Bomb

A binary bomb is on the server. A series of passwords will defuse the bomb. Determine the passwords that defuse it. You are not expected to defuse every single stage.

Each stage of the bomb should have its own section in the main document. Detail the tools employed to examine that stage, false starts or trails, breakthroughs, and eventual discovery of the solution. The documentation of your thoughts, processes, and attempts are vastly more important than how far you get in the bomb. A person reading the main document should be able to follow along and discover how to defuse the bomb themselves without having to patch the program.

Remember: The goal of reverse engineering is to distill and *reduce* the amount of detail by *increasing* the higher-level understanding of a device, protocol, or program. An explanation of "reverses the string" is good, an explanation of "swaps the the byte at `rbx + rsi` with the byte at `rbx + rdi` a total of `rcx` times" is not.

Group Work Note

For this project, the main document (`bomb.pdf`) may and should be worked on by all group members. This is the main technical document and work product; the individual student writeups are still required from each student.

Requirements

Area	Requirement
Documentation	The main technical document should be in `doc/bomb.pdf`.
Documentation	The student writeup should still be in `doc/`*`username`*`.pdf`.
Implementation	The source code of any helper scripts or custom-written programs must be submitted with the project.
Implementation	Project must be submitted by 1159EST on the due date specified.
Documentation	All documentation must be in PDF, TeX, or LaTeX format.
Documentation	When assembly is written, Intel syntax is required.
Implementation	The bomb itself must not be stored in the project.
Implementation	The project should be stored in your assigned VCS account, under the project name `bomb`.
Implementation	No third-party files/libraries/programs may be used unless signed off by the Program Managers or Instructors.
Implementation	Each logical portion or phase must be written in its own branch.
Implementation	Merge (without fast-forwarding) all branches to master branch and tag releases appropriately.
Implementation	The default branch to clone should be master.

DIcE Rubric

Documentation	Project Writeup	Demonstrates growth; Documents challenges and surprises; Summarizes lessons learned; etc.	10%
	Main Document	Does the documentation have clear and concise sections?	3%
		Does the documentation provide a clear general overview of the project?	2%
		Is the documentation easy to understand and follow?	10%
		Is Phase 1 work documented appropriately?	6%
		Is Phase 2 work documented appropriately?	6%
		Is Phase 3 work documented appropriately?	6%
		Is Phase 4 work documented appropriately?	6%
	Writing	Grammatical and spelling errors, formatting inconsistencies, etc.	11%
	Total		**60%**
Implementation	Version Control	Effective branching, merging, and tagging; Appropriate version control usage; one branch/merge per feature or requirement; no "active work" directly in master branch; No generated files in history; etc.	10%
	Total		**10%**
Execution	Requirements	Is Phase 1 defused?	3%
		Is Phase 2 defused?	4%
		Is Phase 3 defused?	4%
		Is Phase 4 defused?	4%
	Total		**15%**

Suggested Additional Features

Area	Feature	Bonus
Documentation	Submit all documentation in *well-written* TeX or LaTeX.	+5
Documentation	Add useful explanatory diagrams as part of the documentation, where appropriate.	+2
Execution	Each documented defused stage past the fourth.	+3