INTRODUCTION

At the United States Penitentiary, Leavenworth, Kansas, and the Federal Correctional Institution, Milan, Michigan, some fifty convicted felons are learning to be computer programmers. Inmates supplement academic training with experience by developing systems for the Veterans Administration, Department of Labor, and other Federal agencies. Programming services are sold to agencies at market prices to generate revenue for inmate salaries and other expenses. This custom software enterprise is, of course, managed and supervised by civil service staff.

A software house whose programmer staff are convicted felons must pay attention to the potential of computer fraud in its security program. But it is not enough to have a program deemed adequate by experts in the field. To the general public, the computer is still an awesome and mysterious device. Our operation, and indeed any computer operation susceptible to public opinion must, like Caesar’s wife, not only be above reproach but appear above reproach.

We learned this fundamental lesson when an allegation was made that inmates were using our computers to defraud IRS. Nothing of the sort had happened, nor could happen, under the stringent procedures in force at Leavenworth. We knew it; but we could not convince our critics, because the explanation was too complex.

This paper explains how we subsequently changed our security program to make it more clearly obvious to any observer that inmates could not misuse our resources. It should prove useful not only to other prison systems with similar rehabilitation programs, but to any organization affected by public opinion. Everyone may not have inmates as programmers, but must have some form of Achilles’ heel. The need to go beyond “secure” to “obviously secure” in your own sensitive area of security may be as great as was ours.

The IRS episode also resulted in criticism of the fundamental purpose of our program. There are some who argue that a skill so easily turned against society should not be taught to known felons. Although an inmate, once released from custody, is outside the scope of our security program, this paper would not be complete without an explanation of why we feel the program is a boon, rather than a bane, to society.

The Federal Prison System houses some 30,000 offenders in prisons located throughout the country. The Prison System has a duty beyond merely keeping them isolated from us. It is expected to afford inmates humane treatment and the opportunity for rehabilitation.

Humane treatment includes helping the inmate pass the endless hours without losing his sanity. This entails the need for meaningful work programs; ideally, these programs contribute also to rehabilitation by teaching a skill usable on the “outside.”

“Rehabilitating offenders” is a phrase not now heard in the field of corrections as often as before. Now we admit the best we can do is provide opportunities for self-improvement. So, education, vocational training, religious opportunities, and work programs are made available, on a voluntary basis, along with close counseling by social workers. Some inmates sincerely avail themselves of these opportunities; others merely “do their time,” and often it is these who come back to prison again and again.

The Leavenworth programming enterprise is only one of many self-supporting work/training programs operated by Federal Prison Industries, Incorporated. This wholly-owned government corporation was established by Congress in 1934 to operate as a private manufacturing concern, buying raw materials, producing products, and selling them to generate its operating revenue. Its management and supervisors are government civil servants; its workers, inmates. It was given freedom to develop and manufacture any product, but restricted to selling to other Federal agencies. It was capitalized with $4 million of machinery, equipment, and other assets with no provision for subsequent Federal funding.

Today, FPI, now operating under the trade name of “UNICOR,” has seventy factories and shops in thirty-two penal institutions throughout the country. It is virtually a conglomerate, with six product line divisions and forty industrial activities. UNICOR makes textiles; several lines of office furniture; footwear; wiring harnesses for missiles, tanks and aircraft; operates print plants, sign factories, microfilming shops engineering drafting shops, and other graphic arts facilities; and offers data entry, systems analysis, and computer programming, and other commodities and services. It employs 5,900 inmates, 600 civilians, and has annual sales
of over $90,000,000. It has been self-sustaining for the forty-three years of its existence. Unlike some government corporations, and not a few private ones, it has never needed another infusion of Federal funds.

Probably the most ambitious UNICOR effort to provide inmates usable skills was the establishment, in 1972, of the Leavenworth computer programming industry. As with most UNICOR industries, its workers have little or no prior experience. A year of classroom training, taught by certified teachers, is provided to prepare carefully selected inmates for the program. Graduates then work in a closely supervised environment to increase their skills with live work.

Students are selected from among a usually long list of volunteer applicants who have been recommended by their counselors. The potential trainee must have a high school diploma or GED certificate, perform at the tenth grade level in general math and language arts, and score a minimum of 45 out of a possible 90 points on the IBM Programmer Aptitude Test. A panel of custody, counselor, and industry personnel reviews such documents as the inmate's prior criminal record, pre-sentencing report, caseworker's progress reports, and overall incarceration history to reach a decision on each applicant's suitability for the program.

The course consists of thirteen four-week instructional modules designed to provide the equivalent of vocational school programmer training. It includes hands-on training with an NCR Century 200 computer with 64k bytes of main memory, dual disk storage units, a card reader, and line printer. The system software includes the operating system, assembler, and both FORTRAN and COBOL compilers. The course stresses COBOL, although an introduction to the assembler language and to FORTRAN are included. One module introduces IBM JCL and operating system utilities.*

A new course begins roughly every six months to start a new class of fifteen students on the road to becoming programmers. By the end of the year, dropouts and elimination by rigorous testing has dwindled the group to about 12. This number is adequate to maintain production staffing at about thirty programmers and eight analysts.

Production programmers are grouped into six or seven man teams, carefully staffed to maintain a balance of experienced and novice programmers. A non-working civilian supervisor, qualified by years of experience, heads each team. The leader is responsible not only for the quality and quantity of each team's production, but for each member's continued development and growth.

Additional staff required to operate the facility includes two Leavenworth-based analysts and two Washington-based analysts/marketing representatives. The Leavenworth analysts are augmented by a team of four inmate analysts each. An administrative officer and a computer room operator round out the staff.

The training program currently costs over $150,000 annually; operating costs for the production element bring the total to nearly $550,000. But with the potential for over 60,000 programmer hours per year, the industry is still capable of operating "in the black" despite the unusual need to extensively train every programmer.

When the program began in 1972, a customer was found conveniently close to Leavenworth: The Agriculture and Stabilization and Conservation Service in Kansas City. The ASCS sent Leavenworth detailed programming specifications which were reviewed and clarified by Leavenworth analysts before assignment to the teams. Leavenworth was soon turning out programs whose reject rate of 8.7 percent was deemed by the customer "much better than average." Inmates were honing their skills on live production work, and were beginning to find jobs in data processing upon release. By June 1976, twenty-four of the forty-eight released had become programmers or analysts in private industry.

It soon became apparent, however, that having one customer was not enough. To provide a continuous flow of work and keep all programmers busy most of the time, we needed several customers to fill in the gaps. Turning to other agencies, however, meant computer compatibility problems, since the Federal government, by design, has many different computers. Because of our proximity to ASCS/Kansas City, we had been able to use the customer's machine remotely; now we were looking at the distant market in Washington, D.C., where such arrangements might be impractical.

The obvious solution was commercial time-sharing. For each job, we would find a vendor who offered the same machine as our potential customer's, and an existing network to give us local access.

We started with Infonet and its UNIVAC 1108s. Inmates quickly learned to use time-sharing and the new tools it afforded, such as interactive programming. Infonet had not been chosen casually; it boasted more Federal customers than any other network. By contracting with these agencies, we could develop systems on the same computer as our customers thus eliminating compatibility problems.

That on-line access to computers shared with Internal Revenue, the Veterans Administration and others also posed the hazard of subversion was not overlooked. The initial, and as it turned out, only, terminal was placed in a secure room. Civilians made all telephone contacts, and observed inmates continuously as they used the machine. Plans for additional terminals included a PBX to be modified by the telephone company so that outgoing calls could be made only at the switchboard, in response to verbal requests by civilian staff. The switchboard would be located in the administrative area of the prison, outside the main security gates. All telephone lines would be disconnected at the switchboard at 4:00 P.M.

But we had not gone far beyond the training stage with this new tool before an unrelated incident occurred in another part of the prison that embroiled us in a national controversy, and eventually eliminated time-sharing as a resource of this program.

A CRISIS AND ITS IMPACT

Among Leavenworth's population at that time was an inmate busily defrauding the Internal Revenue Service.

* A copy of the curriculum is provided at the end of this paper.
There was nothing sophisticated in his operation; he was taking advantage of the commonly known fact that IRS does not crosscheck, except on a limited scale, the amount a taxpayer’s W-2 form shows was withheld with what his employer reports. To his phony tax returns showing money he had not earned, the inmate simply attached bogus W-2 forms showing taxes that had never been withheld. IRS processed the returns routinely, and sent a refund check to his wife on the outside.

The inmate was eventually caught and prosecuted. Although he was not connected with our program in any way, it was perhaps inevitable that it would be assumed our computer was being used in this scheme. Overnight, the story spread and grew by leaps and bounds. In major newspapers, wire service reports, and national television news broadcasts, it was implied, and sometimes stated outright, that inmates trained by the prison were using computers to crack the IRS security codes, file hundreds of bogus returns, and embezzle millions of dollars.

Computer security experts’ statements, such as “No computer system exists that cannot be penetrated” made it difficult for the press to understand our assertions that our inmates could not penetrate Infonet. No one wanted to hear that, as a practical matter, penetrating a system requires certain resources, such as a prior knowledge of the operating system and unlimited access to a terminal, neither of which our staff had. In the end, we suffered as much adverse publicity as had we been guilty.

FPI Security

The Leavenworth security program had to counter three distinct possibilities for computer fraud at the time of the IRS scandal:

1. Client System Penetration. An inmate might penetrate the Infonet system of controls and gain unauthorized access to sensitive data of the FPI client or other Infonet users. He might also modify copies of programs stored on Infonet so that they would produce something of value to himself at some future time. Likewise, he might penetrate the ASCS IBM 360 machine to which the Leavenworth computer was linked.

2. Client System Exploitation. An inmate might produce a program that, in addition to its intended function, secretly provided some benefit, such as issuing a check to the programmer, or indirectly benefited him by alerting the client’s operating system to subvert his security controls for later unauthorized use.

3. Unauthorized Use of Computer Resources. An inmate might use the Leavenworth computer or the Infonet system to directly produce something of value for himself, such as an official-appearing transcript; or to run a lottery.

These same threats exist in most data processing organizations.1 We felt at the time that, while we should do everything possible to reduce the probability that these threats would be carried out, we would be doing nothing more than any conscientious software house should be doing.

Our countermeasures at that time were based on controls that eliminated the personal freedom and access to equipment needed to carry out the threats previously described. A large measure of control was provided as a byproduct of being in a prison setting, and of having a staff comprised of trainees and “green” programmers.

An important factor in all publicized computer crimes to date, and indeed, in virtually all white collar crime, is the numerous privileges and freedoms enjoyed by the perpetrators.1,4 But the incarcerated Leavenworth inmate leads a life of almost constant supervision. He is deprived of the usual freedoms, not out of fear he will commit a computer crime, but because his custody is the responsibility of some prison employee at all times. He must have express permission to move from one area of the institution to the other, and must always be in a prescribed place at a prescribed time. At night, he is locked away in a cellblock separated from the computer room by numerous barred doors and gates.

During working hours, he has the freedom to be in the computer programming area, but still the requirements of custody apply. His supervisor must be constantly aware that he is present and doing his assigned job. No “outside” programmers are so closely observed.

Few would contest the statement that in the usual programming environment, managers are not enough in touch with their programmers to constantly know what they are doing. Thus a programmer runs little risk of discovery in appearing to be doing legitimate work while actually utilizing the computer for fraudulent purposes.

A close relationship between team leaders and programmer is assured at Leavenworth because the primary mission is different from that of the public sector software house. The supreme goal here is to build each inmate into an accomplished programmer so that he can succeed in the field upon release, rather than to make money for the enterprise. The team leader works closely with the inmate from the initial stages of specification interpretation, through flowcharting, debugging, and testing. This is a well-recognized concept for promoting program integrity, which in turn helps to ensure that a program does only what it is intended to do, and nothing more.1,5

Thus, the environment in which this software house operates provides a degree of control not usually found in the public sector. But additional, more direct security measures were also in force at the time of the IRS incident to counter the three main threats:

1. Management Control of Computer Runs—All run decks are submitted to the inmate’s team leader, who reviews each deck to ensure that the job is one assigned to the inmate, and that the JCL meets prescribed standards: Nothing is suppressed on the output listings, no permanent files are created, and only the test files prepared by civilian staff are accessed. Decks may be brought into the locked computer room only by the team leader, who also must retrieve the printed output.
Every listing is reviewed by the team leader before being turned over to the programmer.

2. Limiting the Possibilities—Live test data is never introduced into the prison, nor were inmates given access to live data when shared computers were being used. An unwritten policy at the time of the incident, since formalized, was to avoid accepting systems to be programmed that provided an opportunity for financial gain.

3. Personnel Selection—The screening process previously described for inmates being considered for entry into the program was designed not only to ensure that the man had the ability for the program, but, to the extent possible, that he had a sincere desire to change his life for the better. Admittedly, this judgment is subjective.

4. Independent Testing and Review of Programs—ASCS policy required that all programs be compiled and retested with their own test data against the program specifications and flowcharts. ASCS also ran each program through the Performance Monitor of METACOBOL, a product that helps in determining program integrity by listing unexecuted paragraphs.

Most professionals would agree that the above procedures, if consistently and conscientiously applied, would come as close to eliminating the possibility of computer fraud by Leavenworth inmates as possible, given the existence of the threats outlined. Indeed, a detailed analysis of this environment by the MITRE Corporation, well-known for its efforts in security on behalf of the Department of Defense, led them to state in their first report following the IRS incident: "The security procedures in use at this time are probably stronger than those used in most software development shops, and are more than adequate for the types of programs currently being developed and the current development system."

Nevertheless, we realized that this was not enough to prevent a repetition of the incident and its consequence of adverse publicity. An indirect threat existed for which we had no countermeasure: The public outcry against a danger more perceived than real. It could spell the end of our program. A combination of a general belief that with computers, anything is possible, and the prejudice that everyone who has previously committed a crime will do it again, given an opportunity, meant the issue of computer fraud was extremely sensitive in our situation.

The conclusion was that countermeasures against the three threats was not enough; we had to go further and eliminate as many of the threats themselves as possible. Under the guidance of the MITRE Corporation, we moved to reduce all threats to the issue of program integrity, and then to maximize our countermeasures against it.

This is being accomplished by isolating the development process from the client or target system. We withdrew from Infonet and will sever the connection with the ASCS machine, and begin using our own dedicated machine for system development. Thus, the only thing that will flow between the development facility and the client system is the program itself, which will be moved manually. The 'dedicated facility' is a well-known security technique, recognized by Department of Defense.5

The residual issue of 'program integrity' was addressed by strengthening the procedures already in effect. The technique of structured programming was adopted to help expose faulty or malicious code. A comprehensive, but common-sense, set of procedures and controls was also adopted to protect the program from the time it is "cleared" by the Leavenworth staff until installed on the target system. To further reduce the threat of program integrity, we have defined a set of formal guidelines against which to screen future client systems to reduce as much as possible the chance that an inmate will work on a program that could benefit him. These exclude: Programs that directly update data files that are used as the basis of financial transactions; programs in any way involved in the disbursement or transfer of funds to individuals, organizations, or accounts; and programs that access files which contain classified or extremely sensitive data that could be used for immediate or future financial gain.

All of these procedures are designed to protect society from the inmate programmer still incarcerated. They of course have no effect of the inmate programmer once he is released. The IRS fraud scandal raised another issue that still appears from time to time: Is it proper for the Federal government to turn known criminals into computer programmers? It is interesting to note that in a report following its investigation of the Leavenworth incident, a Congressional committee criticized the use of timesharing and advised against FPI involvement in systems that disburse funds, but approved of the program itself.6

TEACHING PROGRAMMING IN PRISONS

The basic argument against teaching computer programming is that it provides criminals with tools to commit more rewarding crimes, with less effort and risk than their previous adventures required. If the saying "once a criminal, always a criminal" were true, this argument would be unassailable. But it isn't; nearly two-thirds of all Federal releases have been shown to be "successes."**

Because we are dealing with something so complex and unpredictable as human beings, it is impossible to say just why some releases succeed in remaining out of prison while others fail. But we have clues. We know that people who commit crimes have the same basic desires as the rest of us; security, status, and so forth. The main difference seems to be that the criminal is more impatient, wanting his desires met immediately. And they often lack the training to fulfill their desires through legitimate means.

The FPI data processing program is a long-term, rigorous ordeal. An inmate must apply himself diligently for twelve months of training to avoid being dropped from the course. After completing the course, he has to put in two or three

* From "Success and Failure of Federal Offenders Released in 1970," A Bureau of Prisons study which defines a "success" as having no parole revocation or new sentence of 60 days or more.
years of further training in live production work, steadily improving his skills, in order to land a job upon release. Many never make it through the first few months of training.

Earlier release is not the motivating factor for these participants. The Parole Board gives no consideration to the programs an inmate completes. But, over the years, inmates have watched our participants released to well-paying, interesting jobs, and they hear through correspondence about the satisfaction these people get out of their new lives. This is the predominant reason given for wanting to participate in our program.

The program itself, then, is self-screening. We feel it attracts and keeps only the inmate who has made a firm commitment to play by the rules. The attitude of wanting instant, effortless success would not carry a trainee halfway through the first course. It is difficult to imagine someone still caught up in the syndrome of instant gratification having the patience to undergo such a grueling ordeal.

We feel the program is worthwhile because it provides an opportunity for those suited for a data processing career, who otherwise might continue robbing banks, or stealing securities to change their behavior. We have clear indications, after six years of operation, that our successful participants do go on to lead productive lives.

While it is difficult under existing procedures to follow up individuals released from custody, we can tell from the FBI's National Crime Information Center Computerized Criminal History File whether a releasee commits a new offense. From a continual review of these records, we know that no former participant has ever been charged with a computer crime. Moreover, these records indicate that 85 percent of our successful participants have had no further involvement of any kind with the law.

Even sophisticated statistical studies (which the above is not) dealing with the issue of relating a releasee's rehabilitation to his custodial treatment are subject to attack. There are simply too many factors that account for human behavior. Because the data, as positive as it appears, is inconclusive, one has to look elsewhere for additional indications that the society is better served by continuing the program. One such indicator is the attitude of private firms who hire our "graduates."

I hesitate to list these firms for fear of possibly embarrassing ex-inmates now leading productive lives in their employment. Key officials in these companies are of course aware of the men's backgrounds, but peers in some cases may not be. These companies would be recognized immediately by name. They are national and international in scope and have become prominent through hardened decision-making. These firms have decided that hiring ex-inmates as computer professionals is good business, and send the same teams that conduct campus interviews to our "campus." Recurrent hiring by the same firm is not uncommon.

It seems, in the final analysis, that judging whether teaching inmates to program computers is wise is, and should be, in the hands of these companies that hire our releases. It is they, after all, who stand to lose if our critics are right. If they felt that the program was dangerous, they could stop its effects, and the program itself, by simply refusing to hire participants. Instead, they urge us to continue; one firm tells us they get more from our experienced people than from fresh college graduates.

Computer programming in the Federal institutions will therefore go on. Our internal security procedures, and the attitudes of the inmate participants, are adequate to prevent inmates' misusing of this knowledge while in our custody. Because private sector employers have confidence in the program and its graduates, the data processing community as a whole should accept these individuals, not as criminals with new tools, but former offenders making a useful contribution to society.

REFERENCES


APPENDIX—CURRICULUM

The one-year training phase is designed around behavior/skill producing modules which permit:

Flexibility in teaching and learning in a penitentiary;

Recently measured goals which culminate in the PTU objectives, and,

Accurately defined functional areas of computer programming technology.

Thirteen modules are established as follows (each four weeks long, 160 classroom hours) with the objectives as stated.

Computer Programming Vocational Training by Behavioral Module

1. The Computer and its Program, People, and Peripherals. An introduction to computer programming as a technical science.
2. The Scientific Programmer. Learning and applying the FORTRAN programming language.
3. The Business Programmer. Learning and applying the COBOL programming language up through the use of tables and index-sequential files.
4. The Systems Programmer. Learning and applying the NEAT3, Level 1 programming language.
7. **All De-Buggers are not Household Insect Sprays.** The art of writing a computer program from specification to completed program, without waste or futility.

8. **The Super Business Programmer.** Advanced COBOL through internal sorts, conversion to IBM from NCR, and beginning JCL.

9. **NCR versus IBM—How Does it work?** IBM utilities and advanced JCL and relationships to NCR’s ways.

10. **Garbage In, Garbage Out.** Designing data systems, and test data.

11. **Critical Issues Seminar in Computer Science.** Discussion of current developments in the field, i.e., computer security, information privacy, etc. Use of outside guest lecturers such as NCR/IBM representatives, prior releasees, representatives from users organizations.

12. **I think you heard what I meant, but I’m not sure I said what you think you heard!** Documenting a computer program and current documentation standards required in the production phase.

13. **Where do we go from here?** Student evaluation. This module broken into four parts for use by instructors to evaluate students at 3, 6, and 9 month intervals, and at the end of the course of instruction for acceptance in the production phase.