

<h2>SAP-Passwort-Sicherheit</h2> <p>Dieser Artikel befasst sich mit verschiedenen Aspekten der SAP-Passwort-Sicherheit:</p> <ul style="list-style-type: none">• Schutz gegen Angreifer mit bestimmten Kenntnissen oder Berechtigungen• Schutz gegen unautorisierte Logins• Schutz gegen zu schwache Passworte, Hinweise zur Wahl sicherer Passworte <p>Außerdem enthält er Vorschläge für Erweiterungen des SAP-Standards.</p>	<h3>Über den Autor</h3> <p>Frank Dittrich ist seit 10 Jahren in der SAP-R/3-Entwicklung tätig.</p> <p>email: frank_dittrich@hotmail.com</p>
--	---

1 Analyse der Schutzmaßnahmen gegen Angreifer mit besonderen Kenntnissen oder Berechtigungen

1.1 Schutz des im Klartext eingegebenen Passwortes

Der Quelltext des Programms SAPMSYST, das während der Anmeldung sowie bei einer vom Benutzer vorgenommenen Passwort-Änderung ausgeführt wird, ist in Kundensystemen seit Release 4.0 weder mit den Transaktionen der Development Workbench noch mit dem für das Lesen von Quelltexten vorgesehenen ABAP-Befehl READ REPORT lesbar.

Auf diese Weise wollte SAP sicherstellen, dass Entwickler dieses für die SAP-System-Sicherheit so wichtige Programm nicht modifizieren können.

Zwar sind Modifikationen tatsächlich erschwert, jedoch für einen versierten Angreifer mit Entwickler-Berechtigung immer noch möglich.

Ein Entwickler kann also den Quelltext modifizieren, zum Beispiel um die von den Benutzern während der Anmeldung eingegebenen Passwörter abzufangen und zu speichern (Hinweis: zum Abfangen der vom Administrator für einen Benutzer vergebenen Passwörter sind im Internet Exploits veröffentlicht worden). Da der Quelltext des Programms SAPMSYST "geschützt" ist, sind Modifikationen schwerer zu erkennen.

Dieses Problem betrifft zunächst nur Entwicklungssysteme. Wenn jedoch kein (manuelles oder automatisches) Code Review für alle in die nachfolgenden Systeme zu transportierenden Objekte erfolgt, ist ein Entwickler in der Lage, auch die Sicherheit eines Produktivsystems zu kompromittieren.

Ein Entwickler könnte außerdem die - auf diese Weise aus Test- oder Entwicklungssystemen erlangten - Passwörter nutzen, um eine nicht autorisierte Anmeldung am Produktivsystem zu versuchen.

Um keine Anleitung für Manipulationen an SAPMSYST zu liefern, wird hier eine Methode vorgestellt, wie entsprechende Modifikationen erkennbar sind, auch ohne das Verfahren zum Lesen des Quelltextes unter Umgehung des vorhandenen Schutzes zu beschreiben.

Der Quelltext von Programmen ist in komprimierter Form in der Datenbank abgelegt: bis Release 4.6 in Tabelle D010S, seit Release 6.10 in der Tabelle REPOSRC.

Es ist möglich, die zum Programm SAPMSYST gehörenden Tabellen-Einträge in eine interne Tabelle einzulesen und unter Verwendung von SAP-Standard-Funktionsbausteinen einen Hashwert (zum Beispiel MD5) für den Inhalt zu bilden. Dieser Hashwert lässt sich mit dem zuletzt ermittelten Hashwert vergleichen.

Natürlich sind auf diese Weise bereits vorhandene Manipulationen nicht erkennbar, sondern nur solche, die seit der ersten Ermittlung des Hashwertes erfolgten.

Nach jedem Releasewechsel bzw. Basis Support Package, in dem der Quelltext des Programms SAPMSYST in einer neuen Version ausgeliefert wird, ist der Hashwert neu zu ermitteln.

1.2 Schutz der Passwort-Hashwerte

SAP legt die Hashwerte zum Passwort eines Benutzers - sowie zu den fünf vorhergehenden - in der Tabelle USR02 ab. Zusätzlich speichert SAP bei Passwort-Änderungen den Hashwert, das Datum und die Uhrzeit der Änderung sowie den Namen des letzten Änderers in der Tabelle USH02 (Änderungshistorie für Logon-Daten).

In einem Entwicklungssystem kann man nicht verhindern, dass ein Entwickler den Inhalt der Tabellen USR02 und USH02 mandantenübergreifend ausliest und so an die Passwort-Hashwerte aller Benutzer gelangt. Außerdem ist es einem Angreifer mit Entwicklerrechten möglich, den Hashwert in Tabelle USR02 durch einen anderen Hashwert zu ersetzen, um sich dann erfolgreich als anderer Benutzer anzumelden.

Auch in produktiven Systemen sind die Möglichkeiten solche Zugriffe zu verhindern begrenzt. Die Berechtigung zur Anzeige des Tabelleninhaltes per Transaktion SE16 kann zwar über das Berechtigungsobjekt S_TABU_DIS (Berechtigungsgruppe SC) gesteuert werden. Jedoch erfolgt die entsprechende Berechtigungsprüfung nicht, wenn ein Benutzer den zuvor in Transaktion SE16 von einem anderen Benutzer generierten Report zur Anzeige des Tabelleninhaltes im Hintergrund ausführt (Hinweis: in früheren Releases war sogar noch die direkte Ausführung dieses Reports im Dialog per Transaktion SE38 bzw. SA38 möglich).

Ein weiteres Problem ist, dass nahezu alle Transaktionen und Reports, welche die Gültigkeit eines in einem Dialogfeld eingegebenen Benutzers prüfen, die Tabelle mit `SELECT SINGLE * FROM USR02 ...` lesen. Daher ist es nicht praktikabel, alle Zugriffe auf die Tabelle USR02 per SQL Audit Log zu protokollieren, weil die `SELECT`-Anweisungen eines Programms zum Sammeln der Hashwerte in all den anderen `SELECT`-Anweisungen untergehen.

Außerdem genügt die Debugging-Berechtigung in einem Produktiv-System schon ohne die Berechtigung, Feldinhalte im Debugger zu verändern, um an die Passwort-Hashwerte beliebiger Benutzer zu gelangen.

1.3 Schutz der Hash-Funktion

SAP versucht, nicht legitime Aufrufe der im SAP-Kernel enthaltenen Funktion zur Ermittlung des Passwort-Hashwertes zu erkennen und zu verhindern. Ein als nicht legitim erkannter Aufruf dieser Funktion wird im System Log protokolliert, der entsprechende Nutzer gesperrt und vom System abgemeldet.

Versuche, mit Hilfe der Hashwerte per Crack-Programm an die Klartext-Passwörter zu gelangen, sind daher spätestens seit Release 4.6B nicht mehr so einfach möglich, wie dies früher der Fall war.

Dennoch gibt es in allen aktuellen Releases nach wie vor die Möglichkeit, SAP-Passwörter zu knacken. Der Autor hat zu Testzwecken ein Programm erstellt, das zu beliebigen Passwörtern die Hashwerte ermittelt und mit den Hashwerten aus Tabelle USR02 vergleicht. Dieses Programm eignet sich zum Knacken von SAP-Passwörtern für alle Releases bis einschließlich 6.x.

Mit einem handelsüblichen PC (CPU: AMD Athlon XP 2100+) ließen sich pro Minute ca. 7,5 Millionen Kombinationen aus Benutzername und Passwort prüfen und die ermittelten Hashwerte mit denen aus Tabelle USR02 vergleichen. Ein Angreifer braucht also nur einen aktuellen PC, um ca. 10 Millionen Kombinationen aus Passwort und Benutzername pro Minute auszuprobieren.

So sind mit Hilfe von im Internet verfügbaren Wortlisten und einigen auch aus Crack-Programmen für andere Systeme bekannten "mangling rules" nicht nur triviale, sondern auch eine Reihe vermeintlich sicherer Passwörter zu knacken.

1.4 Schwächen des Hash-Algorithmus

Der früher verwendete Hash-Algorithmus (Codeversion 'A' als Feldinhalt von USR02-CODVN für das aktuelle Passwort, USR02-CODV1 - USR02-CODV5 für die 5 vorhergehenden Passwörter) wies gravierende Schwächen auf:

- nur die ersten 6 Zeichen des Benutzernamens gingen in den Hashwert ein. Ein Angreifer musste also nicht für jeden Benutzernamen zu allen möglichen Passwörtern die Hashwerte ermitteln, um sie mit den in Tabelle USR02 gespeicherten Hashwerten zu vergleichen, sondern nur einmal für beliebige Benutzer, deren Namensbeginn in den ersten 6 Zeichen identisch ist – dies ist insbesondere dann kritisch, wenn zum Beispiel die Personalnummern als SAP-Benutzernamen verwendet werden.
- aus dem Hashwert ließen sich bestimmte Vorhersagen über das Passwort treffen, zum Beispiel zur Länge des Passwortes.
- es kam zu häufigen, nicht vorhersagbaren Hash-Kollisionen, so dass unter Umständen mit Trivial-Passwörtern eine Anmeldung möglich war, obwohl das tatsächlich vom Benutzer verwendete Passwort nicht trivial war.

Aus diesem Grund sollten die Passwörter aller Nutzer geändert werden, für die das aktuelle Passwort noch nach diesem alten Algorithmus gebildet wurde (Hinweis: da dieser Algorithmus schon seit Jahren nicht mehr für neu vergebene Passwörter verwendet wird, dürften entsprechende Einträge allenfalls noch in selten genutzten Mandanten oder z.B. für RFC-User zu finden sein).

Wegen dieser Schwächen führte SAP einen neuen Hash-Algorithmus (Codeversion 'B') ein. Genaue Informationen zum Hash-Algorithmus liefert SAP nicht (Hinweis: zu den verfügbaren Informationen siehe OSS-Hinweis 2467).

Auch der neue Hash-Algorithmus ist trotz einiger Verbesserungen nicht frei von Schwächen. In der Dokumentation ist bis einschließlich Release 4.6C die Information zu finden, dass in einem Passwort alle Zeichen verwendbar sind, die "an einem Terminal eingegeben werden können" [1]. Die Dokumentation weist lediglich darauf hin, dass nicht zwischen Groß- und Kleinschreibung unterschieden wird. Dies schränkt die Benutzer bei der Wahl guter Passwörter erheblich ein.

In der Dokumentation zum Release 6.20, SP 19 [2] findet sich die Information, dass alle Buchstaben von A-Z, alle Ziffern von 0-9 sowie Interpunktionszeichen verwendbar sind. Es findet sich allerdings kein Hinweis auf eventuelle Probleme, wenn im Passwort Zeichen enthalten sind, die nicht zum 7bit-ASCII-Zeichensatz gehören.

Eine durch den Autor vorgenommene Analyse der erzeugten Hashwerte hat jedoch ergeben, dass in einem Passwort nur folgende Zeichen verwendet werden sollten, um Hashwert-Kollisionen (identische Hashwerte für verschiedene Passwörter) zu vermeiden:

- die 26 Buchstaben ABCDEFGHIJKLMNOPQRSTUVWXYZ (Kleinbuchstaben sind gleichwertig)
- die 10 Ziffern 0123456789
- die 31 Sonderzeichen !"#%&'()*+,-./:;<=>?@[_`{|}~
- das Leerzeichen.

Folgenden Zeichen sollten nicht in Passwörtern verwendet werden, obwohl sie vom System ohne Warnung akzeptiert werden:

- das Caret-Zeichen ^

- nicht zum 7bit-ASCII-Zeichensatz gehörende Zeichen, wie z.B.
 ¡¢£¥ ¦§¨©ª«¬®¯°±²³µ¶·¸¹º»¼½¾¿ÀÁÂÃÄÅÆÇÈÉÊËÌÍÎÏÐÑÒÓÔÕÖ×ØÙÚÛÜÝÞßàáâãäåæçèéêëìíîïðñòóôõö÷øùúûüýþÿ (sowie die entsprechenden Kleinbuchstaben).

Denn für den Hashwert zu einem Passwort ist es gleichgültig, welches dieser Zeichen an einer bestimmten Stelle im Passwort vorkommt. Je häufiger die zuletzt genannten Zeichen in einem Passwort vorkommen, desto mehr Hashwert-Kollisionen mit anderen Passwörtern gibt es. Das heißt im Extremfall, dass alle (mehr als $3 \cdot 10^{14}$) achtstelligen Passwörter, die nur aus dem Caret-Zeichen und aus nicht zum 7bit-ASCII-Zeichensatz gehörenden Zeichen gebildet sind, pro Benutzername zum gleichen Hashwert führen.

Ein Angreifer braucht also nur für eines dieser vielen Passwörter den Hashwert zu ermitteln. Wenn der Benutzer zum Beispiel ß³\$^µ€£ als Passwort gewählt hat, kann der Angreifer sich mit ÄÄÖÖÜÜßß oder sogar ^^^^^^^ als Passwort anmelden.

Selbst das Ausprobieren aller Kombinationen, die aus wenigen 7bit-ASCII-Zeichen und vielen nicht zum 7bit-ASCII-Zeichensatz gehörenden Zeichen bestehen, ist für einen Angreifer wegen der großen Zahl von Hashwert-Kollisionen noch lohnend:

- um alle Passwörter auszuprobieren, die nur 3 7bit-ASCII-Zeichen außer ^ enthalten, braucht ein Angreifer nur 1,7 Minuten, wenn man von 10 Millionen überprüften Kombinationen pro Minute ausgeht.
- um alle Passwörter mit 4 solchen Zeichen zu überprüfen, braucht ein Angreifer ca. 2,4 Stunden.
- erst für die Prüfung aller Passwörter mit 5 Zeichen aus dem 7bit-ASCII-Zeichensatz außer ^ und 3 sonstigen Zeichen braucht ein Angreifer ca. 5,5 Tage. Wenn er jedoch nur die Passwörter prüft, in denen 5 Buchstaben A-Z oder Ziffern 0-9 sowie 3 nicht zum 7bit-ASCII-Zeichensatz gehörende Zeichen enthalten sind, reduziert sich der Aufwand auf weniger als 6 Minuten.

So wie die Verwendung des Caret-Zeichens oder die Verwendung von nicht zum 7bit-ASCII-Zeichensatz gehörenden Zeichen im Passwort zu Hashwert-Kollisionen führt (z.B. identischer Hashwert für die - durch einen Angreifer auch ohne Hashwert-Kollision leicht zu ermittelnden - Passwörter SÖHNE123 und SÜHNE123 oder KANÄLEXY und KANÜLEXY), kommt es auch bei Benutzernamen, die nicht nur aus 7bit-ASCII-Zeichen bestehen, zu einem identischen Hashwert, wenn sie das gleiche Passwort vergeben haben.

Falls also Benutzer P_MÜLLER und Benutzer P_MÖLLER das gleiche Passwort verwenden, ist auch der in Tabelle USR02 gespeicherte Hashwert für diese Benutzer identisch.

Zwar ist in diesem Fall die Wahrscheinlichkeit von Hashwert-Kollisionen deutlich geringer als beim alten Algorithmus (CODVN 'A'). Dennoch kann ein Angreifer unter Umständen die Anzahl auszuprobierender Kombinationen von Benutzername und Passwort reduzieren.

Offensichtlich werden also vor der Ermittlung des Hashwertes alle im Benutzernamen und im Passwort enthaltenen Zeichen, die nicht zum 7bit-ASCII-Zeichensatz gehören, durch das Caret-Zeichen ^ ersetzt. Grund für diese Ersetzung ist vermutlich, dass die erzeugten Hashwerte unabhängig von der jeweils verwendeten Code Page sein sollen. Anderenfalls könnten Benutzerstammdaten nicht per Transport zwischen beliebigen Systemen ausgetauscht werden.

Auch mit dem alten Algorithmus gab es übrigens - zusätzlich zu den auftretenden "zufälligen" Hashwert-Kollisionen - diese für einen Angreifer vorhersagbaren Kollisionen. Allerdings führten nicht zum 7bit-ASCII-Zeichensatz gehörende Zeichen hier zum gleichen Hashwert wie ein Apostroph an der gleichen Stelle.

2 Analyse der Schutzmaßnahmen gegen nicht autorisierte Anmeldung

2.1 Erkennung von fehlgeschlagenen Dialog-Anmeldeversuchen

Um nicht autorisierte Anmeldungen zu verhindern, ist es nötig, fehlgeschlagene Anmeldeversuche zu erkennen. Anderenfalls kommt ein Angreifer früher oder später durch simples Ausprobieren verschiedener Passwörter zum Ziel.

Das System protokolliert Fehlversuche bei der Dialog-Anmeldung, außerdem legt es die Anzahl fehlgeschlagener Versuche seit dem letzten erfolgreichen Login im Benutzer-Stammsatz ab. Problematisch daran ist, dass in der Regel nur System-Administratoren, nicht die jeweiligen Benutzer über solche Fehlversuche informiert werden. Ein System-Administrator kann allerdings kaum beurteilen, ob der tatsächlich autorisierte Benutzer sich bei der Anmeldung nur vertippt hat, oder ob sich ein anderer unberechtigt Zugang zum System verschaffen wollte.

Allenfalls darf vermutet werden, dass der tatsächlich autorisierte Benutzer den Anmeldeversuch unternommen hat, wenn unmittelbar danach eine erfolgreiche Anmeldung von der gleichen IP-Adresse aus erfolgt.

Die in Transaktion SM04 angezeigte Terminal-ID eignet sich nur bedingt, da sie je nach Frontend-Betriebssystem leicht manipulierbar ist. Andererseits ist beim Einsatz von Thin Clients und WBT-Servern eine eindeutige Identifikation des jeweils genutzten Thin Client auch nicht ohne weiteres möglich, da das SAP-System in diesem Fall nur die Terminal-ID und IP-Adresse des WBT-Servers kennt.

Nur der Benutzer weiss, ob er sich beim vorhergehenden Anmeldeversuch vertippt hat. Sinnvoll wäre also eine Möglichkeit, die Benutzer über eventuell fehlgeschlagene Anmeldeversuche seit dem letzten erfolgreichen Login zu informieren. Diese Möglichkeit ist im SAP-Standard jedoch nicht vorgesehen.

Auch im Login-Userexit ist so ein Hinweis nicht ohne größeren Aufwand möglich. Zu diesem Zeitpunkt ist - aus Sicht des SAP-Standards - die erfolgreiche Anmeldung bereits abgeschlossen und der Zähler für fehlgeschlagene Anmeldeversuche auf 0 zurückgesetzt.

Eine Variante, diese Funktion dennoch im Login-Userexit bereitzustellen, besteht darin, auf Datenbank-Ebene einen Trigger zu definieren, der bei jeder Erhöhung des Wertes von USR02-LOCNT entsprechende Informationen (Zähler Falsch-Anmeldungen, Datum, Uhrzeit) in einer kundeneigenen Tabelle speichert.

Diese kundeneigene Tabelle könnte während des Login-Userexits ausgewertet werden, um bei Bedarf eine entsprechende Warnmeldung an den Benutzer zu senden und die Felder der kundeneigenen Tabelle anschließend zu initialisieren.

2.2 Sperrung des Benutzers nach fehlerhaften Anmeldeversuchen

Normalerweise wird ein Benutzer nach der im Profile-Parameter login/fails_to_user_lock festgelegten Anzahl fehlerhafter Anmeldeversuche gesperrt. Dies ist notwendig, da ansonsten durch Ausprobieren diverser Passwörter irgendwann eine nicht autorisierte Anmeldung möglich ist.

Unter <http://www.securityfocus.com/bid/7007/discussion/> ist jedoch ein Hinweis veröffentlicht, dass diese Sperrung nicht erfolgt, wenn ein Angreifer das Programm sapinfo nutzt, um damit beliebige verschiedene Passwörter auszuprobieren. Damit erhöht sich gegenüber direkten Anmeldeversuchen im Dialog die Chance, unerkannt an das Passwort eines anderen Benutzers zu gelangen.

Es wäre also zu überprüfen, ob dieses Programm einem Benutzer zur Verfügung steht, der es eigentlich nicht benötigt.

2.3 Aufforderung zur Passwort-Änderung

Um einen erfolgreichen Angriff auf Passwörter zu erschweren, müssen die Benutzer die Passwörter in regelmäßigen Abständen ändern, so dass die Zeitspanne, die zum Knacken eines Passwortes benötigt wird, möglichst größer ist als die Dauer der Benutzung dieses Passwortes.

Die Gültigkeitsdauer eines vom Benutzer vergebenen Passwortes kann mit Hilfe des Profile-Parameters `login/password_expiration_time` festgelegt werden. Ist die entsprechende Anzahl von Tagen seit der letzten Passwort-Änderung vergangen, wird der Nutzer bei der Anmeldung gezwungen, ein neues Passwort zu wählen.

Es gibt keinerlei Vorwarnung und auch nicht die Möglichkeit, das eigentlich abgelaufene Passwort ausnahmsweise noch ein weiteres Mal zu verwenden, damit der Benutzer sich in Ruhe ein neues Passwort ausdenken kann. Dadurch erhöht sich einerseits die Gefahr, dass die Benutzer Trivial-Passwörter wählen. Andererseits steigt die Wahrscheinlichkeit, dass das neue Passwort vergessen wird.

Hilfreich für den Benutzer wäre es also bereits 2 Arbeitstage vor der wirklich fälligen Passwort-Änderung, einen entsprechenden Hinweis zu bekommen. Da im SAP-Standard eine solche Möglichkeit nicht vorgesehen ist, bietet es sich an, diese Funktionalität im Login-Userexit bereitzustellen.

Hier kann der Inhalt des Profile-Parameters `login/password_expiration_time` ermittelt und geprüft werden, ob die Gültigkeitsdauer eingeschränkt ist. Zusätzlich wird das Datum der letzten Passwort-Änderung aus Tabelle `USR02` gelesen.

Unter Berücksichtigung des Fabrikkalenders wird mit Hilfe entsprechender SAP-Standard-Funktionsbausteine berechnet, wie viele Arbeitstage das Passwort noch gültig ist. Ist eine bestimmte (entweder im Userexit hart kodierte oder per SET/GET-Parameter ID im Benutzerstamm definierte) Anzahl Tage unterschritten, wird eine Meldung ausgegeben, die den Benutzer auf die bevorstehende Passwort-Änderung hinweist.

Diese Erweiterung kann sowohl die Qualität der gewählten Passwörter verbessern als auch den Aufwand, der durch vergessene Passwörter entsteht, reduzieren.

2.4 Anmerkungen zur Qualität von Passwörtern

Am sichersten sind möglichst lange Passwörter, die aus wirklich zufälligen Kombinationen von Zeichen bestehen, wobei Buchstaben, Ziffern und Interpunktionszeichen in zufälliger Abfolge vorkommen sollten. Außerdem sollte kein Passwort aus anderen (ehemals verwendete Passwörter, Passwörter für andere Systeme) abgeleitet werden können.

Allerdings kann sich kaum jemand eine große Menge solcher Passwörter merken.

So kommt es dazu, dass oft Passwörter verwendet werden, die für einen Angreifer leicht zu erraten sind. Leicht heißt in diesem Zusammenhang: mit Kenntnis der Hashwerte unter Zuhilfenahme eines Crack-Programms zu knacken.

Die Sicherheit der gewählten Passwörter hängt also entscheidend davon ab, inwieweit die Nutzer informiert wurden, wie sichere Passwörter zu wählen sind, und ob sie sich an entsprechende Empfehlungen halten.

2.4.1 Schlechte Passwörter

Beispiele für schlechte Passwörter:

- Vornamen

- Firmennamen
- Markennamen
- Familiennamen, insbesondere Namen bekannter Persönlichkeiten (z.B. Sportler, Künstler, Politiker)
- Fachbegriffe
- Abkürzungen
- Ortsnamen
- Wörter irgendeiner Sprache
- simple Modifikationen aller bisherigen Beispiele (anhängen eines beliebigen Zeichens; anhängen von zwei Ziffern; voranstellen eines beliebigen Zeichens; austauschen einzelner Buchstaben, z.B. S->\$, S->5, I->!, Z->2; Buchstaben eines Wortes in umgekehrter Reihenfolge usw.)
- Buchstabenfolgen oder Ziffernfolgen, wie z.B. 23456789, DEFGHIJK
- Folgen von auf der Tastatur nebeneinander liegenden Tasten, z.B. SDFGHJKL
- Verwendung der System-ID, des Mandanten, der Terminal-ID usw. im Passwort
- zu kurze Passwörter, egal wie kompliziert sie scheinbar sind. Wenn man von ca. 10 Millionen Tests pro Minute ausgeht, lässt sich jedes bis zu 4 Zeichen lange Passwort durch einen Angreifer in weniger als 2,5 Minuten ermitteln, jedes bis zu 5 Zeichen lange Passwort ist in weniger als 3 Stunden geknackt, jedes bis zu 6 Zeichen lange Passwort kann innerhalb von 7,5 Tagen geknackt werden. Um alle bis zu 7 Zeichen langen Passwörter auszuprobieren, bräuchte ein Angreifer dagegen schon ca. 1,5 Jahre.
- Verwendung mehrerer nicht zum 7bit-ASCII-Zeichensatz gehörender Zeichen im Passwort
- durch simple Modifikationen aus ehemals verwendeten Passwörtern gebildete Passwörter

Auch die in der Dokumentation [2] genannten Beispiele für gültige Passwörter kann ein Angreifer, der Zugriff auf die in Tabelle USR02 gespeicherten Hashwerte hat, mit vertretbarem Aufwand knacken:

- frtas - um alle gültigen Passwörter, die aus 5 Buchstaben (A-Z) bestehen, auszuprobieren, genügen $(26*26*26-26)*26*26 = 11.863.800$ Versuche. Wenn man von 10 Millionen Versuchen pro Minute ausgeht, sind also alle Passwörter, die aus 5 Buchstaben bestehen, in wenig mehr als einer Minute geknackt.
- jullo=6 - dieses Passwort ist vermeintlich relativ sicher. Jedoch kommt jullo in einer im Internet verfügbaren Wortliste mit mehr als 3 Millionen "Wörtern" vor, von denen etwa 350.000 aus maximal 6 Zeichen bestehen.

Durch Anhängen von 2 beliebigen Zeichen aus dem 7bit-ASCII-Zeichensatz ergeben sich ca. 1,6 Milliarden Kombinationen, die ein Angreifer in weniger als 3 Stunden ausprobieren kann. Allerdings ist davon auszugehen, dass ein Angreifer zuerst eine Reihe von simpleren Modifikationen bzw. Modifikationen an weniger umfangreichen Wortlisten ausprobiert, so dass der tatsächliche Aufwand zum Knacken dieses Passwortes etwas höher sein dürfte.

- 3bar - dieses Passwort ist durch simples Voranstellen einer Ziffer vor ein Wort, das in etlichen Wortlisten vorkommt, gebildet. Daher wird es mit Sicherheit schnell geknackt. Selbst wenn ein Angreifer stattdessen alle vierstelligen Passwörter ausprobiert, die nur aus Buchstaben A-Z und Ziffern 0-9 bestehen, braucht er nur $(36*36*36-36)*36 = 1.678.320$ Versuche. Jedes auf diese Weise gebildete Passwort ist also innerhalb von 10 Sekunden geknackt.

2.4.2 Gute Passwörter

Eine Technik, sichere Passwörter zu wählen, besteht darin, sie aus ausreichend langen Phrasen oder Sätzen zu bilden, die sich leicht merken lassen, jedoch für andere schwer zu erraten sind. Wenn man nun beispielsweise das jeweils erste oder letzte Zeichen jedes Wortes sowie die Interpunktionszeichen verwendet und dann noch einzelne Buchstaben durch Ziffern oder Sonderzeichen ersetzt (z.B. S->5, H->#, oder T->+), erhält man ein Passwort, das kaum noch mit vertretbarem Aufwand zu knacken ist.

Ein potentieller Angreifer müsste dann schon Vermutungen über die Sprache sowie innerhalb der Sprache vorkommende Buchstabenhäufigkeiten anstellen, um die Menge der zu prüfenden Passwörter überhaupt irgendwie einschränken zu können.

Allerdings gibt es auch ungeeignete (zu bekannte) Phrasen. So kommt zum Beispiel das aus "To be or not to be" abgeleitete Passwort 2bon2b in einer im Internet verfügbaren sehr kurzen Liste von aus Phrasen oder Sätzen abgeleiteten Passwörtern vor. So führen selbst weitere Modifikationen, wie zum Beispiel das Anhängen von 2 Sonderzeichen, immer noch zu einem Passwort, das ein Angreifer durchaus knacken kann. Ebenso sollten keine geläufigen Sprichwörter verwendet werden.

Auch ein Satz wie "Gabi erwartet heute einen indischen Mitarbeiter." [3] ist wenig geeignet, um aus den Anfangsbuchstaben ein Passwort zu bilden.

2.4.3 SAP-spezifische Einschränkungen bei der Wahl von Passwörtern

Bei der Wahl eines gültigen Passwortes gibt es im SAP-System im Vergleich zu anderen Systemen diverse Einschränkungen. Nicht alle Einschränkungen sind dokumentiert, und nicht immer sind die in der Dokumentation erwähnten Einschränkungen auch gültig.

Zu den bekannten Einschränkungen zählen:

- Mindestlänge 3 Zeichen
- maximale Länge 8 Zeichen
- keine Unterscheidung zwischen Groß- und Kleinbuchstaben
- das erste Zeichen darf nicht ! oder ? sein - laut SAP-Dokumentation darf unter den ersten für die minimale Passwortlänge erforderlichen Zeichen kein Leerzeichen sein [1][4] bzw. das erste Zeichen des Passwortes kein Leerzeichen sein [2]. Dennoch ist es (zumindest zu Release 6.10) möglich, ein Passwort zu wählen, das in den ersten 3 Stellen Leerzeichen enthält oder sogar mit einem Leerzeichen beginnt.
- die ersten drei Zeichen des Passwortes dürfen nicht identisch sein
- das Passwort darf nicht PASS oder SAP* lauten
- das Passwort darf nicht mit den letzten fünf Passwörtern des Benutzers identisch sein

Die in der Dokumentation (für alle Releases bis einschließlich 6.20) ebenfalls genannte Einschränkung, dass die ersten 3 Zeichen des Passworts nicht in der gleichen Reihenfolge im Benutzernamen vorkommen dürfen, gilt z.B. im Release 6.10 nicht. So kann ein Benutzer FDTEST durchaus das Passwort FDT wählen.

In der per Transaktion SM30 pflegbaren Tabelle USR40 (Tabelle für verbotene Kennworte) können Passwörter und generische Werte (mit * als Platzhalter für eine beliebige Zeichenfolge und ? als Platzhalter für ein beliebiges einzelnes Zeichen) eingetragen werden. Passwörter, die den in Tabelle USR40 vorhandenen Einzelwerten oder Mustern entsprechen, werden vom System nicht akzeptiert. Weitere Einschränkungen lassen sich über Profile-Parameter regeln (s. Anhang bzw. [5]).

Inwieweit die vorgenommenen Einstellungen tatsächlich zu einer Verbesserung der Sicherheit beitragen, ist fraglich. Jede Einschränkung erschwert es dem Benutzer, ein gültiges Passwort zu finden. Andererseits sind selbst Festlegungen wie minimale Passwortlänge von 8 Zeichen oder Mindestanzahl enthaltener Ziffern = 2 keine Garantie dafür, dass wirklich sichere Passwörter gewählt werden.

Beispiele für Regeln, die - falsch eingesetzt - eher zur Verringerung der Passwort-Sicherheit beitragen oder andere ungünstige Nebeneffekte haben:

- zu geringe Anzahl Tage, nach denen das Passwort geändert werden muss - kaum ein Nutzer wird in der Lage sein, sich jede Woche ein neues sicheres Passwort auszudenken und zu merken, erst recht nicht jeweils verschiedene Passwörter für mehrere SAP-Systeme und Mandanten. Es werden also vorzugsweise Trivial-Passwörter bzw. Passwörter nach einem bestimmten Schema gewählt, so dass ein Angreifer die jeweils neu vergebenen Passwörter leicht knacken kann, nachdem er ein einmal verwendetes Passwort geknackt hat.

- Aufnahme umfangreicher Wortlisten in die Tabelle USR40 - da die Tabelle bei jeder Passwort-Änderung komplett eingelesen werden muss, wirkt sich eine zu große Anzahl Einträge negativ auf die Performance aus. Mit Hilfe dieser Tabelle lassen sich keinesfalls alle Passwörter ausschließen, die ein Angreifer innerhalb weniger Minuten mit einem Crack-Programm ausprobieren kann. Allenfalls einige wenige Trivial-Passwörter lassen sich so verhindern.
- Festlegung zu komplizierter Passwort-Regeln - ohne vorhergehende Schulung, wie man sichere Passwörter wählt, die man sich dennoch leicht merken kann, vergessen Benutzer häufig ihre Passwörter, insbesondere nach längerer Abwesenheit. Oder die Passwörter werden irgendwo aufgeschrieben oder auf dem PC gespeichert. Werden zu enge Restriktionen festgelegt, verringert sich auch für einen Angreifer die Menge der per Brute Force auszuprobierenden Passwörter.

2.5 Überprüfung, ob sichere Passwörter gewählt werden

In früheren Releases gab es (im Programm SAPMS01J) die Möglichkeit, die gewählten Passwörter zu überprüfen und zu einfache Passwörter abzulehnen. Wegen der Möglichkeit eines Missbrauchs gibt es seit Release 4.0 nur noch die bereits erwähnte Tabelle USR40, mit der sich Trivial-Passwörter ausschließen lassen.

Der Report RSUSR003 prüft nur, ob die Benutzer SAP*, DDIC, SAPCPIC und EARLYWATCH in irgendeinem Mandanten die allgemein bekannten Standardpasswörter haben. Diese simple Prüfung ist unzureichend, da zum Beispiel nach Kompromittierung des DDIC-Accounts beliebige Manipulationen im System vorgenommen werden können.

Es gibt daher keine im SAP-Standard vorgesehene Methode zu gewährleisten, dass die Benutzer wirklich sichere Passwörter wählen, die sich selbst mit Kenntnis der Hashwerte nicht knacken lassen. Da ein Zugriff auf die Hashwerte zumindest in einem Entwicklungs- oder Testsystem nicht ausgeschlossen werden kann, sind die bestehenden Möglichkeiten einer Überprüfung unbefriedigend. Selbst wenn ein Benutzer weiß, wie sichere Passwörter zu wählen sind, besteht noch die Gefahr, dass zufällig ein Passwort vorkommt, welches durch simple Modifikation eines finnischen oder japanischen Wortes gebildet werden kann.

Von gecrackten Passwörtern eines Testsystems geht zwar noch keine unmittelbare Gefahr für die Sicherheit von produktiven Systemen aus. Mittelbar besteht diese Gefahr aber durchaus:

- eventuell wird das Testsystem in regelmäßigen Abständen aus Daten des Produktivsystems neu aufgebaut.
- einige Nutzer verwenden vermutlich in Test- und Produktivsystemen oder in verschiedenen Mandanten die gleichen Passwörter (wenn die Benutzernamen gleich sind, lassen sich solche identischen Werte durch Vergleich der Hashwerte erkennen).
- selbst wenn die Nutzer in verschiedenen Systemen unterschiedliche Passwörter verwenden, kann ein Angreifer aus dem aktuellen Passwort und aus den ehemaligen Passwörtern im Testsystem möglicherweise Annahmen über im Produktivsystem verwendete Passwörter treffen (Beispiel: Passwörter T11ADM01, T11ADM02 und T11ADM03 im Testsystem, P11ADM03 im Produktivsystem) und so durch Ausprobieren erfolgreich sein. Wenn dabei die maximale Anzahl Fehlversuche bis zum Sperren des Users nicht überschritten wird, hat er gute Chancen, unentdeckt zu bleiben.

Ein Programm zum Knacken von SAP-Passwörtern (nach dem Vorbild der für andere Systeme verfügbaren Crack-Programme wäre daher ein nützliches Tool für den Systemadministrator, um die Sicherheit von SAP-Passwörtern zu überprüfen.

Dabei müssen jedoch Datenschutz-Belange beachtet werden. Außerdem muss sichergestellt werden, dass nicht durch Missbrauch eines solchen Programms neue Risiken entstehen, statt zu einer Verbesserung der Qualität von Passwörtern beizutragen. Das Crack-Programm sollte deswegen keinesfalls in einem für normale Nutzer oder Entwickler zugänglichem System zum Einsatz kommen. Stattdessen sollten die Hashwerte in einem separaten System analysiert werden.

3 Wünschenswerte Erweiterungen des SAP-Standards

Laut Aussage von SAP ist für zukünftige Releases die Verwendung von SHA-1 zur Bildung der Passwort-Hashwerte geplant. Es bleibt zu hoffen, dass die dann 160 Bit langen Hashwerte in einer separaten Tabelle gespeichert werden. Dann ließen sich alle Zugriffe protokollieren, die nicht lesend mit `SELECT SINGLE ... WHERE MANDT = SY-MANDT AND BNAME = SY-UNAME` bzw. mit gleicher WHERE-Klausel ändernd (nach einer Passwort-Änderung) auf diese neue Tabelle zugreifen.

Außer bei Mandantenkopien dürften dann nur noch Passwort-Änderungen durch Systemadministratoren in den Protokollen auftauchen, so dass ein nicht legitimer Zugriff leicht zu erkennen wäre.

Außerdem sollte sowohl die Beschränkung der maximalen Passwortlänge auf 8 Zeichen als auch die fehlende Unterscheidung von Groß- und Kleinbuchstaben entfallen. Dann wäre es viel leichter, sichere Passwörter zu wählen.

Wenn aus Kompatibilitätsgründen auch mit dem neuen Hash-Algorithmus Kollisionen bei Verwendung von nicht zum 7bit-ASCII-Zeichensatz gehörenden Zeichen bestehen bleiben, sollte in der Dokumentation explizit darauf hingewiesen werden, welche Zeichen besser nicht in einem Passwort verwendet werden sollten.

Jeder Benutzer sollte im SAP-Standard (also ohne kundenspezifische Erweiterungen im Login-Userexit) über eventuelle fehlgeschlagene Anmeldeversuche informiert werden.

Ebenso sollte im SAP-Standard eine Warnung implementiert werden, die es einem Benutzer rechtzeitig vor der fälligen Passwort-Änderung ermöglicht, sich ein neues sicheres Passwort auszudenken.

4 Literatur

[1] Logondaten pflegen
(Online-Dokumentation Release 4.6C)
<http://help.sap.com/sapdocu/core/46c/helpdata/DE/52/67119e439b11d1896f0000e8322d00/content.htm>

[2] Kennwort
(Online-Dokumentation Release 6.20, SP 19)
<http://help.sap.com/sapdocu/netweaver/webas/620sp19/helpdata/DE/73/69ec1355bb11d189680000e829fbbd/content.htm>

[3] Original-Beitrag in Newsgroup de.alt.sysadmin.recovery:
Message-ID 20030508222808.1370.1.NOFFLE@mliss.myfqdn.de
Einreichung für Newsgroup de.alt.netdigest:
Message-ID 08e0a7e1acfa61a1cbbdeaa2fd83e7c5@die.wuer.de
<http://groups.google.de/groups?selm=08e0a7e1acfa61a1cbbdeaa2fd83e7c5@die.wuer.de>

[4] Anmelde- und Kennwortschutz im SAP-System
(Online-Dokumentation zum Release 6.10)
<http://help.sap.com/sapdocu/netweaver/webas/610/helpdata/DE/52/6717ed439b11d1896f0000e8322d00/content.htm>

[5] Login-Parameter
(Online-Dokumentation zum Release 6.10)
<http://help.sap.com/sapdocu/netweaver/webas/610/helpdata/DE/22/41c43ac23cef2fe10000000a114084/content.htm>

5 Anhang

Profile-Parameter zur Festlegung von erlaubten Passwörtern:

- *login/min_password_lng*
minimale Passwort-Länge (Werte < 3 sind nicht zulässig)
- *login/min_password_diff* (neu zu 6.10)
minimale Anzahl Zeichen, in denen sich das neue Passwort vom vorhergehenden Passwort unterscheiden muss (Diese Prüfung erfolgt natürlich nur, wenn der jeweilige Nutzer sein eigenes Passwort während des Anmelde-Dialoges ändert. Zusätzlich wird simples Verschieben - z.B. altes Passwort GJP\$EUB7, neues Passwort P\$EUB7GJ - nicht als Abweichung gewertet.)
- *login/min_password_digits* (neu zu 6.10)
minimale Anzahl im Passwort enthaltener Ziffern (0-9)
- *login/min_password_letters* (neu zu 6.10)
minimale Anzahl im Passwort enthaltener Buchstaben (A-Z)
- *login/min_password_specials* (neu zu 6.10)
minimale Anzahl im Passwort enthaltener sonstiger Zeichen

Andere Profile-Parameter, die auch Einfluss auf die Sicherheit von Passwörtern haben:

- *login/password_expiration_time*
Anzahl Tage, nach denen spätestens eine Passwort-Änderung erfolgen muss
- *login/password_max_new_valid*
Anzahl Tage, für die das (vom Administrator vergebene) Passwort neu eingerichteter User gültig bleibt
- *login/password_max_reset_valid*
Anzahl Tage, für die ein vom Administrator zurückgesetztes Benutzerpasswort gültig bleibt.
- *login/fails_to_session_end*
Anzahl Fehlversuche bei der Anmeldung bis zur Beendigung der SAPGUI-Session
- *login/fails_to_user_lock*
Anzahl Fehlversuche bis zur Sperrung des Benutzers
- *login/failed_user_auto_unlock*
Kontrolle der Entsperrung von durch Falschanmeldungen gesperrten Benutzern
- *login/no_automatic_user_sapstar*
Mit diesem Parameter lässt sich steuern, ob eine Anmeldung als Benutzer SAP* mit Passwort PASS möglich ist (Wert 0, Default) oder nicht (Wert 1, empfohlen), wenn kein entsprechender Benutzer-Stammsatz im angegebenen Mandanten existiert.