would love to get any suggestions and notices about misspellings and typos.

Exercise numbers

There is no correlation between exercise number and hardness. Sorry: I add new exercises occasionally and I can't use some fixed numbering system, so numbers are chaotic and has no

meaning at all.

On the other hand, I can assure, exercise numbers will never change, so my readers can refer to them, and they are also referred from my book for beginners.

Duplicates

There are some pieces of code which are really does the same thing, but in different ways. Or maybe it is implemented for different architectures (x86 and Java VM/.NET). That's OK.

Can I use Google?

It's up to you. I would first try to solve exercise without googling. If I would stuck, I would try to google some constants, text messages, etc.

Solutions

I made decision not to publish solutions.

Some computer science/programming books has solutions for exercises (like TAOCP), some has not (K&R, SICP, Niklaus Wirth - "Algorithms and Data Structures", Brian Kernighan/Rob Pike - The Practice of Programming" to name a few).

This website has been inspired by Project Euler and "the matasano crypto challenges" - and there are no solutions as well.

In my own opinion and experience, published solutions are killers to incentive (or motivation). When you see solution to the exercise, you lost an intellectual curiosity to solve it. It's like when magician reveals his tricks - he will loose all attention after the moment.

So please do not publish solutions on googleable forums/blogs/social networks, etc. On the other hand, you can freely discuss exercises with your friends and other people on the closed non-googleable private forums. Of course, I can't force anyone to do so, I just ask. Here is also explanation by one of "the matasano crypto challenges" website authors: https://www.reddit.com/r/haskell/comments/1fa8br/matasano_crypto_challenges_in_haskell/ca8em35.

If you are unsure if you solved some exercise correctly, just ask me by email (*dennis(a)yurichev.com*) (please also put exercise number in subject line like "exercise #123"). If you don't want to reveal your name/email, you may use BitMessage (my ID is **BM-2cUyrfGUDyjuUHyzAA63kq5NHAB2zMo4Q2**) and a temporary throw-away ID.

How can I measure my performance?

- As far as I can realize, If reverse engineer can solve most of these exercises, he is a hot target for head hunters (programming jobs in general).
- Those who can solve from ¼ to ½ of all levels, perhaps, can freely apply for reverse engineering/malware analysts/vulnerability research job positions.
- If you feel even first level is too hard for you, you may probably drop the idea to learn RE.

How can I learn Reverse Engineering?

Here is my book for beginners: http://beginners.re/.

Can I use these exercises for my teaching?

Yes, that is why they are (and this website as a whole) licensed under Creative Commons terms (CC-NC-ND), like my book about RE for beginners.

Updates/news

I'll add more exercises in future, if you interesting, you may subscribe to my blog and/or twitter: @yurichev and/or facebook.

Contact me

http://yurichev.com/contacts.html