
BUILDING A SECURE COMPUTER SYSTEM

MORRIE GASSER



ACF2 is a trademark of Uccel Corp.

AOS is a trademark of Data General Corp.

DEC, PDP-11, VMS, SES/VMS are trademarks of Digital Equipment Corp.

IBM is a registered trademark of International Business Machines, Inc.

Intel, 80286, iAPX 432 are trademarks of Intel Corp.

Motorola, 68000 are trademarks of Motorola, Inc.

TOP SECRET is a trademark of CGA Software Products Group, Inc.

Unix is a trademark of American Telephone & Telegraph Co.

Copyright © 1988 by Van Nostrand Reinhold Company Inc.

Library of Congress Catalog Card Number 87-27838

ISBN 0-442-23022-2

All rights reserved. No part of this work covered by the copyright hereon may be reproduced or used in any form or by any means—graphic, electronic, or mechanical, including photocopying, recording, taping, or information storage and retrieval systems—without written permission of the publisher.

Printed in the United States of America

Designed by Beth Tondreau

Van Nostrand Reinhold Company Inc.

115 Fifth Avenue

New York, New York 10003

Van Nostrand Reinhold Company Limited

Molly Millars Lane

Wokingham, Berkshire RG11 2PY, England

Van Nostrand Reinhold

480 La Trobe Street

Melbourne, Victoria 3000, Australia

Macmillan of Canada

Division of Canada Publishing Corporation

164 Commander Boulevard

Agincourt, Ontario M1S 3C7, Canada

16 15 14 13 12 11 10 9 8 7 6 5 4 3 2 1

Library of Congress Cataloging-in-Publication Data

Gasser, Morrie, 1947–

Building a secure computer system.

Bibliography: p.

Includes index.

1. Computers—Access control. 2. System design.

I. Title.

QA76.9.A25G37 1988 005.8 87-27838

ISBN 0-442-23022-2

tributed systems. As networking becomes a more and more pervasive part of computing, the role of physical security will continue to diminish. In a large heterogeneous network, it is probably impossible to guarantee (and risky to assume) that any system other than your own is physically protected.

Personnel security covers techniques that an employer uses in deciding whom to trust with the organization's system and with its information. Most governments have procedures whereby a level of security clearance is assigned to individuals based on a personal background investigation and (possibly) additional measures such as polygraph examinations. These procedures allow the government to assign different degrees of trust to different people, depending on the needs of their particular job and the depth of their investigation. Personnel screening in industry is far less formal than in government, and people are usually given "all or none" access. Where selective access to information is required, it is determined on a case-by-case basis.

Procedural security covers the processes of granting people access to machines, handling physical input and output (such as printouts and tapes), installing system software, attaching user terminals, and performing countless other details of daily system administration.

Internal and external controls go hand in hand, and it is possible to trade off a control in one area for a control in the other. For example, even the most primitive multiuser systems today have password protection. The password mechanism is an internal control that obviates the need for external controls such as locked terminal rooms. In designing a secure system, we generally strive to minimize the need for external controls, because external controls are usually far more expensive to implement. Procedural controls are also notoriously error-prone, since they rely on people each time they are invoked.

3.2 THE SYSTEM BOUNDARY AND THE SECURITY PERIMETER

A *system* is a vague entity that comprises the totality of the computing and communications environment over which the developers have some control. Everything inside the system is protected by the system, and everything outside it is unprotected (fig. 3-1). What is important is not the generic definition of the term *system* but the definition as it applies in each particular case. In any effort to plan for security features, it is crucial to establish a clear understanding of the *system boundary* and to define the threats (originating outside the boundary) against which the system must defend itself. You cannot construct a coherent security environment without understanding the threats.

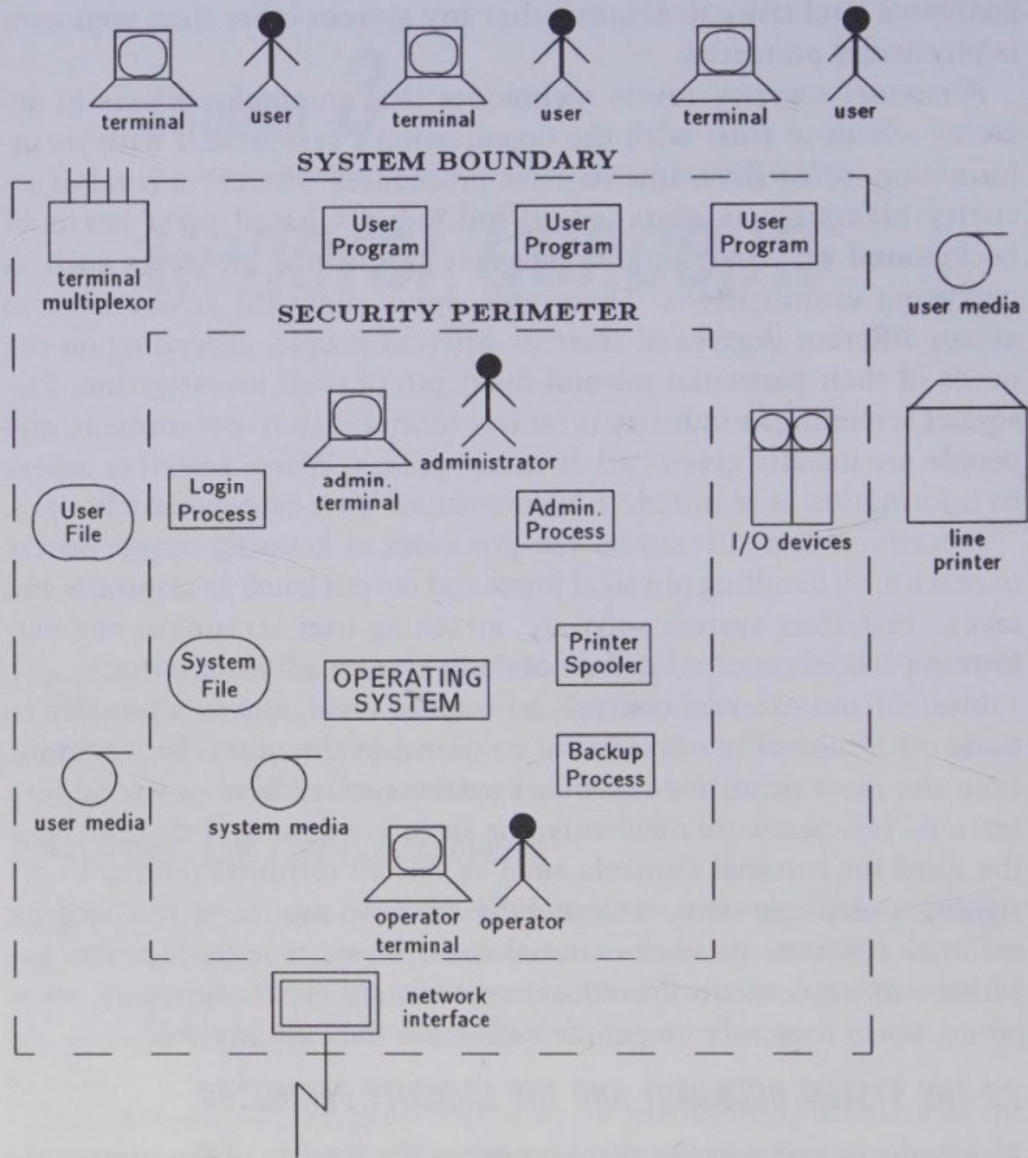


Figure 3-1. System Boundary and Security Perimeter. The entities collected inside the system are protected by the security-relevant portions within the security perimeter, as long as rules about access to the system from the outside are enforced by means of external security controls. Rules for access to the security perimeter interface are enforced by the internal controls implemented in the security perimeter.

Identifying the system boundary hinges on precisely specifying the interface between the system and the outside world. External security controls enforce this interface; and as long as those controls are in place, the internal controls protect information within the system against the specified threats. All bets are off, however, if something that should not be there bypasses the external controls and enters the system or if the system is threatened from the outside in an unanticipated way.

For example, a user might walk into the machine room and enter commands on the system console, or the system administrator might divulge a password to an outsider. These are failures of external controls that the system cannot defend against. It may, however, be able to defeat attempted incursions by unauthorized terminals, modems, or users who access the system remotely, as long as they are constrained to enter the system according to the rules of the system interface.

The components inside the system are of two types: those responsible for maintaining the security of the system (those, in other words, that are security-relevant), and all others. The security-relevant components implement the internal controls. Separating the two types of components is an imaginary boundary called the *security perimeter*. The operating system and computer hardware usually lie within the security perimeter; outside the perimeter are user programs, data, terminals, modems, printers, and the items that the system controls and protects. The nature of all components within the security perimeter must be precisely defined, because a malfunction in any one can lead to a security violation; in contrast, the nature of the components outside the perimeter is rather arbitrary, subject only to constraints enforced at the time they enter through the system boundary. A malfunction within the security perimeter has the effect of expanding the security perimeter to the system boundary, causing components previously outside the perimeter to become security-relevant.

Just as a precise interface must be identified across the system boundary, a well-defined interface across the security perimeter is crucial, as well. This interface is enforced by the security-relevant components. For example, the list of system calls in an operating system or the electrical specifications of a communications line are interfaces into the security perimeter. As long as the system boundary is enforced externally, the security perimeter will be maintained by the security-relevant components. In order to implement the components within the security perimeter, great care must go into defining a complete, consistent, and enforceable set of perimeter interface rules.